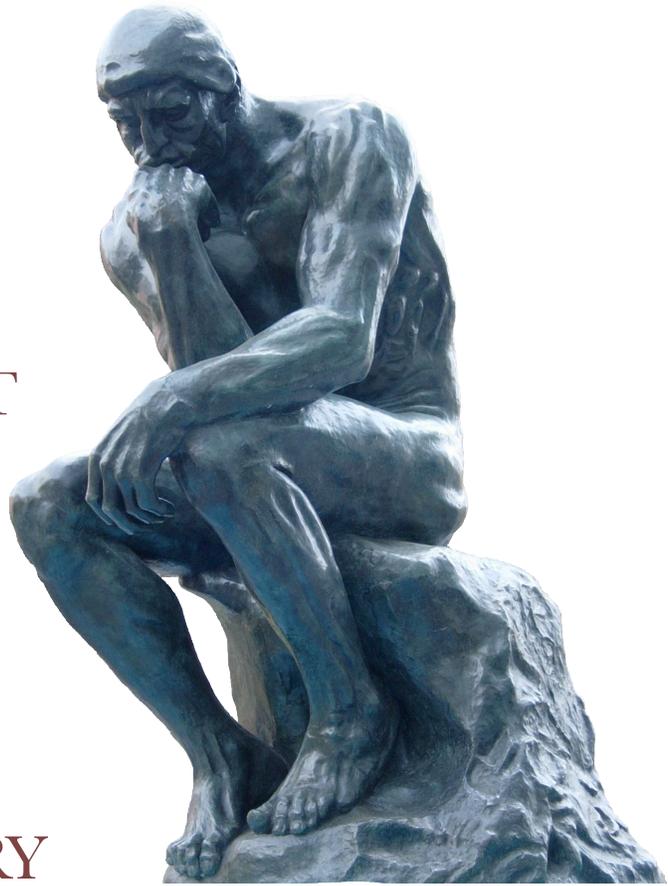


Third Annual Cyber Security and Information Infrastructure Research Workshop

May 14-15, 2007

TOWARDS
COMPREHENSIVE
STRATEGIES THAT
MEET THE
CYBER SECURITY
CHALLENGES OF
THE 21ST CENTURY



Frederick Sheldon, Axel Krings, Seong-Moo Yoo, and
Ali Mili (Editors)

OAK RIDGE NATIONAL LABORATORY

MANAGED BY UT-BATTELLE FOR THE DEPARTMENT OF ENERGY



CSIIRW07: Cyber Security and Information Infrastructure Research Workshop

May 14-15, 2007

Oak Ridge National Laboratory, Oak Ridge, Tennessee, USA

Frederick Sheldon, Axel Krings, Seong-Moo Yoo and
Ali Mili (Editors)

Towards comprehensive strategies that meet the cyber security challenges of the 21st century

As our cyber infrastructure grows ever larger, more complex and more distributed, the systems that compose it, become not only more prone to failures, but more prone to security violations. At the same time, as our cyber infrastructures take more life-critical, mission-critical and infrastructure-critical roles, the stakes of failure-free and violation-free operation grow ever larger. Furthermore, as perpetrators become more sophisticated, it becomes increasingly difficult to build adequate protection defenses. The combination of increased vulnerability, increased stakes, and increased threats make cyber security one of the most important emerging challenges in the evolution of modern cyber infrastructure design and deployment.

Though they may play a significant role in an overall strategy, piecemeal solutions to security vulnerabilities are not a match for the magnitude of the challenge at hand. As cyber infrastructure dependents, how do we know we can trust what we see? If we consider the viewpoints of all cyber infrastructure stakeholders then must we maximize the satisfaction of the policy makers, system administrators, resource consumers while anticipating the perpetrators' options?

The goal of the workshop is to challenge, establish and debate a far-reaching discussion that broadly and comprehensively outlines a strategy for cyber security that is founded on sound technologies that meet the challenge of cyber security (beyond a Maginot line mentality). The characteristics that we should see in such a strategy should include:

- √ Better understanding of existing and emerging threats.
- √ Advances in insider threat detection, deterrence, mitigation and threat elimination.
- √ Ensuring the continuing security, survivability and dependability of our critical infrastructures.
- √ Guaranteeing availability of time-critical scalably secure systems.

- √ Observable/ measurable/ certifiable security effects, rather than hypothesized causes.
- √ Quantitative metrics of security, that enable us to specify security requirements, formulate security claims, and certify security properties.
- √ Solutions that provide a measure of assurance against known and unknown (though perhaps pre-modeled) threats (e.g., cryptography, QKD, building scalable secure systems, information provenance and assurance).
- √ Mission fulfillment, whether or not security violations have taken place (rather than mitigating all violations indiscriminately) and whether or not they affect the system's mission (including situational understanding and attack attribution).

Last year's theme was: Beyond the Maginot line. To pursue a military analogy, we must shift our focus away from winning battles, towards a strategy for winning the war. Our ultimate goal is to elevate trust in the mission and its underlying critical infrastructures.

CSIIR Workshop 2007

Table of Contents

Chapter	Title and Author(s)	Page
	Preface	2
	Table of Contents	4
1	NICIAR: Pursuing Disruptive Technologies for Information Assurance Keynote Speaker: <i>Carl E. Landwehr, Chief of the Cyber Access and Protection Division of the Disruptive Technology Office under the Director of National Intelligence</i>	6
2	Denial of Service Games <i>C. Dingankar and Richard R. Brooks, Holcombe Department of Electrical and Computer Engineering, Clemson University</i>	7
3	Trusted Passages: Managing Distributed Trust Needs of Emerging Applications <i>Mustaque Ahamad, Martin Carbone, Greg Eisenhauer, Jiantao Kong, Wenke Lee, Bryan Payne, Karsten Schwan and Ramesh Viswanath, School of Computer Science, Georgia Institute of Technology</i>	18
4	Worm Spread in Scale-Free Networks: A Model Using Random Graph Theory (Aka: Network Pathogen Spread Model Using Random Graphs) <i>Christopher Griffin, Penn State Applied Science, Clemson University</i>	30
5	Robustness and Adaptation in Information Ecosystems <i>Stephen Racunas, Stanford University and Christopher Griffin Penn State Applied Science</i>	39
6	Deploying Statistical Anomaly Detection to Improve Cyber Security <i>Greg Shannon, CounterStorm, Inc.</i>	47
7	Application of Risk Management Principles in Information Technology Permitting Decision Makers to Target Funding for Security Investments <i>Dr. Martin A. Carmichael, Chief Information Officer, The Rader Network, Colorado Springs, CO</i>	57
8	Secure Coding Initiative <i>Robert Seacord and Jason Rafail, CERT, Carnegie Mellon University</i>	70
9	Security in the Complex of Dependability <i>Tacksoo Im and John McGregor, Department of Computer Science, Clemson University</i>	85
10	Integrated Hardware/Software Security Support <i>Richard Brooks and Sam Sanders, Holcombe Department of Electrical and Computer Engineering, Clemson University</i>	97
11	Models of Models: Digital Forensics and Domain-Specific Languages <i>Daniel Ray and Phillip Bradford, Department of Computer Science, The University of Alabama</i>	108
12	Automatic Generation of Certifiable Space Communication Software <i>Johann Schumann and Ewen Denney, Robust Software Engineering Group, NASA Ames Research Center</i>	121

13	Long Term Vision for IT Security Keynote Speaker: <i>Scott Studham, CIO for the Computing and Computational Science Directorate, Oak Ridge National Laboratory</i>	131
14	TCIP: Trustworthy Cyber Infrastructure for the Power Grid Keynote Speaker: <i>William Sanders, Director, Information Trust Institute, Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign</i>	138
15	Measuring Dependability as Mean Failure Cost <i>Ali Mili, Computer Science Department, New Jersey Institute of Technology, and Fredrick Sheldon, Computational Science and Engineering, Oak Ridge National Laboratory</i>	152
16	Survivability in Wireless Networks: A Case for Overhead Reduction <i>Axel Krings, Department of Computer Science, University of Idaho</i>	168
17	The Layered Security Model and its Representation using Bigraphs to Analyse Critical Infrastructure (Aka: Using Bigraphs to Model System Architecture) <i>Clive Blackwell, Information Security Group, Royal Holloway, University of London</i>	190
18	Early Detection and Containment of Worm Epidemics <i>Tom Chen, Department of Electrical and Engineering, Southern Methodist University</i>	207
19	Managing Multiple Perspectives on Trust <i>Clifford Neuman, Director, USC Center for Computer Systems Security, Information Sciences Institute, University of Southern California</i>	214
20	Toward Mitigating Denial of Service Attacks in Power-Constrained Sensor Networks <i>Ortal Arazi and Hairong Qi, Department of Electrical and Computer Engineering, The University of Tennessee</i>	222
21	Tolerating Change in a Secure Environment: A Visual Perspective <i>Shawn Bohner, Department of Computer Science, Virginia Tech</i>	239
22	Two Complementary Views for Anomaly-based Intrusion Detection <i>Chin Tser Huang, Department of Computer Science and Engineering, University of South Carolina</i>	254
23	Detection of Undesirable Insider Behavior <i>Joseph Calandrino, Department of Computer Science, Princeton University and Steven McKinney, Department of Computer Science, North Carolina State University, and Frederick Sheldon, CSED, Oak Ridge National Laboratory</i>	294
24	A Systems Development and Implementation Study for 21 st Century Software and Security (Aka: Quantifying the Vulnerability of Tactical Data Networks) <i>Andrew Loebel, James Nutaro, and Teja Kuruganti, Computational Science and Engineering, Oak Ridge National Laboratory, Rajanikanth Jammalamadaka, Electrical and Computer Engineering Department, University of Arizona</i>	309
25	Physical Protection in Mobile Constrained Devices (Aka: Open Problems Pertaining to RFID Anti-Cloning and Some Observations) <i>Benjamin Arazi, Department of Computer Engineering and Computer Science, University of Louisville</i>	322
26	EMS Cyber Security (Aka: Standards and Interoperability Have Exposed Energy Management System Commands and Data to Cyber Attack) <i>Dennis Holstein, OPUS Publishing and Jay Wack, TecSec, Inc.</i>	333

NICIAR: Pursuing Disruptive Technologies for Information Assurance

Carl E. Landwehr, Ph.D.

Chief of the Cyber Access and Protection Division of the Disruptive Technology Office under the Director of National Intelligence

Keynote Abstract

Despite substantial research investments in the past, the U.S. (and global) cyber infrastructure remains highly vulnerable to a wide range of attacks. The Disruptive Technology Office (DTO) is a research organization in the Office of the Director of National Intelligence (ODNI). Its mission is to incubate revolutionary research and development activities that address needs arising from all agencies under the ODNI. DTO originates and manages advanced research and development programs in a variety of domains that will have fundamental impact on future operational needs and strategies of its customers and demand substantial, long-term venture investment to spur risk-taking. The National Intelligence Community Information Assurance Research (NICIAR) program is one such program. NICIAR has initiated two thrusts: (1) technologies to improve accountability in NIC systems, leading to more accountable information flow, and (2) technologies to improve defenses of large scale systems against attacks. This talk will provide some background on the motivation for these thrusts and the recently initiated research projects within the program.

Speaker Bio

Carl E. Landwehr, Ph.D., is Chief of the Cyber Access and Protection Division of the Disruptive Technology Office under the Director of National Intelligence, on assignment from his position as Senior Research Scientist at the University of Maryland's Institute for Systems Research. He is developing new strategies and directions for the programs in this division with the goal of achieving dramatic change in the overall trustworthiness of National Intelligence Community systems. He was recently named Editor-in-Chief of IEEE Security & Privacy Magazine. He has been active internationally as the founding chair of IFIP WG 11.3 (Database and Application Security) and is also a member of IFIP WG 10.4 (Dependability and Fault Tolerance). Dr. Landwehr has received Best Paper awards from the IEEE Symposium on Security and Privacy and the Computer Security Applications Conference. IFIP has awarded him its Silver Core, and the IEEE Computer Society has awarded him its Golden Core. His research interests span many aspects of trustworthy computing, including high assurance software development, understanding software flaws and vulnerabilities, token-based authentication, system evaluation and certification methods, multilevel security, and architectures for intrusion tolerant systems.

Denial of Service Games

C. Dingankar *Student* and R. R. Brooks *Associate Professor*
Holcombe Department of Electrical and Computer Engineering
P.O. Box 340915
Clemson University
Clemson, SC 29634-0915
Email: cdingan@clemson.edu, rrb@acm.org

We use combinatorial game theory to analyze the dynamics of Distributed Denial of Service (DDoS) attacks on an enterprise. An initial approach and in depth analysis of DDoS problems can be found in [5]. The attacker (Red) launches a DDoS on the distributed application (Blue). Both Red and Blue play an abstract board game defined on a capacitated graph, where nodes have limited CPU capacities and edges have bandwidth constraints. Our technique provides two important results that aid in designing DDoS resistant systems:

- (i) It quantifies the resources an attacker needs to disable a distributed application. The design alternative that maximizes this value will be the least vulnerable to DDoS attacks.
- (ii) When the attacker can not harvest enough zombies to satisfy the limit in (i), we provide near optimal strategies for reconfiguration of the distributed application in response to attempted DDoS attacks. While it is intractable to find the optimal strategy for typical applications, since our problem is P-Space complete (worse than NP-complete) [9], our approach finds a strategy that is within a known constant offset of the optimal solution [3, 9].

Our analysis starts by finding the feasible network configurations for Blue that satisfy its computation and communications requirements. The min-cut sets [1] of these configurations are the locations most vulnerable to packet flooding DDoS attacks. Red places “zombie” processes on the graph, which can consume network bandwidth. Given enough zombies, Red can win the game by disabling all possible Blue configurations. When the number of Red zombies is limited, the graph structure is used to define a board game. Red moves attempt to break Blue communications links. Blue reconfigures its network to re-establish communications. We analyze this board game using the theory of surreal numbers [2, 6, 7, 8]. If Blue can make the game “loopy” (i.e. move to one of its previous configurations), it wins [3]. If Red creates a situation where Blue can not successfully reconfigure the network, it wins. We use “thermograph” based strategies, originally developed to analyze endgames for Go, to find near optimal reconfiguration regimes [2, 3, 4].

We define a simple two player game to be played on a computer network. The “physical” graph (computer network) is represented by a directed graph structure (EG) with N nodes:

$$\begin{aligned} EG &= \{EV, EE\} \\ EV &\rightarrow \{\text{vertices (nodes) with known CPU bandwidth}\} \\ EE &\rightarrow \{\text{directed edges with known communications bandwidth}\} \end{aligned} \tag{1.1}$$

The local communications bandwidth available when two processes are placed on the same node is infinite.

The players are:

- *Blue* is a distributed application on the network. A set of programs consume CPU resources on the “physical” nodes. For each pair of programs, there is a known communications bandwidth requirement. These constraints define a “logical” graph. The set of “feasible configurations” is the set of mappings of logical nodes to physical nodes, where the logical graph’s CPU and communications needs are satisfied by the physical graph.
- *Red* is an attacker that places zombie processes on physical graph nodes. These processes can send network traffic over the physical edges to consume network resources. If the Red zombies consume enough communications bandwidth to make the physical graph unable to satisfy one of the logical graph’s constraints, Blue’s configuration is disabled.

To determine the set of feasible configurations for Blue, we use the directed graph structure (BG) with M nodes:

$$\begin{aligned}
BG &= \{BV, BE\} \\
BV &\rightarrow \{\text{nodes representing distributed programs with CPU requirements}\} \\
BE &\rightarrow \{\text{edges representing communications bandwidth needed between two programs}\}
\end{aligned} \tag{1.2}$$

The set of feasible configurations for blue is the set of mappings of BV onto EV that satisfy these two classes of constraints:

- *Node Capacity Constraints:* The sum of the CPU requirements for the set of nodes from BV assigned to each element of EV is less than or equal to the CPU bandwidth of that element.
- *Edge Capacity Constraints:* For each element be_{ij} of BE connecting two elements of BV (bv_i and bv_j), where bv_i (bv_j) is mapped to pv_i (pv_j)¹ the max-flow [1] on EG from pv_i to pv_j must be greater than equal to the bandwidth requirement of bv_{ij} . If pv_i and pv_j are the same node, the value of the max-flow is infinite.

The set of feasible blue configuration mappings is denoted as:

$$BC = \{BC_1, BC_2, \dots, BC_L\} \tag{1.3}$$

Red disrupts a Blue configuration by placing zombies so as to either:

- Attack node capacities – Red places zombies on a node pv_i hosting one or more Blue processes. If Red zombies consume enough CPU cycles, the performance of the Blue processes on pv_i becomes unacceptable. The associated feasible configuration is disabled.
- Flood arcs – Red places zombies on nodes that do not host Blue processes. These nodes produce network traffic that consumes communications bandwidth on edges in EE . If the capacity of the min-cut of EG corresponding to an element be_{ij} of BE in the current configuration falls beneath the value be_{ij} , the associated feasible configuration is disabled.

The node capacity attack is rather trivial and not very interesting. Also, it is typically difficult for Red to compromise the servers used by Blue. When this does occur, Blue can also easily detect Red's presence and disinfect the server. We therefore concentrate our analysis on flooding attacks.

To determine the set of zombies needed by Red, we:

- (i) Calculate the mincut for each element of BE in BC_i [1],
- (ii) Find the amount of blue slack capacity (BS) at the mincut,
- (iii) Find the expected number of blue packets dropped for a given volume of zombie traffic, and
- (iv) Find the volume of red traffic needed to make the number of blue packets dropped be greater than the slack capacity of the mincut.

This gives us the red traffic (RT) we need to generate in order to flood an element of BE .

$$RT = C - \frac{\lambda}{1 - BS/\lambda} \tag{1.4}$$

where,

λ packets is the Blue (legitimate) traffic

C is the capacity of the physical arc to be attacked

Zombie placement is done by looking at the maxflow between elements of EV . If the maxflow to a node in the mincut of an element of BE is greater than the value in step (iv), that node is a candidate for zombie placement. To find the minimum number of zombies required, we look for zombie nodes that can disable more than one element of BC . The smallest set of zombies needed to disable all elements of BC quantifies the resistance of Blue to DDoS attacks.

If the attacker does not have enough zombies to disable all Blue configurations, Blue can reconfigure to recover from the DDoS attack. This defines a simple board game, where:

- Blue starts the game.
- Each player is allowed one move at a time.
- Once Red places a zombie on a node it cannot move that zombie until its next turn.
- Blue reconfigures by migrating a single process from one element of PE to another.

For the moment, we give both Blue and Red perfect knowledge of each other's configurations. Red tries to force Blue into a position where it cannot recover by transitioning to another element of BC . Blue tries to find a "loopy" game [3], where it can always return to a previous configuration. If Blue succeeds in

¹ $pv_i \in PV$ and $pv_j \in PV$

creating the “loopy” game where Red cannot escape from the loop, it wins since it can recover from any attack.

Let’s consider an example. Each player’s moves are denoted by surreal numbers. Blue is the Left player. The moves of the Blue player are the configurations it can reach from the current configuration. Our example has 3 Blue nodes and 6 physical nodes. We find that 10 Blue configurations [1, 2, 3.....10] satisfy all the constraints and we require 5 zombies to disable all ten Blue configurations. But Red can only use 2 zombies at a time. Each combination of 2 zombies is a Red move denoted by A, B, C, ... Table 1 shows the details of the game.

Blue Configuration	Reconfigure	Zombie Move	Zombies	Disrupt Blue Configurations
1	2, 3, 4	A	3, 5	1, 2, 7
2	1, 5, 6	B	1, 6	3, 4, 5
3	1, 7	C	3, 4	2, 6, 9
4	1,10	D	1,5	8, 9, 10
5	2, 8, 9			
6	2, 9			
7	3, 10			
8	5			
9	5, 6			
10	4, 7			

Table 1: Details of an example game

We assume that Blue starts the game with Blue configuration 2 and in response to that move Red can either choose A or C (as both the zombie moves – A and C can disable Blue configuration 2). Let’s assume that Red chooses A. The game tree for the above example would look like Figure 1. If we have to denote these game trees in the form of surreal numbers [2, 6, 7, 8, 9], the first level would look like {1, 5, 6 | A, B, C, D} where 1, 5, 6 are moves Blue can make (from Blue configuration 2) and A, B, C, D are moves Red can make. As we can see in Figure 1a at level 2 if Blue chooses configuration 6 it will loose but if it chooses configuration 5 it has a stronger chance of winning. If Blue can form a loop as shown in Figure 1b [2-5-2] and [5-8-5] Blue will never loose as it can keep looping between these two nodes and Red will never be able to disrupt it as it will keep reconfiguring itself. In 1a we can see that Red has led Blue into a position where Blue is unable to reconfigure itself so Red wins the game.

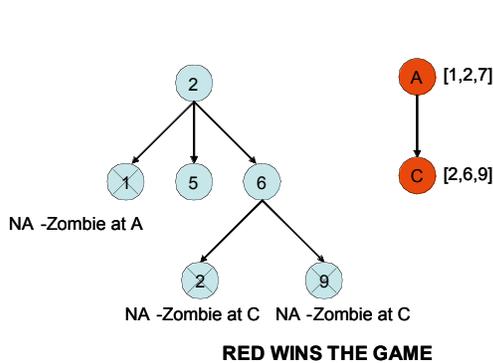


Figure 1a: Game trees
NA – Not Allowed

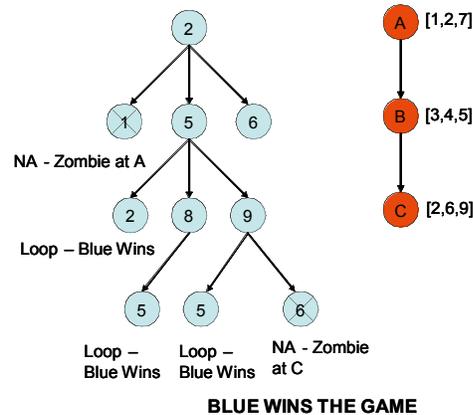


Figure 1b: Game trees

In practice, any given enterprise relies on multiple distributed processes. Similarly, an attacker can not expect to destroy all of the processes used by the enterprise at any point in time. The attacker will try to maximize the number of processes it can disable at any point in time. This situation describes a “sum of games” problem [2], where Blue and Red alternate moves. At each turn, the player chooses a game (process) and a move to make in that game. This problem has been shown to be P-Space complete [9] but

Berlekamp has used thermographs, like the one in Figure 2, to tractably find near optimal solutions [2, 3, 4]. In fact, the Sentestrat [2, 3] approach finds solutions within a known constant offset from optimal. In this approach, players calculate the amount of influence a given move can have on the final result and play the moves that remove volatility from the system first.

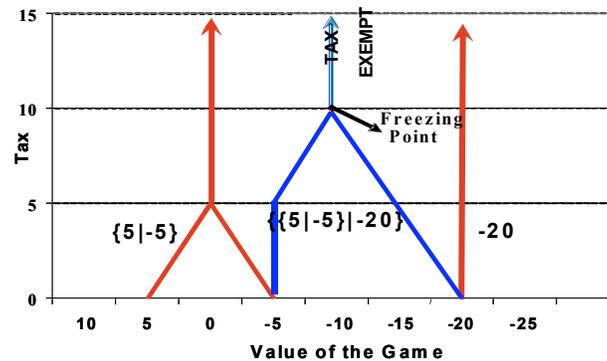


Fig. 2 Thermograph for $\{\{5 | -5\} || -20\}$

The following application domains could benefit from this approach:

1. *Local Area Networks (LANs)*: We assume there are no zombies on local machines, but zombies exist in the larger Internet that may target processes on the LAN. This approach identifies system bottlenecks and tells the administrator if the volume of the external traffic is enough to compromise distributed processes on the LAN.
2. *Corporate Networks*: When geographically separate offices (remote locations) are connected over the Internet using a Virtual Private Network (VPN), zombies can attack the VPN traffic that travels through the global Internet. By considering the graph structure of the VPN connections between corporate controlled autonomous systems, it is possible to create an adaptive VPN infrastructure that can tolerate DDoS attacks.
3. *Global routing problems*: Routing between autonomous systems (AS's) uses the Border Gateway Protocol, which is subject to instability in the presence of flooding DDoS attacks. Since some domains (*.edu, *.net, *.ru, ...) are more likely to host zombies than others (*.mil, *.gov, ...), we can analyze the AS graph structure to determine if the volume of traffic reaching sensitive BGP nodes is enough to disrupt the routing between critical agencies.

References:

- [1] R. K. Ahuja, T. L. Magnanti, J. B. Orlin, *Network Flows*, Prentice Hall, Upper Saddle River, NJ, 1993.
- [2] E. R. Berlekamp, J. H. Conway, and R. K. Guy, "Winning Ways for your mathematical plays Volume 1: Games in General," Academic Press, New York, 1982.
- [3] E. R. Berlekamp, "The Economist's View of Combinatorial Games," in: It Nowakowski (Ed.), *Games of No Chance*, MSRI Publications, Vol. 29, Cambridge University Press, Cambridge, 1996, pp. 365-405.
- [4] E. Berlekamp and D. Wolfe, *Mathematical Go or Chilling Gets the Last Point*, A K Peters, Ltd, Wellesley, MA, 1994.
- [5] R. R. Brooks, *Disruptive Security Technologies*, CRC Press, Boca Raton, FLA, 2005.
- [6] J. H. Conway, *On Numbers and Games*, AK Peters, LTD, 2000.
- [7] B. C. A. Milvang-Jensen, "Combinatorial Games, Theory and Applications," Thesis, IT University of Copenhagen, 2000.
- [8] C. Tondering, *Surreal Numbers – An Introduction*, <http://www.tondering.dk/claus/surreal.html> (last accessed 08/10/2006).
- [9] L. J. Yedwab, "On playing well in a sum of games," M.S. Thesis, MIT, 1985, MIT/LCS/TR-348.

Distributed Denial of Service Games

by
Chinar Dingankar, Student
Dr. R. R. Brooks, Associate Professor
Holcombe Department of Electrical and Computer Engineering
Clemson University
Clemson, SC 29634-0915
Tel. 864-656-0920
Fax. 864-656-1347
email: rrb@acm.org

05/2007

ORNL Presentation

Introduction

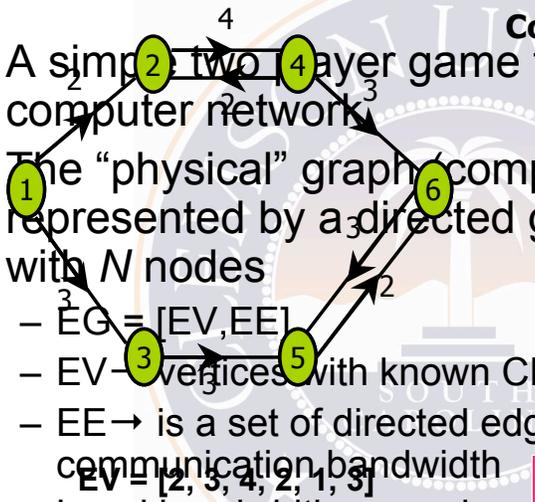
- Combinatorial game theory to analyze the dynamics of DDoS attacks on an enterprise
- A game is played on a capacitated graph (computer network)
 - Nodes have limited CPU capacities
 - Links/Edges have bandwidth constraints
- A distributed application runs on the computer network
- Our approach gives two important results –
 - It quantifies the resources an attacker needs to disable a distributed application
 - If the attacker does not have enough zombies required → provide near optimal strategies for reconfiguration of the distributed application in response to attempted DDoS attacks

05/2007

ORNL Presentation

Physical Environment

- A simple two-layer game for a computer network
- The “physical” graph component is represented by a directed graph with N nodes
 - $EG = [EV, EE]$
 - $EV \rightarrow$ vertices with known CPU load
 - $EE \rightarrow$ is a set of directed edges with known communication bandwidth
 - Local bandwidth on each node



Connectivity matrix (EE) =

0	2	3	0	0	0
0	0	0	4	0	0
0	0	0	0	3	0
0	2	0	0	0	3
0	0	0	0	0	2
0	0	0	0	3	0

Note (6,6) = 0

Players

- Two players in the game –
 - **Blue** – A set of distributed programs running on physically connected computers
 - $BG = [BV, BE]$
 - $BV \rightarrow$ is a set of nodes representing distributed programs with known CPU load
 - $BE \rightarrow$ is a set of edges or links representing the communications bandwidth needed between two programs
 - Local bandwidth on each node - Infinite
 - Represented by the color **BLUE**
 - **Red** – **Red** is an attacker that places zombie processes on physical graph nodes.
 - Zombies send network traffic over the physical edges
 - Number of zombies and where to place them
 - Represented by the color **RED**

Feasible Blue Configurations

- Set of feasible **Blue** configurations \rightarrow set of mappings of BV onto EV that satisfy two classes of constraints:
 - Nodal Capacity Constraint:

Nodal capacity of Physical node \geq Nodal Capacity of Blue
 - Edge Capacity Constraint:

Maxflow between two Physical nodes \geq Arc capacity of two Blue nodes

 - Two **Blue** nodes on same Physical node – Infinite Arc capacity
 - **Maxflow** for each pair of **source** and **sink** on the network
- Set of feasible configurations $\rightarrow BC = \{BC_1, BC_2 \dots BC_L\}$

05/2007

ORNL Presentation

RED disrupts Blue

- **Red** disrupts a **Blue** configuration by placing zombies so as to -
 - Attack node capacities - **Red** places zombies nodes hosting one or more **Blue** processes.
 - The node capacity attack is rather trivial and not very interesting
 - Difficult for Red to compromise the servers used by **Blue**
 - Flood arcs – **Red** places zombies on nodes that do not host **Blue** processes.
 - Zombies produce network traffic that consumes communications bandwidth on edges in EE
 - A **Blue** configuration is disabled \rightarrow required arc capacity of any Blue edge (s-t) becomes greater than the available maxflow from s-t on the physical graph
 - Our analysis focuses on flooding attacks

05/2007

ORNL Presentation

Zombie Traffic and Zombie Placement

- To determine the **set of zombies** needed by Red, we:
 - Calculate the **mincut** for each element of *BE*
 - Blue slack capacity at the **mincut** (BS)
 - Expected number of blue packets dropped
 - Volume of red traffic so that → no. of blue packets dropped > BS
 - Red traffic (RT),

$$RT = \frac{C}{\left[1 - \frac{BS}{\lambda}\right]} - \lambda$$

λ packets is the Blue traffic
C is the capacity of the physical arc

- Zombie Placement:** If the **Maxflow** to a node in the **mincut** of an element of *BE* is > **RT** then that node is a candidate for zombie placement.
 - We need minimum number of zombies → so look for zombie nodes that can disable more than one element of *BC*.
 - We get a smallest set of zombies needed to disable all elements of *BC*

05/2007

ORNL Presentation

Game

- If the attacker does not have enough zombies to disable all blue configurations → **Blue** has a chance to recover from the DDoS attack by reconfiguring.
- A simple board game.
- Rules for the game:
 - **Blue** starts the game.
 - Each player is allowed one move at a time.
 - **Blue** can take one possible configuration out of the available BC's for one move.
 - **Blue** cannot have redundancy i.e. multiple **Blue** copies.
 - Once **Red** places a zombie on a node it cannot move that zombie until its next turn
 - **Blue** reconfigures by migrating a single process from a physical node to another.
 - **Blue** and **Red** have perfect knowledge of each other's configurations.
- Aim of each player
 - **Red** tries to force **Blue** into a position where it cannot recover by transitioning to another element of *BC*.
 - **Blue** tries to find a "loopy" game where it can always return to a previous configuration.

05/2007

ORNL Presentation

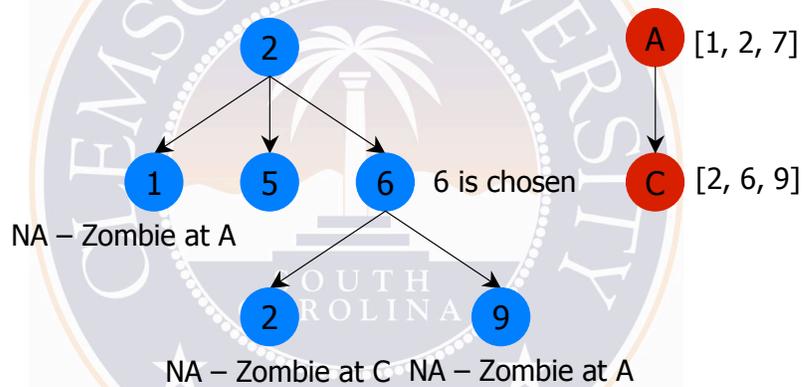
An Example Game

Blue Configuration	Reconfigure	Zombie Move	Zombies	Disrupt Blue Configurations
1	2, 3, 4	A	3, 5	1, 2, 7
2	1, 5, 6	B	1, 6	3, 4, 5
3	1, 7	C	3, 4	2, 6, 9
4	1, 10	D	1, 5	8, 9, 10
5	2, 8, 9			
6	2, 9			
7	3, 10			
8	5			
9	5, 6			
10	4, 7			

05/2007

ORNL Presentation

An Example Game

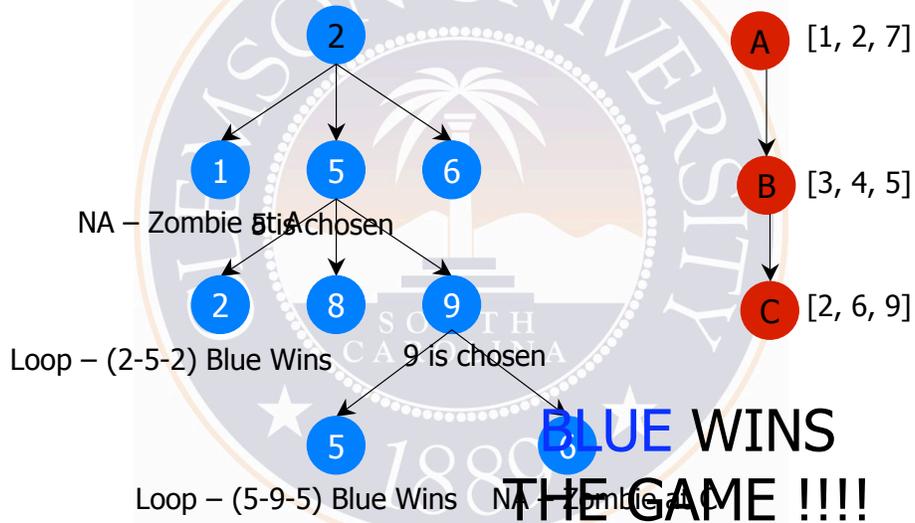


RED WINS THE GAME !!!!

05/2007

ORNL Presentation

An Example Game

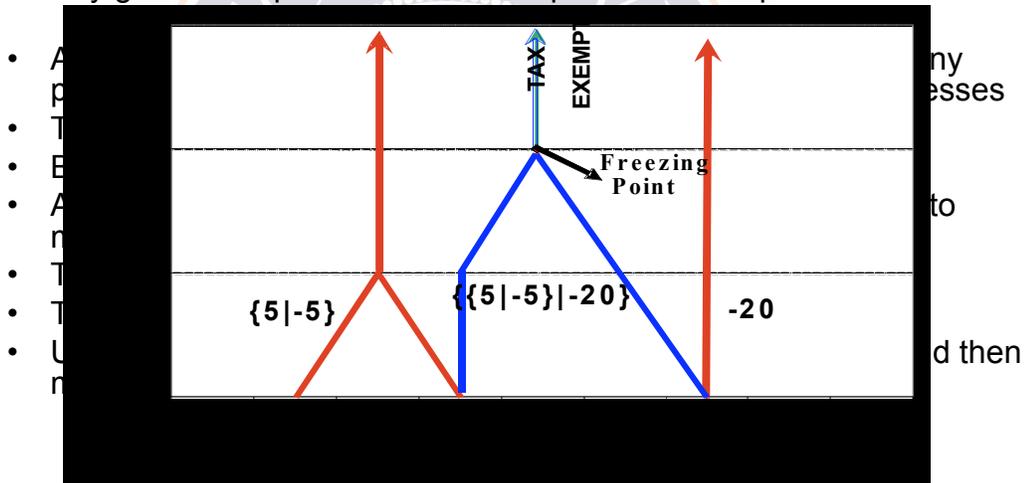


05/2007

ORNL Presentation

Thermographs

- Any given enterprise relies on multiple distributed processes



05/2007

ORNL Presentation

Applications

- **Local Area Networks (LANs):** Zombies in the larger Internet may target processes on the LAN
 - To identify system bottlenecks
 - To determine if volume of the external traffic can compromise distributed processes on the LAN.
- **Corporate Networks:** Zombies can attack the VPN traffic traveling through global Internet
 - Graph structure of the VPN connections can be used to create an adaptive VPN infrastructure that can tolerate DDoS attacks.
- **Global routing problems:** Routing between AS uses the BGP, which is subject to instability in the presence of flooding DDoS attacks.
 - AS graph structure can be used to determine if the volume of traffic reaching sensitive BGP nodes is enough to disrupt the routing between critical agencies.

Trusted Passages: Managing Distributed Trust Needs of Emerging Applications

Mustaque Ahamad, Martim Carbone, Greg Eisenhauer, Jiantao Kong, Wenke Lee, Bryan Payne, Karsten Schwan and Ramesh Viswanath
School of Computer Science
Georgia Institute of Technology
Atlanta, GA 30340

1 Motivation

Distributed systems and applications are becoming so complex that it is difficult for end users to understand or control (1) where their data will be accessed and stored, and (2) where their processing will be performed. This is because modern information processing infrastructures routinely cache data, intermediate results and parameters; they routinely integrate data, mine it, or operate on it on dynamically selected application servers; and they are now beginning to extend these host-level actions to also make use of underlying platform elements. At the same time, businesses must use the service architectures and infrastructures provided by industry, to control costs, to be able to interoperate with their partners, and more generally, to carry out the distributed IT processes that have now become routine. The following questions arise about the distributed systems and environments in which applications critical to an enterprise’s operational capabilities are run. First, to what extent can one trust the open service-based infrastructures companies must use to contain costs and to gain interoperability with external partners? Second, can open systems like these be used to construct distributed applications that deliver information critical to an enterprise’s ability to function, in a timely fashion and with trustworthy results? Third, is it possible to use the cost-effective shared Internet-based infrastructures for critical information processing and delivery in place of expensive enterprise-specific or point-to-point solutions.

Unfortunately, the answer to the questions posed above is that today’s security technologies are insufficient to provide this type of trust for large, distributed applications. In many applications, data is passed between databases, data processing applications, data format applications, and data serving applications. Moreover, request parameters and intermediate results are cached in various locations between clients, applications, and backends. Traditional security technologies, therefore, are unable to effectively monitor all of these interactions and make autonomous trust decisions for the user. For example, a VPN could secure the data traveling between the client and the server, but it cannot make guarantees about the processing that happens within the distributed system ‘behind’ the server. Should the client trust the data produced by the server? Traditional security solutions, which focus on secure storage and transmission of data, are not adequate when data is processed and stored at multiple points that change over time. New security solutions are needed to ensure that all applications that produce and process the data remain trustworthy.

2 Approach

Our approach relies on the notion of trusted passages which provide the framework necessary to address the problem of securing applications that run on large-scale distributed systems. Specifically, this abstraction dynamically manages trust for applications that execute on open distributed systems. From the client’s perspective, any single component in a distributed system can be trusted only if it satisfies certain properties. These properties can vary based on the client’s needs. Some examples include safety (e.g., correct execution of requested operations), proper handling of information, and acceptable response times. Likewise, the entire distributed application can only be trusted if each component that affects the application satisfies these properties. This last point is critical to understanding trusted passages. Information traveling through the components of a distributed application creates a virtual passageway. Therefore, the primary challenge is to ensure that each part of the passage is trusted.

Dynamic trust management starts by continuously measuring the trust level of each component in the distributed system. We define a platform as *trusted* if it processes data without any tampering of the data

or the processing platform. A trusted passage builds on this to also prevent manipulation of data in transit or the sending of corrupted data. While this definition, based on monitoring of components, is somewhat weaker than that of a system that is guaranteed to remain secure at all times, it is more practical for complex systems and is sufficient for the correct operation of important applications (i.e., to perform the steps originally programmed into the application without undetected modifications due to malicious attacks). A trusted passage accounts for all aspects of data processing, storage, and transport within the passage.

The trusted passage framework is composed of several components. A local *trust controller* monitors its host’s activities using virtual machine introspection and innovative intrusion detection techniques. Multiple trust controllers are connected to create a distributed system that can use the local information from each host to provide dynamic management of trust and to construct trusted passages that meet application needs. New system-level abstractions support efficient trust controller operation, imposing only small overheads on application and system execution.

2.1 Using Trusted Passages – An Example

Government and industry are increasingly relying on complex distributed systems to form their core computing infrastructure. For example, companies like Google, Amazon, and eBay use tens of thousands of computers to support Web service applications. The Federal Bureau of Investigation, along with various state agencies, maintains a distributed database for crime-related information called the National Crime Information Center. Delta Air Lines, one of our research partners, maintains a critical distributed system responsible for processing flight and passenger information. Each of these systems has unique requirements with regards to uptime, performance, storage, bandwidth, etc. However, they all utilize complex methods for processing and sharing data: (1) these demanding, distributed applications are considerably more complex than the traditional client-server model; and (2) data is passed throughout the system in complex paths, and it is stored, processed, and forwarded at many points between its source and destination.

To illustrate how trusted passages address the emerging security issues in these applications, consider the operational information system (OIS) used by Delta Air Lines. As shown in Figure 1, this massively distributed system combines transactional processing, with push-based event delivery and manipulation, with client-server actions at end points. Its purpose is to continually provide the company with up-to-date information about all of its flight operations, including data events about passenger boarding, flight arrivals and departures, flight positions, and baggage. Event generation, transport, processing, and output use a wide-area distributed network of end systems, servers, and networking equipment that connects them.

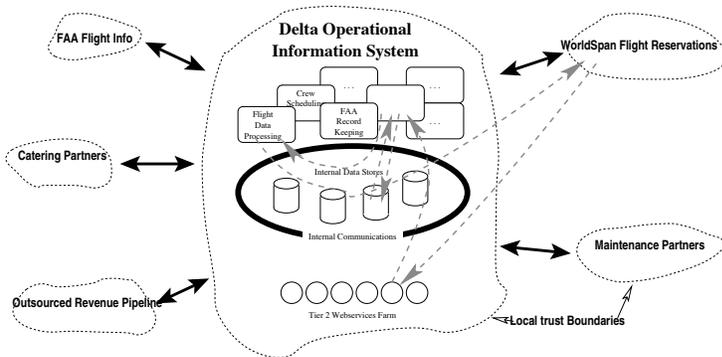


Figure 1: Delta Air Lines OIS.

Business logic applied to data creates meaningful information and generates the additional events used for tasks ranging from the update of airport terminal displays to notifications sent to caterers of passengers’ food preferences. This logic is run on multiple, high end server systems that continuously process input streams comprised of FAA flight position updates and Delta-specific flight information. These server systems form an Internal Event System (IES) that interacts with clients, both by generating continuously derived system-state updates and/or by responding to explicit external requests for information. The large number of

clients, the complexity of the business logic being applied and its working set size of hundreds of gigabytes, and a 24/7 uptime requirement dictate that business logic is implemented by multiple subsystems, some of which may be replicated across multiple nodes (and locations, for disaster recovery). Requirements on a

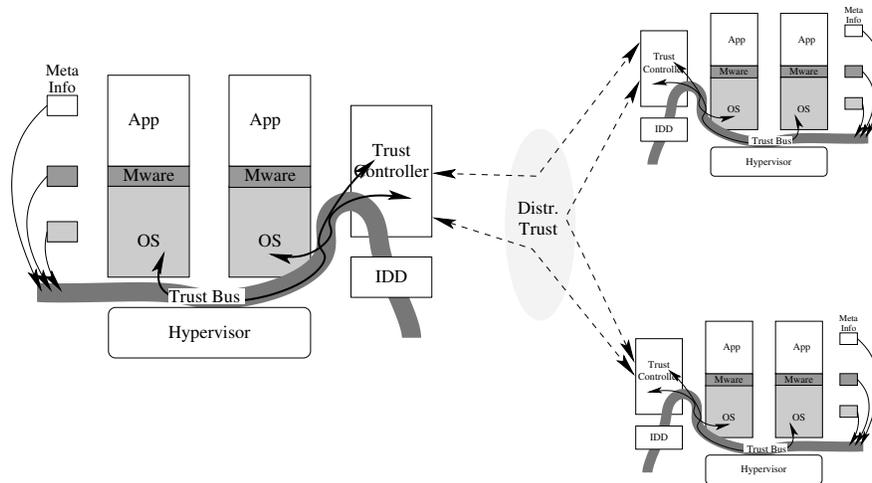


Figure 2: Trusted Passages Architecture.

critical system like this include high performance, high reliability and availability, and the ability to maintain constant service levels (as perceived by system clients), even under ‘unusual’ operating conditions.

The Delta OIS system exhibits many of the architectural features that make these distributed applications challenging to secure. *Trusted passages* are designed to provide security to an architecture with these key features:

- **Distributed Data Processing.** Data is generated, processed, and inspected at multiple locations between its origination and destination, across machines internal to a company and with a variety of external partners who are in different trust domains.
- **Distributed Data Storage.** Data is stored at multiple locations. This can be for redundancy, locality, or other architectural reasons.
- **Architectural Redundancy.** Critical architectural components are replicated in order to provide the reliability and uptime that these applications demand.

Together, these features describe the framework of a complex distributed system. Trusted passages go beyond existing security techniques to actively ensure trustworthy operation of these complex systems.

3 Architectural Overview

Figure 2 shows a trusted passage view based on different subsystems and machines jointly providing services to external clients. The presence of trust controllers running on all of the machines used by the application processes, with a vertical line between trust controllers and application processes indicating platform-enforced isolation between them, allows secure monitoring of the application processes. In addition to the the solid lines indicating application-level communications, dotted lines indicate communications between cooperating trust controllers. Not explicitly shown in the figure is how trust controllers detect problems and make decisions about what machines and software systems to trust. Here, we simply note that such decisions will be based on (1) *localized trust* – per platform monitoring of applications’ behavior, including their communication actions, and (2) *distributed trust* – information exchanged between multiple trust controllers. Trust decisions are made for each application, and the trust controllers to which an application has subscribed will endeavour to maintain some viable trusted passage that enables it to carry out its distributed processing tasks.

In our ongoing middleware research [2], we use a combination of active and passive standby nodes to attain reliable operation in the presence of nodes that can no longer be trusted. In the Agile Store project [4], we combine replication with secret sharing techniques to deal with dynamically detected attacks. In research conducted in peer-to-peer settings, we have used a combination of timeouts and result comparison to detect compromised subsystems, then react with runtime request re-routing and re-replication [5]. We envision a strategy where a trusted passage is able to create a new, uncompromised domain on a machine used by the application, and where the application uses its own methods for re-joining the computations and data exchanges being performed. Conversely, trusted passages will attempt to detect compromised subsystems and machines under attack, but application specific handling may be necessary to deal with detected problems.

3.1 Localized Trust

Trusted Passages demand that each participating platform actively monitors and manages its activities to ensure that certain trust properties are met. At a local level, this is handled by the trust controller. The trust controller has three primary responsibilities:

- *Monitor the local host*, collect this information, and make a local trust decision.
- *Cooperate with remote hosts* to support dynamic, distributed trust management.
- *Interact with local host applications*, to give applications access to trust information and therefore, the ability to deal with trust gain or loss.

In order to properly perform these tasks, it is critical that the trust controller be inherently trusted. This trust is provided using virtual machines to form a distinct boundary between the trust controller and the monitored platform. Leveraging existing work, the proposed architecture uses the Xen hypervisor [1] as a virtualization platform. The trust controller can execute in a privileged domain, and the monitored platform can execute in an unprivileged or user domain. This separation isolates the trust controller from traditional attacks. In addition to the isolation properties, each trust controller will operate in a protected environment complete with a hardened operating system and an intrusion detection system. Combining this with the isolation provided by the Xen hypervisor, the trust controller is able to operate at a significantly higher level of assurance than the monitored operating system. Finally, in order to ensure trustworthy communication between trust controllers, information must be securely transmitted. We plan to leverage existing work here and use techniques seen in virtual private networks (VPNs) such as encryption, authentication, and integrity checking using certificates.

Each trust controller will be responsible for monitoring any other domains running on the same hypervisor. The process of monitoring between virtual machines is known as virtual machine introspection (VMI). VMI allows one domain to monitor the current state of other domains including all physical memory, the CPU, device I/O, and any other data that passes between the hypervisor and domain. In order to facilitate interactions such as monitoring and response, we defined the XenAccess Library to provide the trust controller with a high-level view of each domain. This higher level abstraction will facilitate rapid development and exploration of new ways to leverage the powerful technique of virtual machine introspection.

Using the XenAccess Library, we are exploring innovative ways to monitor a domain. First, trust controllers will monitor program execution and compare the results with execution of the same request on other nodes. This technique is related to the behavior distance work by Gao et al. [3]. However, in contrast to this work, we will explore the use of input beyond system calls. The XenAccess Library will allow behavior distances to be computed using anything from the raw memory in a process image to user-level API calls. We will research different distance metrics to understand which input provides the most useful measurement for trusted passages.

In addition to the behavior distance work, we plan to use the XenAccess Library to provide input for anomaly detection of local program executions. As described above, the XenAccess Library provides an opportunity to experiment with new types of system information as well as new abstractions. Our work starts with a traditional anomaly detection approach, and then determines which new data sources (e.g.,

resource utilization by the application) to add into the training set. The end result will provide an additional tool that will detect deviation from a pre-defined normal behavior.

Combining these two techniques, the trust controller will have a powerful view into the state of the monitored domain. If the anomaly detector indicates a problem, the trust level of that domain and its host can immediately go down. However, if the domain passes the anomaly detection test, then the behavior distance approach will provide a more fine-grained view into its operations. These two approaches complement each other in such a way that it would be much more difficult for an attacker to avoid detection while still carrying out a malicious task.

3.1.1 Distributed Trust

We require that multiple trust controllers executing at different machines coordinate and compare their results. A compromised domain's observations are likely to differ from others and based on such comparisons, a trust value is associated with the domain. We plan to use models where trust values, that are meaningful at the application level, dynamically change in the range from 0 and 1 based on observed behavior of nodes. These values are used to represent the level of trust a controller associates with a domain and the platform where it runs. Higher trust values indicate a more trusted platform that meets the needs of a trusted passage and lower values indicate that the resources at the platform cannot be trusted to support the passage. A trust value degrades rapidly when the trust controller suspects that its observations indicate anomalous behavior or when they differ from observations of other controllers. We want the trust value of a platform that effectively supports a trusted passage to gradually increase and get close to 1 with time. Notice that we establish a distributed network of trusted hypervisors and controllers, and the trust value of a platform is used to determine if the platform can meet the safety and performance needs of the trusted passage. In this sense, our network of trust controllers acts as a distributed trust management infrastructure for a passage. Our research is exploring models for dynamically evolving trust values based on trust controller observations. We are also investigating how to effectively utilize dynamically computed trust values of multiple platforms to make resource management decisions that ensure that a trusted passage can be supported effectively.

4 Conclusions

The trusted passage project is addressing multiple challenges to effectively meet trust needs of applications. The primary thrust is on building trust controllers that are run on a distributed set of platforms to manage the resources that support a trusted passage. To achieve this goal, we monitor platform execution using virtual machine introspection and other performance metrics relevant to the trusted passage. The trust controllers at different nodes share their local information and coordinate their actions to ensure that the entire passage fulfills our definition of trust. Trust controllers enable us to provide the rich distributed processing and communication abstraction that we call a trusted passage. We plan to demonstrate the usefulness of the trusted passage abstraction and our approach for implementing them by experimenting with lab-scale versions of applications that deal with distributed information processing and dissemination.

References

- [1] Paul Barham, Boris Dragovic, Keir Fraser, Steven Hand, Tim Harris, Alex Ho, Rolf Neugebauer, Ian Pratt, and Andrew Warfield. Xen and the Art of Virtualization. In *Proc. of 18th Symposium of Operating Systems Principles (SOSP-18)*, Bolton Landing, NY, 2003.
- [2] Zongtang Cai, Vibhore Kumar, Brian F. Cooper, Greg Eisenhauer, Karsten Schwan, and Rob Strom. Utility-driven fault-tolerance in enterprise-scale information flows. Middleware, 2006.
- [3] Debin Gao, Michael K. Reiter, and Dawn Xiaodong Song. Behavioral distance for intrusion detection. In *RAID*, pages 63–81, 2005.

- [4] Lei Kong, Deepak J. Manohar, Arun Subbiah, Michael Sun, Mustaque Ahamad, and Douglas M. Blough. Agile store: Experience with quorum-based data replication techniques for adaptive byzantine fault tolerance. In *Proceedings of the International Symposium on Reliable Distributed Systems (SRDS)*, 2005.
- [5] Ramesh Viswanath, Mustaque Ahamad, and Karsten Schwan. Harnessing non-dedicated wide-area clusters for on-demand computing. In *Proceedings of IEEE International Conference on Cluster Computing (Cluster 2005)*, 2005.

“Trusted Passages”: Meeting Trust Needs of Distributed Applications

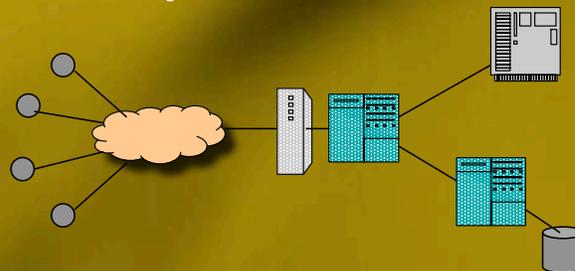
Mustaque Ahamad, Greg Eisenhauer, Jiantao Kong,
Wenke Lee, Bryan Payne and Karsten Schwan

Georgia Tech Information Security Center & Center
for Experimental Computer Systems
Georgia Institute of Technology



Funded by grants from NSF and Intel.

Application Characteristics & Needs Example: 3-tier Web Services



- Challenges:
 - Execution with distributed set of resources
 - Information creation, flow, transformation, caching, and access
 - End user services with well-defined properties
 - *Timely* information transport and processing; *responsiveness* despite external threats/attacks; *valid* outcomes and results; ...
 - Need for online management
 - Continuous monitoring and trust assessment
 - Runtime reorganization to maintain high levels of trust



Trusted Passages Approach

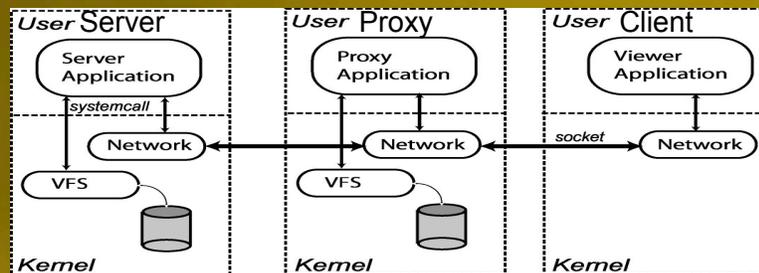
- Create and manage distributed information processing overlays
 - Example: sets of proxy servers `connecting` clients with servers
- Actively manage the overlays to provide online trust guarantees
 - Example: monitor proxy server behavior and adjust overlay accordingly

Approach: Example (refined): Content Caching by Proxies

Simple Trusted Passage:

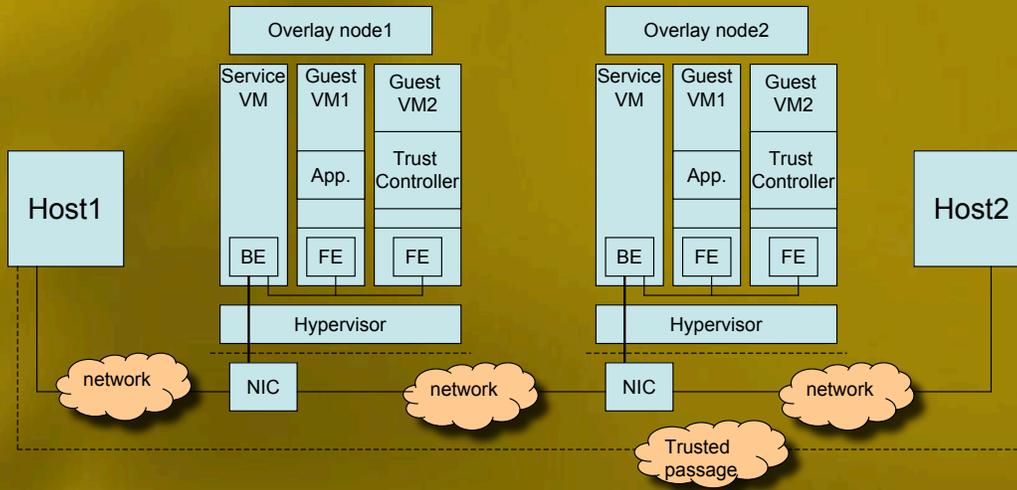
Data caching with web proxies: need for a trusted passage:

Client <--> Proxy <--> Server

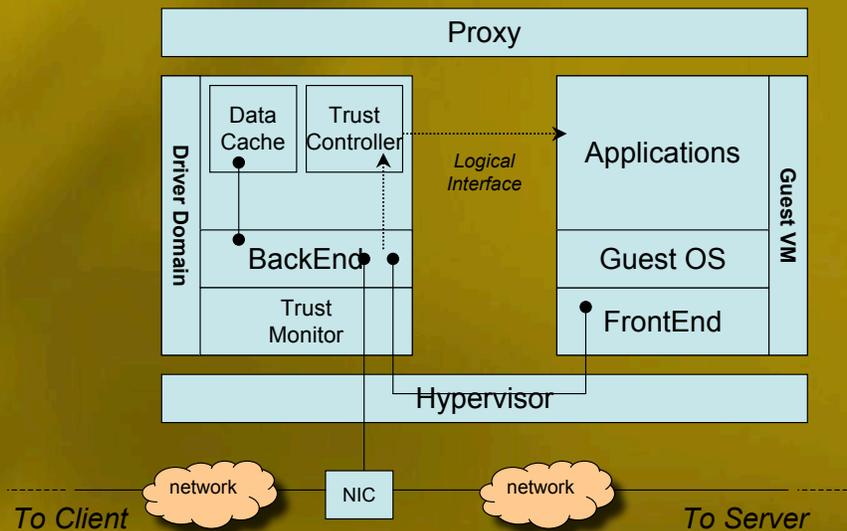


Need for Trusted Passage

Trusted Passages on Virtualized Platforms: General Concept



Trusted Passages on Virtualized Platforms: Example: Proxy



Trusted Passages - Summary of Concept

- Trusted computing base (hypervisor)
- Hardware support for isolation and safety (VT technologies)
- Sophisticated monitoring and detection models and tools:
 - Isolated trust controllers
 - Exploiting front end/back end device interactions to eliminate need to instrument Guest OSs

"Trusted passages" uses emerging technologies to provide new functionality to end users

Trust Modeling and Management

- Trusted node is one that meets application needs
 - Delivers desired performance levels
 - Properly processes and handles information
 - Probably not compromised
- `Better' trusted nodes should be selected to support a "Trusted Passage"
- Management example: use redundant processing and communication paths to attain higher overall levels of trust

Dynamic Trust Evolution

- Trust Controller (TC) monitors actions of a VM participating in a “Trusted Passage”
 - Chosen measurements of VM code and data
 - Logging of externally observed actions (e.g., virtualized device access via Service VM)
- Trust Controllers compare their measurements for replicated activity
- Incorrect results or incorrect operations degrade trust in node, whereas correct operations increase trust level
- Experiment with methods like trust incentives

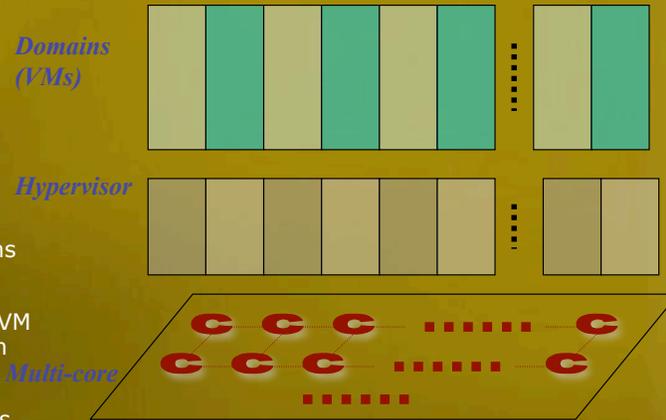
Platform-level Online Monitoring and Introspection

- Alternative techniques for monitoring guest OS activity
 - Intercept system activity (e.g., devices vs. using hardware like performance counters)
 - Dynamic integrity checking (e.g., use OS knowledge to capture and compare key structures jump tables)
 - Other methods (e.g., middleware instrumentation)
- Evaluate performance impact of monitoring
 - Assistance from platform monitoring services?
- Experiment with trust violations

Future Platforms and Services for Trusted Passages



- **Multi-> Many-Core:**
 - **Trusted Passages: Using VMs:**
 - **Specialized VMs:** performance impact of using trust controllers
 - **Management VMs:** trust controller actions on single machines and interactions across multiple machines
 - **Monitoring VMs:** costs of dynamic trust assessment, introspection, ...
- **Future Platform Services:**
 - **For Trusted Passages:**
 - **TrustBus:** system mechanisms for hardware support for efficient monitoring and management, for on-chip VM-VM interactions, for cross-platform interactions
 - **Adaptive Scheduling:** for guest VMs vs. Trust Controllers (TC), in response to threats
 - **Isolation:** Isolating TC-TC from VM-VM interactions for improved survivability



Summary



- Trusted Passages: new functionality relevant to large class of applications
 - Information stream processing (multimedia, event based systems – e.g., business activity monitoring, caching services with proxies, ...)
- Exploits new technologies (virtualization, multi-core, hardware performance counters)
- Research Contributions
 - Useful trust models and dynamic trust evolution
 - Platform level monitoring and introspection techniques
 - Provides insights for potential new services for multi-core platforms



Network Pathogen Spread Model Using Random Graphs

Christopher Griffin and Richard R. Brooks

March 23, 2007

Abstract

This talk considers the spread of *worms* in computer networks using insights from epidemiology and random graph theory. We provide three new results. The first result refines previous work showing that epidemics occur in scale-free graphs more easily because of their structure. We argue, using recent results from random graph theory that for scaling factors between 0 and ~ 3.4875 , any computer worm infection of a scale-free network will become an epidemic. Our second result uses this insight to provide a mathematical explanation for the empirical results of Chen and Carley (L. Chen and K. Carley, The Impact of Countermeasure Propagation on the Prevalence of Computer Viruses, IEEE Trans. Sys. Man and Cybernetics, 34(2), 823-834), who demonstrate that the Countermeasure Competing strategy can be more effective for immunizing networks to viruses or worms than traditional approaches. Our third result uses random graph theory to contradict the current supposition that, for very large networks, monocultures are necessarily more susceptible than diverse networks to worm infections.

To facilitate understanding, we will introduce the concepts of "Random Graphs" computer network worm, scale-free graphs. We will also provide essential results from epidemiology when necessary. The results we present were published in (C. Griffin and R. R. Brooks, A Note on the Spread of Worms in Scale-Free Networks, IEEE Trans. Sys., Man and Cybernetics, 36(1):198-202, Feb. 2006).

Worm Spread in Scale-Free Networks

A Model Using Random Graph Theory

PRESENTED TO:

CSIIR Workshop
Oak Ridge National Lab

PRESENTED BY*:

Christopher Griffin
Penn State Applied Research

*Richard R. Brooks of Clemson University contributed to this study.

1

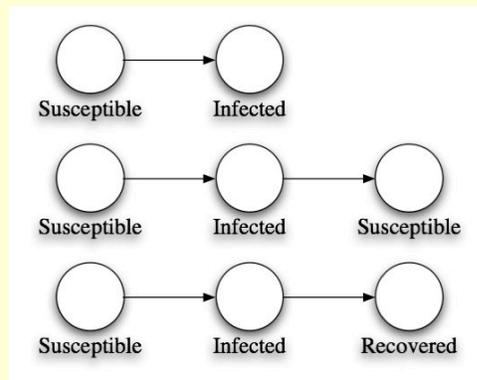
Goals of Presentation

- Summarize the epidemiological models of worm spread in the Internet
- Introduce *Random Graphs* as models of the Internet
- Propose a natural model of worm spread using *Random Graphs*
- Demonstrate quantitative results showing this model may be appropriate

2

- **Computing a self-replicating program able to propagate itself across a network, typically having a detrimental effect.**
- The name 'worm' comes from The Shockwave Rider, a science fiction novel published in 1975 by John Brunner.
- Researchers John F Shoch and John A Hupp of Xerox PARC chose the name in a paper published in 1982; The Worm Programs, Comm ACM, 25(3):172-180, 1982), and it has since been widely adopted.

- **Epidemiology:** The study of the spread of disease in populations.
- Diseases may spread quickly and then die out (Ebola) or remain endemic within a population (Chicken Pox)
- Populations can be modeled in a number of ways:
- “SI”, “SIS” or “SIR” models are most common.





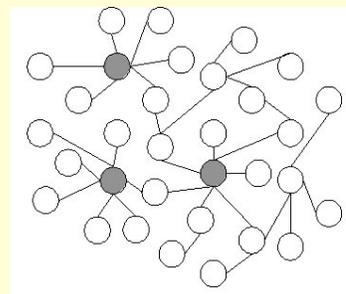
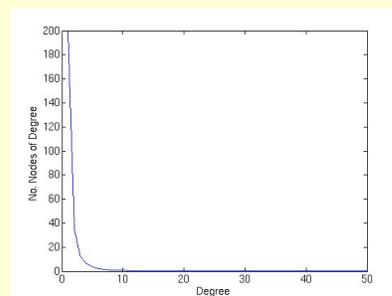
Mathematical Epidemiology

- Classical Mathematical Epidemiology Uses 3 Key Parameters in SIS/SIR Models:
 - R_0 : The number of secondary infections that occur when **one** infective agent is introduced into a population.
 - $\sigma(t)$: The average number of effective contacts an individual has during his/her infected period.
 - $R(t)$: The average number of secondary infections produced by an individual during his/her infected period.
- In general, epidemic is only possible if $R_0 > 1$.



Scale-Free Networks

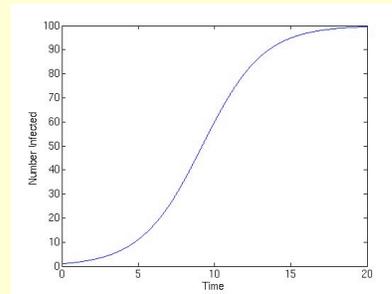
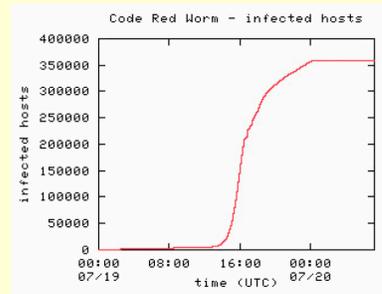
- A graph $G=(V,E)$ is scale free if the number of vertices with degree d follows an inverse power law. That is:
- $n(d) = k/d^\alpha$
 - $n(d)$ is the number of vertices with degree d .
 - k is a constant of proportionality, and
 - α is the scaling parameter.
- Scale-free graphs have gained popularity in recent years.
- Examples: The World Wide Web, Human Sexual Contacts, Protein-Protein Interaction Networks.





Worm Models with Epidemiology

- R. Pastor-Satorras and A. Vespigani studied the spread of worms in Internet-like networks using classical mathematical epidemiology.
 - Differential Equation Model of Infection Spread
 - Mean-Field Theory Approximations
- They show that for certain scale-free networks with scaling parameter < 3 , epidemics will occur for all diseases with $R_0 > 1$.



7



OK, this model “looks” good. Why not use it?

- Three reasons to search for a different model:
 - These models assume a completely mixed population.
 - Classical mathematical epidemiology assumes a fluid-like behavior of individuals.
 - R. Pastor-Satorras and A. Vespigani were studying general scale-free networks, not computer networks specifically.
- Two dangers to note:
 - In the absence of *ad hoc* mesh networks, computers do not mix.
 - The effective R_0 is highly dependent on the initial infection position.

8



Graphs and Random Graphs

- A graph $G=(V,E)$ is said to have a *giant component* H if H is a subgraph and contains a majority of the vertices of G .
- A random graph is a misnomer. A random graph is a tuple (Γ,p) , where Γ is a set of graphs and p is an appropriately defined probability measure on a sigma algebra of Γ .
- The *most widely studied* random graph family is $\Gamma(n,p)$, where each graph in Γ has n vertices when any graph G is chosen from Γ the probability that there is an edge between two arbitrarily chosen vertices is p .
- These are the *Erdős-Renyi* Random Graphs.

9



Random Graph Model of SF Graphs

- Aiello et al. have formulated a random graph model of SF graphs.
- Let $\Gamma(\alpha,\gamma)$ be the collection of graphs whose degree distribution follows the curve $n(d)=\lfloor \exp(\alpha)/d^\gamma \rfloor$.
 - Here $\lfloor x \rfloor$ denotes the greatest integer lower bound for x .
 - Aiello et al. have shown that this definition is mathematically sufficient and that a reasonable probability measure can be defined.
- In this model, α (roughly) controls the size of the graph while γ controls the scaling of the graph.

10



Relation to Epidemic Models

- **Lemma [Griffin & Brooks 2006]:** If G is an element of $\Gamma(\alpha, \gamma)$, and vertices of G are uniformly randomly *kept* with probability $0 < p \leq 1$ to produce G' , then a.s. G' has the same properties as G .
- **Theorem [Griffin & Brooks 2006]:** For any infection in graph $G \in \Gamma(\alpha, \gamma)$ with $\gamma > 2$, and with nodes having susceptibility probability p , then for all time

$$R_0 = \sigma(t) = p \frac{\xi(\gamma - 1)}{\xi(\gamma)}$$

11



Infection Potential

- **Theorem [Griffin & Brooks 2006]:** If $2 < \gamma < \gamma_0$, and for any infectious agent with infection probability p , a.s. $\lim_{t \rightarrow \infty} i(t) = p$. Where $i(t)$ is the proportion of infected nodes.
- This result is particularly interesting:
 - Often the affects of Internet worms have been blamed on the monoculture of Microsoft products.
 - This theorem suggests that even in the absence of a network monoculture, for appropriate Internet structures, 100% infection would occur among the susceptible nodes.

12

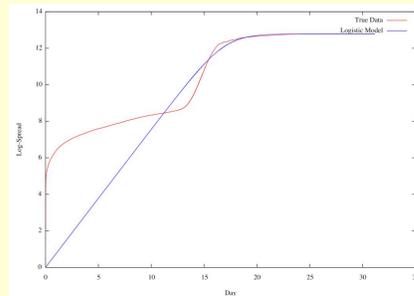
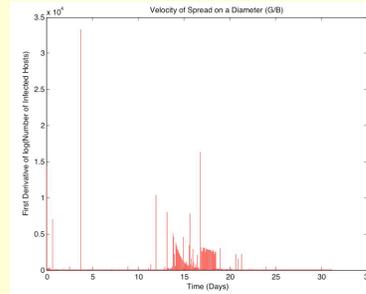


Rate of Infection

- **Theorem [Griffin & Brooks]:** Suppose that the rate of infection is constant, then the time required to achieve total infection is a.s. $O(\log|G|)$.
- Suppose that the infection rate is $r(t)$, then:

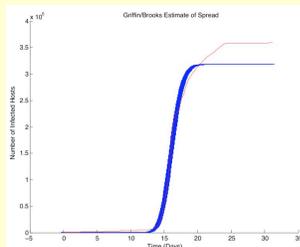
$$I(t) = \exp\left(\int_0^t r(t) dt\right)$$

- For certain $r(t)$ we can obtain an “S” curve matching the data.
- This gives a *natural* model of infection rate that matches the given data and does not appeal to continuous mixing models.

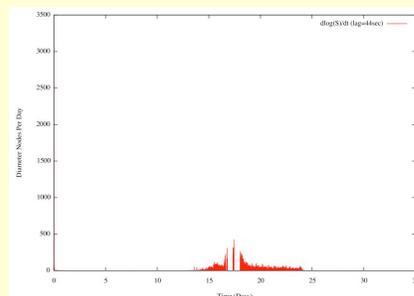
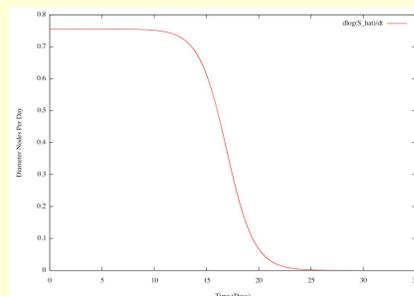


Comparison of Approaches

- When we try a model: $r(t) = \beta_1 \exp(\beta_2(t - \mu)^2)$
- We obtain:



- The model is seen to be imperfect because the true “logarithmic rate” does have hump, but it is probably not Gaussian in nature.
- G/B puts the diameter of this monitored network at ~13--this is a bit smaller than most estimates of the diameter of the Internet.





Infection Countermeasures

- **Theorem [Griffin & Brooks 2006]:** Centralized patch distribution runs in $O(|G|)$, while decentralized “white worms” can inoculate machines in time $O(\log|G|)$ assuming a constant rate of transmission.
- This theorem was “experimentally verified” by Chen and Carley (2005).
- What does this mean?
 - Centralized patch distribution is inefficient but...
 - Centralized patch distribution is safe.
 - Inefficiency is the cost of safety.
- Here is a real tradeoff: either we distribute patches quickly and prevent global infection at the risk of creating patch-based errors or we live with our current security model.

15



Conclusions

- Infectious agents in computer networks can be modeled using natural “random graph” models.
- These models are more appropriate than continuous mixing models.
- For scale-free random graph models, total infection is a.s. whenever $\gamma < \gamma_0$, hence infections are a function of network structure as much as pathogen.
- Infection rates can be well described using the random graph model.
- There is a natural trade-off between security countermeasures efficiency and safety. This confirms experimental results presented by Chen and Carley.

16

Robustness and Adaptation in "Information Ecosystems"

Stephen Racunas, Stanford University

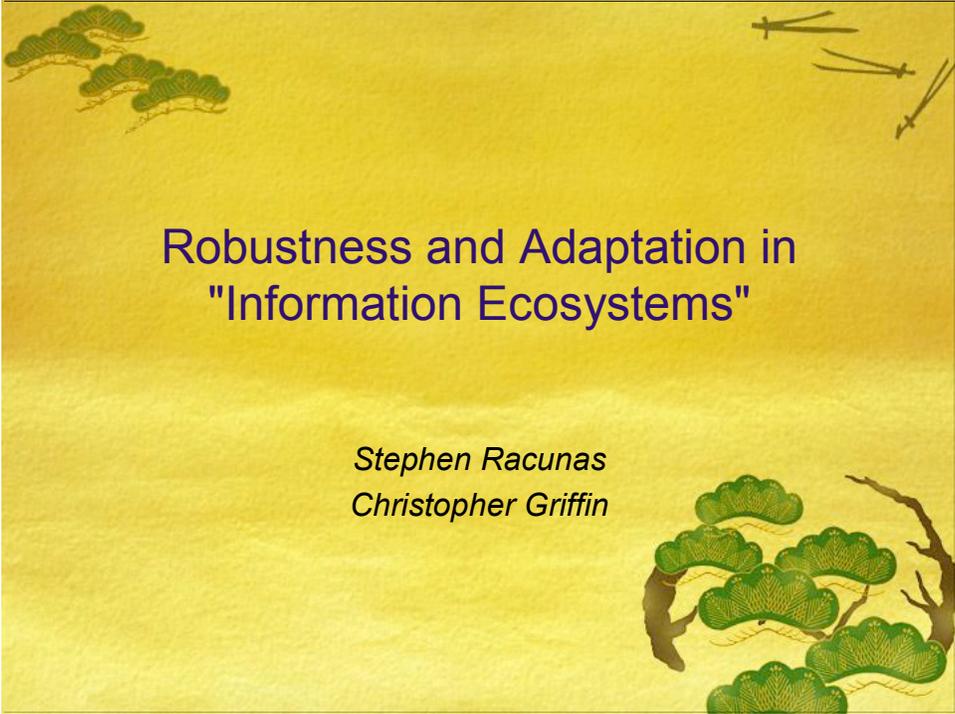
Abstract

We have been developing techniques for monitoring and "proofreading" knowledge resources, and these techniques proved to scale remarkably well. We are easily able to dynamically proofread the knowledge base containing all known and curated human metabolic pathways at speeds that enable real-time human interaction with both the data and with continuous evaluations of that data's internal consistency and reliability.

Recently, both manual and automated techniques for the attachment of machine-understandable meta-data to incoming knowledge streams have proved to be effective ways of verifying and cross-checking information validity at multiple levels of resolution during the evaluation of medical and biological data. We illustrate the power of such techniques by detailing our construction of model verification and hypothesis evaluation software for *S. cerevisiae*.

We outline how one might combine these two technologies to achieve a context-aware system capable of recognizing and mitigating both accidental and malicious information loss and conflict. By attaching meta-data to broad classes of information records to indicate what an adversary would be able to do with such information if it were compromised and what an adversary might do to such information if it were in fact misrepresented, we can use our techniques to automatically formulate and evaluate hypotheses about historical and potential threats.

As attacks become more commonplace and information infrastructure becomes more complex, we believe it will make less and less sense to try to eradicate, forstall or backtrace each individual attack. Organizations produce information which is of value to others. Various allied entities must consume this information in a timely manner for both participants to function optimally, and various antagonistic entities must be prevented from interfering. Rather than considering only individual information transactions, we believe it makes sense to apply biologically-inspired techniques for evaluating and maximizing the fitness of information strategies on the level of entities competing for resources – the level of the "information ecosystem."



Robustness and Adaptation in "Information Ecosystems"

Stephen Racunas
Christopher Griffin



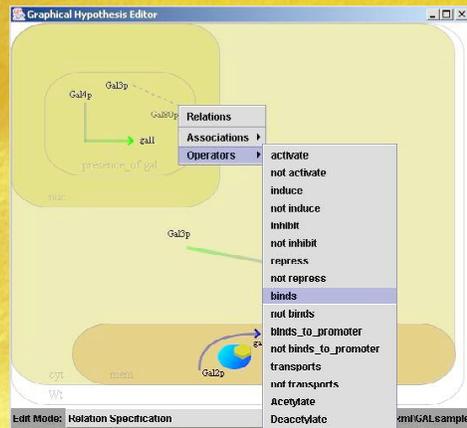
"Cross-Disciplinary Cross-Pollination"

Bioinformatics work Information Security
Managing hypotheses, evaluating scenarios
Inference based on observations *in context*
Integration of heterogeneous data to support inference

Information Security Bioinformatics
Identify, prove, discourage IP violations
Confidentiality vs. availability tradeoffs
Secure management for large and oft-used data sets

Common Problems and Overlapping Interests
Inference without compromise
Provenance and tracking
Medical records || compartmentalized data

Compose and Evaluate Hypotheses



Intuitive interface for composing and editing hypotheses

Library of events

Formal semantics specify hypothesized events *in context*

Identify errors in hypotheses

Multiple data types, sources

Suggest refinements

```
hy1 = (ev0+ev1) and (ev2+ev3)
ev0 = Gal2p transports galactose in mem in wt
ev1 = galactose activate Gal3p in cyt in wt
ev2 = Gal3p Binds_to_promoter gal1 in nuc in wt
ev3 = Gal3p induce gal1 in nuc in wt in presence_of galactose
Total supp:9, Total contr:1
```

Directly Address End-User Concerns

People worry about specific security threat scenarios

These concerns can be used to form hypotheses about historical, ongoing, or potential attacks

Hypotheses relate entities and intent

Do data from observed breaches support the hypothesis?

Do current observations disprove the hypothesis outright?

If there is little support for this particular hypothesis, what entity+intent combinations *are more highly* supported?

How are these highly supported combinations related?

Considerations of Intent

Actions always happen **in a context**

Attacks are caused by a certain set of **root intents**

Perhaps an intruder intends to probe the security of the US DoD infrastructure in order to garner sensitive information

Perhaps an intruder intends to cause economic damage to a specific target within a network or to the network itself

Identification of intent

Help protect from future attacks

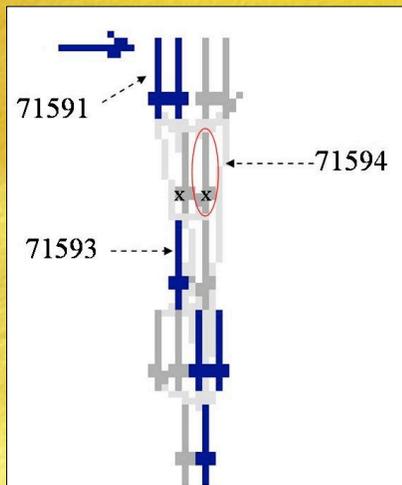
Dramatically reduce false alarms

Mitigate the effects of an attack in progress

Help responders identify intent by identifying streams of events directed to a common goal

Approach the problem with a method firmly rooted in the **formal scientific method** of hypothesize-and-test

“Proofreading” Knowledge Sources



Identify “holes” in knowledge

Proofread for:

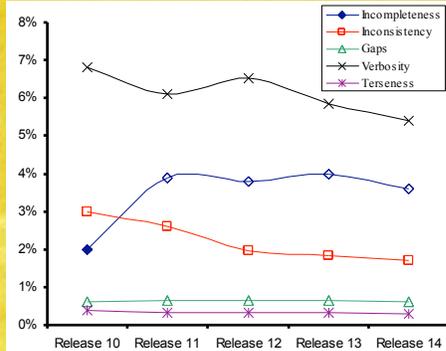
- Inconsistencies
- Incompleteness
- Gaps
- Well-formedness
- Misleading Cycles

Suggest “patches”

- Internal consistency
- Cross-validation with respect to similar resources

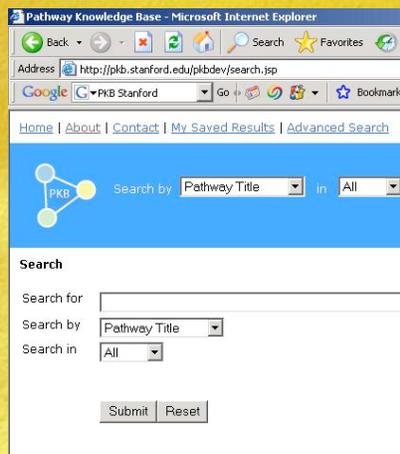
Sample KB Proofreading

Property	R- 10	R- 11	R- 12	R- 13	R- 14
Incompleteness	2 %	3.90 %	3.80 %	4.00 %	3.60 %
Inconsistency	3 %	2.60 %	1.95 %	1.84 %	1.70 %
Gaps	0.60 %	0.64 %	0.65 %	0.65 %	0.60 %
Verbosity	6.80 %	6.10 %	6.52 %	5.85 %	5.40 %
Terseness	0.40 %	0.32 %	0.32 %	0.32 %	0.30 %
90% Well formed	40 %	42 %	45 %	44 %	47 %
Self-loops	0 %	0 %	0 %	0 %	0 %



Techniques and scripts used by KB staff for proofreading their future releases

Knowledge Base Unification

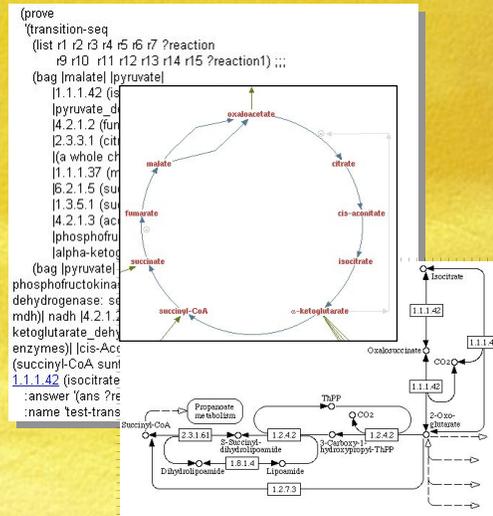


Human (981 pathways)
[Reactome](#) (688 pathways)
[BioCyc](#) (185 pathways)
[KEGG](#) (108 pathways)
 E coli (494 pathways)
[Reactome](#) (184 pathways)
[BioCyc](#) (235 pathways)
[KEGG](#) (75 pathways)
 Yeast (548 pathways)
[Reactome](#) (313 pathways)
[BioCyc](#) (173 pathways)
[KEGG](#) (62 pathways)

Hypothesis-Based Querying

Given at least a “semi-formal” phrasing...

1. **Select** the best-match resource
2. **Highlight** conflicts
3. **Present** the “best completion” from the union of all other resources



Meta-Data and Testing for Intent

What could someone actually *do* if they *could* gain access to a given information resource?

Game theory
Meta-data

Hypothesis space consisting of motives, potential future actions, and the contexts in which actions take place

Analysis of hypothesis space

Past precedents
Online observations
Prune the potential outcome structure in real time
Provide a “least likely to reject” solution set

Biologists and Information Security

Biologists worry about information security too...

Unauthorized use of experimental results by others

Data (very bad)

Paradigm-shifting insights (much worse)

"Getting scooped" can ruin a career or a lab

All this without even broaching the issue of bio-medical data!

How to be helpful without being overly vulnerable?

Help others to avoid following false leads

But don't allow others to steal your unpublished good ideas

Track what results contradict which hypotheses and how

But respect sensitive aspects of methodologies or raw data

Community Tools

The screenshot shows a web browser window titled "HyQue: index - Microsoft Internet Explorer". The address bar shows "http://localhost:3005/hy_que". The page content includes a diagram of a cell with a nucleus and cytoplasm, and a table of recorded contradictions.

Hypothesis 1:

Gal2p transports galactose through the cell membrane in wildtype *Saccharomyces cerevisiae*
and then
galactose activates Gal3p in the cytosol in wildtype *Saccharomyces cerevisiae*
and then
(Gal3p binds to the promoter of gal1 in the nucleus in wildtype *Saccharomyces cerevisiae* in the presence of galactose
and
Gal3p induces gal1 in the nucleus in wildtype *Saccharomyces cerevisiae* in the presence of galactose)

List of Recorded Contradictions:

1	Source	Found	Description	Suggestion	Pubmed	(actions)
	HyBrow (automated evaluation)	Thu Jun 01 20:17:00 Pacific Standard Time 2006	Gal3p is annotated to be a transcriptional repressor. Hypothesis as stated seems to assume Gal3p is a transcriptional activator	Gal 4p is a transcriptional activator		Show Refute Comment
2	Source	Found	Description	Suggestion	Pubmed	(actions)
	Stephen Racunas (manual literature search)	Sat Jun 24 21:38:00 Pacific Standard Time 2006	Transcriptional regulation in the yeast GAL gene family: a complex genetic network. D. LOHR, P. VENKOV AND J. ZLATANOVA	Gal3p releases Gal8p inhibition of Gal4	7601342	Show Refute Comment

Add an additional contradiction from one of your bookmarks

“Information Ecosystem”

Organizations and individuals produce, consume, and compete for information

Most require outside information to function

Primary objective is not to restrict information flow

Objective is to *make an entity or organization **maximally competitive** in its information environment*

Such optimization is possible

Rich history in biological domain

We seek new techniques for combining logical and numerical optimization in a contradiction-based way

The End

Thank you!

Deploying Statistical Anomaly Detection to Improve Cyber Security: Strategy, Benefits, and Results

Dr. Greg Shannon, Chief Scientist, CounterStorm, Inc.

There will always be new vulnerabilities in the information networks that control our nation's critical infrastructure. Problems arise both in legacy systems and in any new device or software application that we deploy. This is a persistent problem because our adversaries work constantly to find new and inventive means to uncover our vulnerabilities and attack our information and control systems. Our current approach to the problem with signature-, rule- and policy- based intrusion detection is self-limiting; it only protects us from what has happened in the past. In contrast, anomaly detection (AD) tools allow us to recognize new conditions on the network that may be indications of previously unknown attack mechanisms, i.e. zero-day attacks. In this presentation, we describe the advantages and challenges of AD for protecting our critical information systems.

Some vendors advocate a broad-based approach to the problem; a one-system fits all solution. One such approach is exemplified by NBAD systems. These systems use anomaly detection but they use it to provide a system administrator with an overview of network performance statistics with the goal of maintaining the health of the network. Our view is much more focused; we are using anomaly detection strictly for security purposes. The Netflow data used by NBAD systems does not provide sufficient granularity for useful analysis of network security-related events. We use normal traffic on a healthy network as a baseline from which to learn when abnormal traffic flow indicates an impending attack.

We have deployed our anomaly detection software successfully to recognize targeted attacks, botnets, and worms. Our experience from use in the field makes us confident that our sensors detected threats for which there are as yet no known signatures; in fact signatures for these attacks were not developed until 4 to 48 hours later.

In this presentation, I first discuss the truly disruptive nature of anomaly detection technology. I'll describe the state of the art in AD and where I believe the technology will most benefit from a new research focus. Deployment considerations for AD sensors will be discussed along with the advantages and disadvantages inherent in each example. One commercially available AD system will be described in detail. Results from field deployment of CounterStorm's Active Threat Recognition Suite will be described along with results of a joint deployment with an Autonomous System Traceback program developed by Southwest Research Institute. The advantages of coupling these two systems - one that can recognize an impending or on-going network attack and one that determines attribution for that attack - will be discussed.

The challenges that need to be addressed with AD tools are similar to any network detection system: Once we have determined that network behavior is significantly different from normal, what should be done? What information does the end-user need and how best do we convey this information? How much of the mitigation of the threat should be done automatically and what oversight role should the human play in this mitigation? Many of these issues depend on the deployment environment. Will there be a CERT team in-place to address the problem? How do we reduce the number of false positives in the system? For a particular environment (such as the Insider Threat for the Intelligence Community), would it be better to not report than to over-report? Who sets these policies and how do we help to determine how they should be set?

Like any truly disruptive technology, adaptation through operational use is the key that will allow AD sensors to radically change the way we currently do business. The future is not in vulnerability patches and fast signature distribution; the future is in recognizing that the system is malfunctioning before the damage is done.

Dr. Greg Shannon: Dr. Greg Shannon is CounterStorm's primary representative to U.S. government agencies and system integrators. He is also the Company's Principal Investigator for its two DHS SBIR Phase II awards. Dr. Shannon's research, development and management experience spans two decades with industry, academic and government working on security, networks and data analysis. Prior to joining CounterStorm, Dr. Shannon worked at Lucent Technologies, Ascend Communications, his own startup, Los Alamos National Lab, and Indiana University on such projects as FCC-recommended cyber security best practices, normalization of encryption export controls for network security equipment, and building scalable network-security appliances. Dr. Shannon received his B.S. from Iowa State University in computer science with minors in mathematics, statistics and economics, and earned his Ph.D. in Computer Sciences from Purdue University.

Company Overview: CounterStorm is a leading provider of modular threat detection and mitigation software development kits (SDKs) to security and infrastructure companies, as well as sophisticated government and commercial end users. Headquartered in New York City, the company was formed in August 2001 to commercialize patent-pending technologies developed at Columbia University under grants from the Defense Advanced Research Projects Agency (DARPA). CounterStorm is venture funded, with Novak Biddle Venture Partners, JK&B Capital, and Paladin Capital Group as lead investors. The company has also been awarded Small Business Innovative Research (SBIR) grants by the Homeland Security Advanced Research Projects Agency (HSARPA) of the Department of Homeland Security's Science and Technology Directorate.

www.counterstorm.com

Deploying Statistical Anomaly Detection to Improve Cyber Security

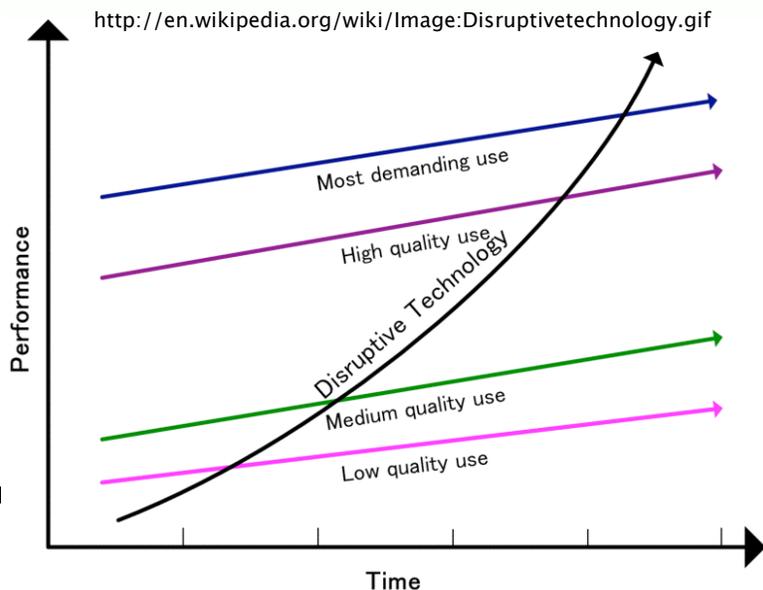
Dr. Greg Shannon
Chief Scientist

shannon@counterstorm.com

May 14, 2007

Comments on Disruptive Technologies

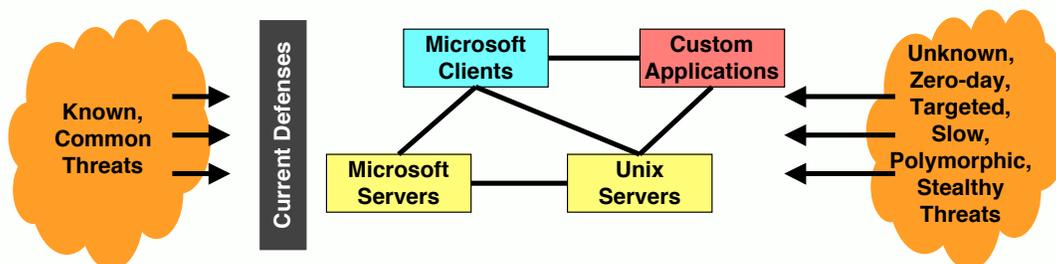
- The DT usually has new / unique capabilities
- The DT often is invented by or at least known by the incumbents
- The DT always has an initial niche area of REAL application



Serious Cyber Threats

How do we protect mission-critical networks and applications against the growing volume of sophisticated cyber attacks?

- Attacks are designed to avoid detection
- Protection is needed BEFORE attacks start



Copyright 2007, CounterStorm, Inc.

May, 2007

3

Thesis

- **Statistical Anomaly Detection is a disruptive technology to the incumbents of signatures, policies and rules for detecting the most serious threats to information systems and networks**
- **Statistical Anomaly Detection now has a niche in improving federal cyber security where incumbent solutions are significantly inadequate**
- **Statistical Anomaly Detection needs R&D investments to improve the demonstratable scope of threats detected, understanding alerts and threat scalability**

Copyright 2007, CounterStorm, Inc.

May, 2007

4

CounterStorm Background

- **Protecting against politically and criminally motivated attacks**
 - Handle the next generation of threats
 - CounterStorm-1 appliance for internal threat protection
 - Software subsystems
- **Founded in 2001**
 - Headquartered in NYC
 - Started with research at Columbia funded by DARPA since the mid-90s
- **Enterprise proven, government supported**
 - Enterprise customers in healthcare, media and finance
 - Two Department of Homeland Security SBIR phase II grants
 - Used in DoD programs



Copyright 2007, CounterStorm, Inc.



May, 2007

5

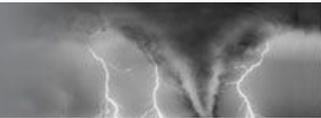
What is Statistical Anomaly Detection?

- **Anomalies reflect important observable behavior**
 - Bad \Rightarrow Anomaly ... Finding Loose Threads
- **Statistical Anomaly Detection:**
 - A machine-learning technique for detecting when observed behavior statistically deviates from a base-line behavior profile
 - E.g., The probability of observing this value/event is .001%
 - A packet of all K's on port 80
 - 100 SMTP connections in 10 seconds
 - Anomaly Detection leverages the inherent complexity of networked systems to detect the growing variety of threats, especially evasive zero-day threats
- **It is NOT:**
 - Detecting protocol violations (protocol anomaly detection)
 - Setting value thresholds (network behavior analysis, NBAD)
 - Drawing graphs for operators to "see" unusual behavior

Copyright 2007, CounterStorm, Inc.

May, 2007

6

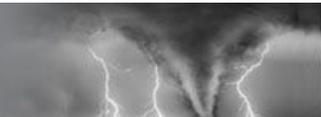


Problem Space – Detecting Cyber Threats

- **Problem**
 - Detect cyber threats in a timely manner

- **Status Quo**
 - Signatures, rules and policies
 - Based on past or anticipated threats

- **Key Detection Properties**
 - Coverage of known threats
 - Installability
 - Alert understandability
 - Coverage of evasive and new threats
 - Scalability to threat volume, velocity and variety
 - Accuracy

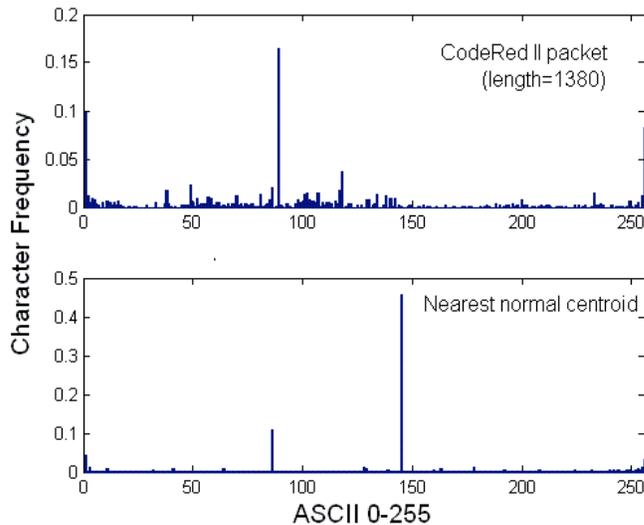


Status Quo v. Anomaly Detection

	Statu Quo	Anomaly Detection
Known Threats	Excellent	Fair
Installation	Excellent	Poor
Understanding Alerts	Excellent	Poor
Evasive & New Threats	Poor	Good
Threat Scalability	Poor	Good
Accuracy	Fair	Fair

What Can Statistical Payload Analysis (SPA) Do?

CR II Distribution v. Normal-Packets Distribution



Copyright 2007, CounterStorm, Inc.

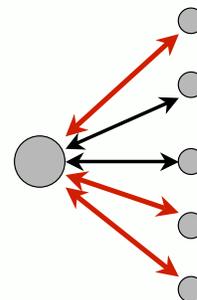
May, 2007

9

Finding Botnets with Statistical Payload Analysis (SPA)

DHS Funded

- **Detects anomalous payloads**
 - Compared with previously seen traffic on that network
 - Correlate anomalies by source or destination
- **Anomalous payloads are common in botnets**
 - Asking hosts to behave differently
 - Difficult for any botnet to avoid detection
- **Creation**
 - Exploits – known and zero-day
- **Command & Control**
 - Tunneled traffic (IRC over HTTP)
 - Unexpected encrypted/compressed traffic
- **Action**
 - Data exfiltration

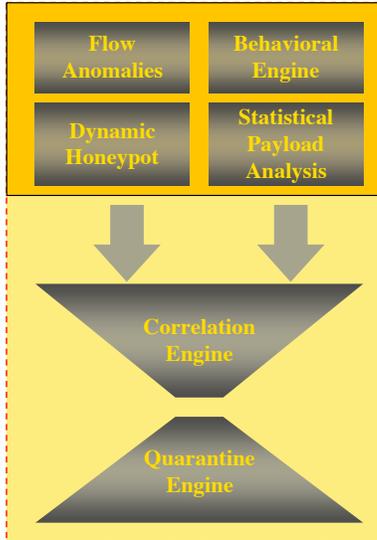


Copyright 2007, CounterStorm, Inc.

May, 2007

10

CounterStorm-1 Appliance: How It Works



- **Multiple advanced detection engines** **Sense**
 - Statistical payload analysis (SPA)
 - Recognizes exploits at layer 7 without signatures
 - Behavioral attack recognition
 - Recognizes initial attack propagation patterns
 - Requires no signatures or protocol rules
 - Flow anomaly detection
 - Dynamically baselines network traffic to detect anomalous activity
 - Dynamic honeypot
 - Detects attacks on unused network addresses
- **Dynamic real-time correlation** **Infer**
 - Processes evidence from multiple sensor engines
 - Eliminates false positives
- **Quarantine engine** **Act**
 - Applies flexible response policy to stop attacks

Copyright 2007, CounterStorm, Inc.

May, 2007

11

CounterStorm-1 | MONITOR | ANALYZE | REPORT | CONFIGURE | Logged in as admin | HELP | LOG OUT

Analyze - All active cases | New alarms [4] | EMERGENCY RESPONSE

Browse Activity

Location: All | Affected services: All | Start date: | Time: | End date: | Time: | Rows: 50 | Go

View: All active cases

Take Action | Mark As | Pause | Add | 1 out of 6 are blocked | Results 1 - 6 out of 6 | 1

	Status	Who	Where	What	When	Activity length	Targets	Technique	Blocked until
<input type="checkbox"/>	Open	10.20.20.24	Accounting	TCP/80	9:49:59 pm Dec 21	5 hours 30 minutes	43817		
<p>Detection reason: Local-segment scanning, Intranet scanning, Fast scanning</p> <p>First activity noticed: 4:19:26 PM 12/21/06</p> <p>Last activity noticed: 9:49:59 PM 12/21/06</p> <p>Time first blocked: N/A</p> <p>Quarantine technique:</p> <p>Switch name:</p> <p>Switch port:</p> <p>VPN username:</p> <p>VPN gateway name:</p> <p>Possible causes: IIS iisadmin access, CodeRed, Poison Null Byte,...</p> <p>Host name: acct24-ds.counterstorm.com</p> <p>Operating system: Windows XP SP1+, 2000 SP3</p> <p>NetBIOS name: 10-20-20-24</p> <p>Username: DWIGHTC</p> <p>MAC address: 00:0c:29:ef:b4:ba</p> <p>VLAN:</p> <p>Sensor: cs-core.counterstorm.com</p> <p>Segments: Accounting</p> <p>Assigned to: Richard Harrison</p> <p>Previous status: Open</p> <p>Label: Not so serious</p> <p>Score breakdown</p> <p># of services: 1 # of alarms: 12 # of sensors: 1 Duration: 5 hours 30 minutes</p> <p>Investigate</p>									
<input type="checkbox"/>	Open	10.20.20.20	Accounting	TCP/80	9:35:18 pm Dec 21	5 hours 38 minutes	58678		
<input type="checkbox"/>	New	10.20.20.17	Accounting	UDP/777 ...	4:18:34 pm Dec 21	8 seconds	80		
<input type="checkbox"/>	New	10.20.20.14	Accounting	UDP/363 ...	4:16:41 pm Dec 21	1 minute	90		March, 2007

Copyright 2007, CounterStorm, Inc.

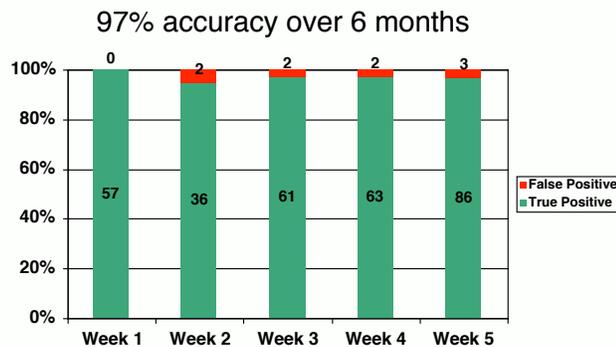
Healthcare Case Study

The customer:

- East coast community hospital system
- 7 centrally managed hospital facilities with 2 data centers
- 2,700 bed facility, 23,000 employees, 4,600 physicians
- Diverse user community

CounterStorm Detected:

- Zero-day attacks
- Botnets
- Root-kits
- P2P
- Worms



Copyright 2007, CounterStorm, Inc.

May, 2007

13

R&D to Strengthen Anomaly Detection's Advantages

- **Evasive and new threats**
 - Training on dirty data
 - Anit-mimicry via combinatorics
- **Scalability to threat volume, velocity and variety**
 - Scalable use of signaures
 - Account for new attack modes
 - Web 2.0/user content
 - IPv6
 - Virtualization
 - Continuous pressure from Moore's Law
- **Accuracy Improvements**
 - T/F positives, false negatives
 - Protocol/modality-specific
 - Correlation/inference techniques

Copyright 2007, CounterStorm, Inc.

May, 2007

14



R&D to Make Anomaly Detect Competitive

- **Coverage of known threats**
 - Update test methodologies to account for training
 - Available / relevant zoos
 - Communicate results
- **Alert understandability**
 - What's an anomaly mean?
 - What do correlated anomalies mean?
 - How can we provide autonomic explanations?
 - How to "pull the lose thread?"
- **Installation**
 - Plug and play
 - Environment drift



Recap

- **Statistical Anomaly Detection is a disruptive technology to the incumbents of signatures, policies and rules for detecting the most serious threats to information systems and networks**
- **Statistical Anomaly Detection now has a niche in improving federal cyber security where incumbent solutions are significantly inadequate**
- **Statistical Anomaly Detection needs R&D investments to improve the demonstratable scope of threats detected, understanding alerts and threat scalability**

Application of Risk Management Principles in Information Technology Permitting Decision Makers to Target Funding for Security Investments

by Dr. Martin A. Carmichael, Chief Information Officer, The Rader Network,
Colorado Springs, Colorado

Abstract

Federal, state, and corporate security officials find it difficult to communicate with decision makers – in the business terms they understand – why investing in information technology security is an imperative¹. In particular, Chief Information Officers (CIOs) and Chief Information Security Officers (CISOs) must effectively explain that security is not an overhead cost but a business enabler, allowing organizations to comply with the Sarbanes-Oxley Act and other governance regulations. This is largely due to the fact there has not been a cost effective, efficient, or comprehensive tool to establish a mathematically-provable business case. Statistical analysis software that applies risk management principles to information technology has been developed to resolve this inadequacy. Technology Risk Manager (TRM) software is specifically designed as a comprehensive strategy to meet the challenge of creating defensible, cyber security business cases.

Problem

The combination of increased vulnerability, increased stakes, and increased threats make cyber security one of the most important emerging challenges in the evolution of modern cyber infrastructure design and deployment.² Correspondingly, there is increased difficulty to establish a “return-on-investment” business case that decision makers can understand and appreciate as cyber infrastructures become larger, more complex, and more distributed. Security officials must be able to communicate the importance of information technology security in terms easily understood by those who control investment in cyber infrastructure design and deployment to prevent loss of life ... and the loss of our way of life.

“If you can’t measure it, you can’t manage it.”³ Regardless, organizations develop budgets and expend funds without an effective means of measuring information technology security. For example, Gartner Incorporated advises organizations to spend from 4% to 6% of their information technology budgets on information security. Yet determining “bang for the buck” is currently subjective, indefinite, ad hoc, indefensible, and lacking in scientific methodology.

Information security budgets are expected to increase 4.5 percent in the next year.⁴ As with the Maginot Line, it is becoming increasingly difficult to build affordable information technology protection defenses.⁵ Organizations that can accurately measure information technology risk can reduce costs.

Approach

Not all numbers qualify as metrics. The so-called “metrics” currently captured during network scans are simply counts -- patches to upload, vulnerabilities noted, past security compromises, etc. The metrics that result from TRM analyses are true metrics that can be scientifically scrutinized. TRM metrics are numerical facts based on statistical analyses. TRM metrics are objective, quantitative, repeatable, and defensible. TRM metrics predict the likelihood of security failure within an information technology environment along each of the four dimensions of risk: confidentiality, integrity, availability, and audit. TRM Risk Indices describe the likelihood of security failure as a statistically-derived percentage along each risk dimension within a defined period of time and a baseline threat. Each process on an enterprise is evaluated for its security characteristics. Adjacencies are measured and the results are aggregated to determine each host’s security characteristics. Adjacencies are measured a second time and the host calculations are aggregated to calculate the Risk Indices. TRM provides a systemic view of information technology security, empowering managers to direct this activity with the same precision they use to manage risks in their other resources. Upon establishing a baseline, a TRM-certified user can accurately model and simulate how strategies and technologies can be best used to protect assets. Decisions can then be made based on proactive analyses and predictive modeling. TRM reduces the certification and accreditation reporting process from months to weeks. TRM converts information assurance from a subjective to an objective management process.

Results

We can only trust what can be quantitatively measured. Implementation of the modeled and simulated recommendations following a TRM analysis historically have resulted in an average 19% decrease in overall risk within an enterprise.

Conclusion

TRM is commercially available software specifically designed to empower security officials to determine return on investment in objective, quantitative, repeatable, defensible, and predictive terms. The selective outputs of a TRM analysis yield results that can be certified and accredited versus in hypothesized prose. TRM provides quantitative metrics of information technology security, which enable users to specify security requirements, formulate security claims, and certify security properties as a comprehensive strategy to meet the challenge of creating defensible, cyber security business cases. TRM is specifically designed as a comprehensive strategy to meet the challenge of cyber security in the 21st century. TRM permits information technology security professionals to shift their focus away from winning battles towards the strategies to win the war. TRM will elevate trust in critical infrastructures. The Rader Network sincerely believes that TRM will fundamentally change information technology security as we currently know it.



Bibliography:

- ¹ "[Security Chiefs Fail to Justify Regulation Spending](#)", Paul Muncaster, Financial Director Magazine, United Kingdom, 19 Sep 06
- ² "[Cyber Security and Information Infrastructure Research Workshop](#)", Dr. Frederick T. Sheldon, Computational Sciences and Engineering Division, Oak Ridge National Laboratory, Oak Ridge, TN, 23 Mar 07
- ³ Peter Drucker, quoted in "[Intelligent Sustainment and Renewal of Department of Energy Facilities and Infrastructure](#)", Chapter 4, Infrastructure Management Performance Measures, Committee on the Renewal of Department of Energy Infrastructure, Board on Infrastructure and the Constructed Environment, Division on Engineering and Physical Sciences, National Research Council of the National Academies, International Standard Book Number 0-309-54652-4, Copyright 2004 by the National Academy of Sciences. All rights reserved.
- ⁴ "[Gartner: Security Costs Fall With Good Policies](#)", by Jeremy Kirk, IDG News Service, 18 Sep 06.
- ⁵ "[The Thickness of Concrete on the Maginot Line](#)", Published by Infowar.Com & Interpact, Incorporated, with permission from The Honorable Paul A. Strassmann, (former) Director of Defense Information, U.S. Department of Defense. Undated.



**Presentation and Demonstration
to participants of the
Cyber Security and Information Infrastructure Research Workshop
14-15 May 07**



Introductions



- * Dr. Martin Carmichael
Chief Information Officer, The Rader Network

- * Mike Rader
President / CEO, The Rader Network

- * Katie Carmichael
Principal Engineer, Technology Risk Manager



- * Quantitative metrics are desirable, and should be attempted in all verification, validation, and accreditation activities
- * Quantitative metrics in information assurance eliminate ambiguity in computational-experimental comparisons
- * Prior to TRM, obtaining quantitative metrics in information assurance and defining their associated success criteria was not possible



The Challenge (Narrative View)



Non-Parametric

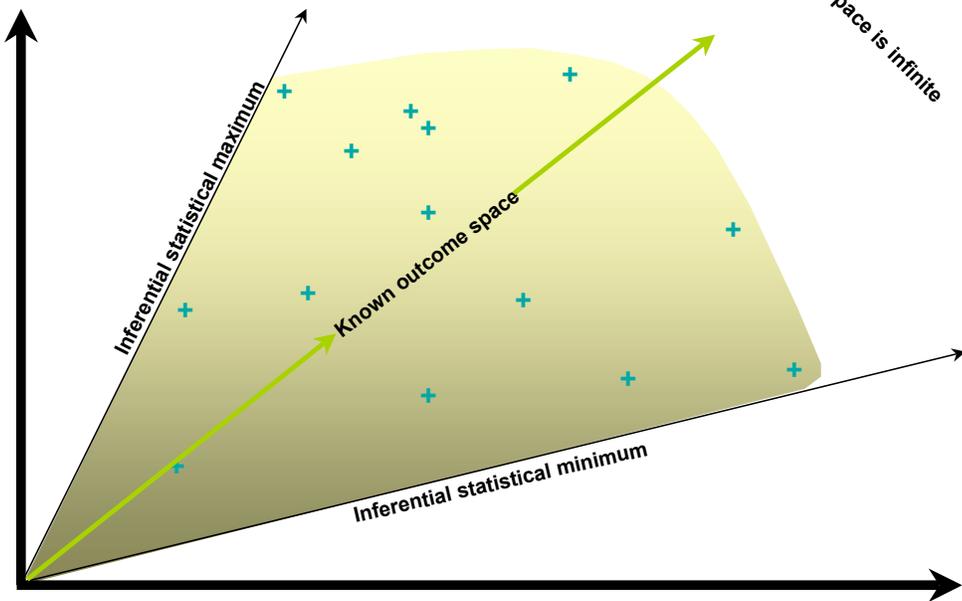
Parametric

<ul style="list-style-type: none"> * National Infrastructure Protection Plan (NIPP) * National Institute of Standards & Technology (NIST) * National Security Agency (NSA) * Security Technical Implementation Guides (STIGs) * Department of Defense (DoD): <ul style="list-style-type: none"> ** DOD Information Assurance Certification and Accreditation Program (DIACAP) ** DoD Information Technology Security Certification and Accreditation Process (DITSCAP) * Defense Information Systems Agency (DISA) * Defense Information Technology Contracting Office (DITCO) * Defense Modeling and Simulation Office (DMSO) * U.S. Air Force (USAF) * Health Insurance Portability and Accountability Act of 1996 * Public Key Infrastructure 	<p style="text-align: center;">Prior to TRM, not possible in information assurance</p>
--	---

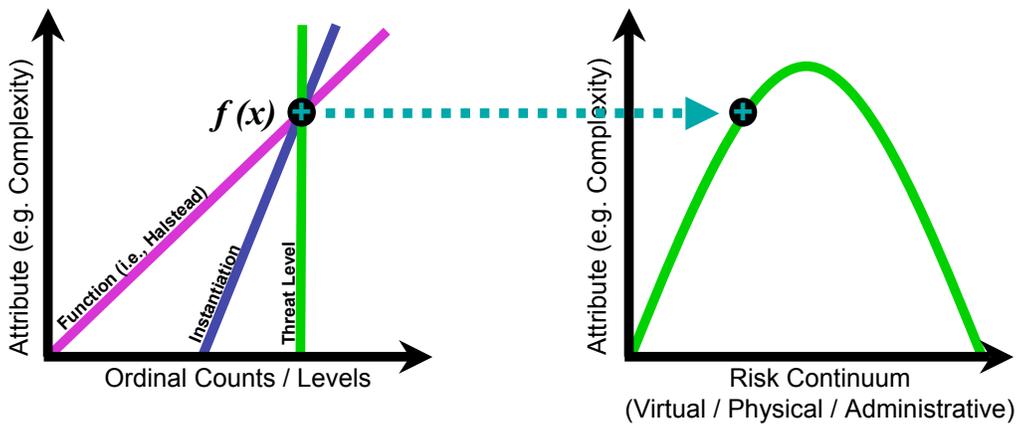


The Challenge (Scientific View)

Possible outcome space is infinite

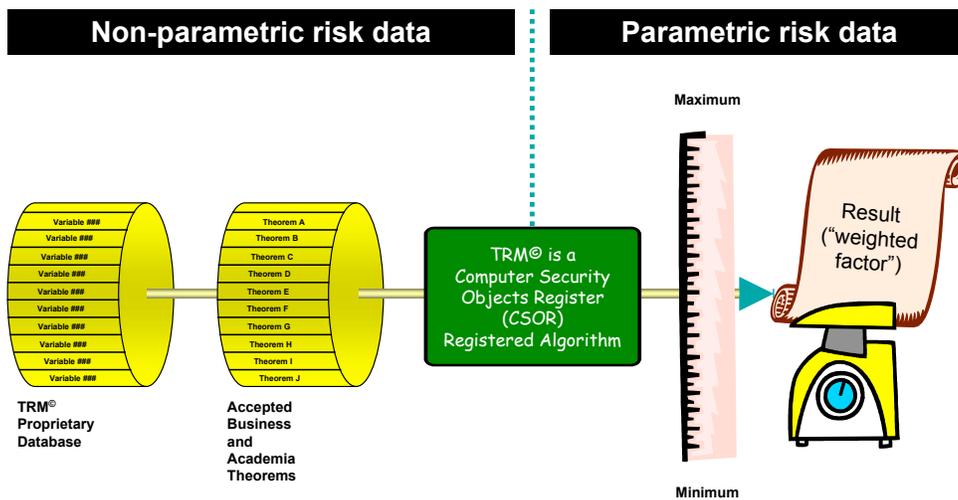


The Challenge (Mathematical View)



Example of only 1 of 22 interactive formula

The Challenge (TRM View)



➔ Only TRM[®] can convert non-parametric values to parametric values ➔

TRM – Calculating Metrics



- * Each process on the network is evaluated for its security characteristics
- * Adjacencies are measured and the results are aggregated to determine each host's security characteristics
- * Adjacencies are measured a second time and the host calculations are aggregated to calculate the Risk Indices



Definitions



- * Not all numbers qualify as metrics. True metrics are numerical facts based on statistical analyses:
 - ** **Objective**: the number is based strictly on mathematical risk characteristics
 - ** **Quantitative**: the number represents the only true statistical knowledge
 - ** **Repeatable**: mathematical or scientific facts must always be repeatable
 - ** **Defensible**: statistical analysis is a long-established and highly respected science



Definitions (continued)



- * **TRM Metrics** are predictive:
 - ** TRM metrics predict the likelihood of a **future** security failure along each of the Four Dimensions of Risk
 - ** Other “metrics” consist of counts -- patches to upload, vulnerabilities noted, past security compromises, etc



Definitions (continued)



* The Four Dimensions of Risk

- ** **Confidentiality**: measures how well an organization can authenticate and authorize its users
- ** **Integrity**: shows how reliable the systems of an organization are in keeping information accurate
- ** **Availability**: measures how likely an authorized individual is to be able to access appropriate information
- ** **Audit**: shows how effectively an organization can determine which individuals accessed what data while on their systems



Definitions (continued)



- * **TRM Risk Indices** describe the likelihood of failure (as a percentage) along the specified dimension given three months and a hacker of average abilities

High Numbers are Bad

Risk Profile	
	Risk Index
Confidentiality	94.393
Integrity	60.980
Availability	64.681
Audit	96.381

Emphasis



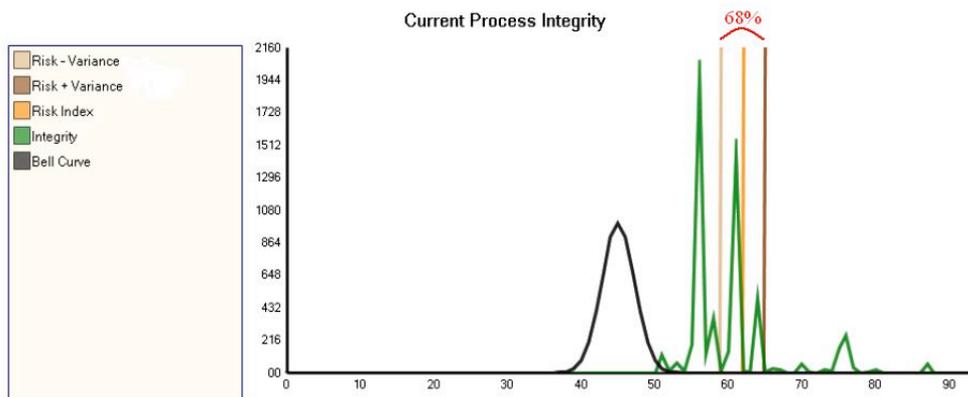
- * **TRM Return on Investment** – TRM’s metrics enable straightforward ROI calculations

Risk Profile	
	Risk Index
Current Confidentiality	94.393
Simulated Confidentiality	66.251
Current Integrity	60.980
Simulated Integrity	45.307
Current Availability	64.681
Simulated Availability	68.255
Current Audit	96.381
Simulated Audit	58.328

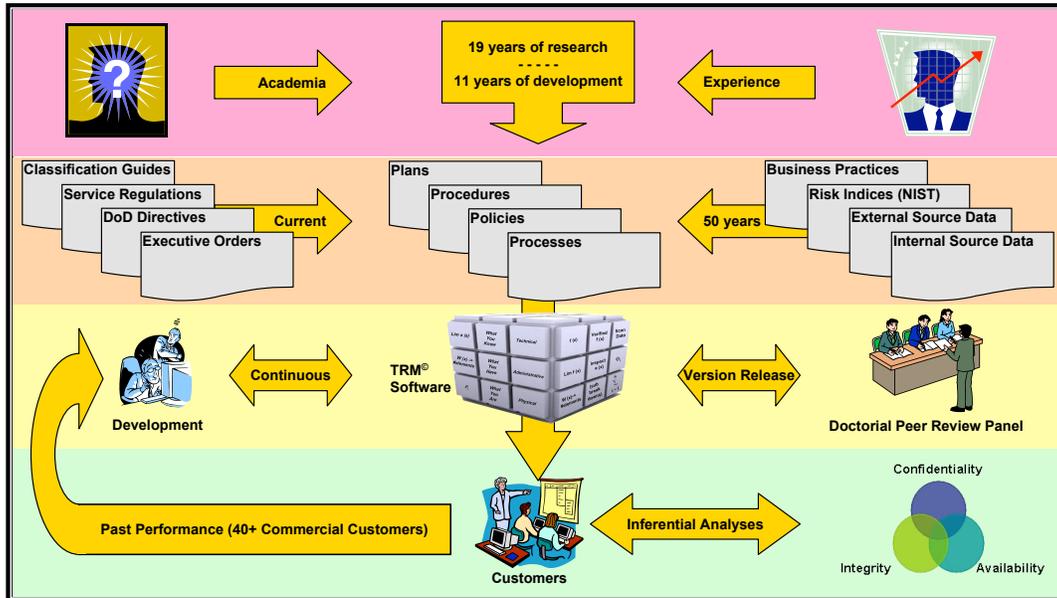
Emphasis (continued)



- * **TRM’s Consistency and Variance** -- reducing the Risk Index is not the whole story. There needs to be consistency across the network.



TRM – Maturity



TRM – Product Comparisons



Parameters	Harris (STAT)	eEye (Retina)	IBM (ISS)	TRM
Metrics and Risk Analysis	X	X	X	▲
C I A A	X	X	X	▲
Quantitative, Objective, & Repeatable	X	X	X	▲
Engineering Principles Based	X	X	X	▲
Adjacencies	X	X	X	▲
Baselines	X	X	X	▲
Modeling and Simulations	X	X	X	▲
Trend Analysis	X	X	X	▲
Administrative Rights Required	Yes	Yes	Yes	No



* ***“If you can’t measure it, you can’t manage it”***

Peter Drucker

* TRM enables you to both measure and manage information assurance with unprecedented accuracy



Principal Contacts



Mike Rader	MichaelRader@ RaderNetwork.com	(719) 930-1183
Katie Carmichael	KatieCarmichael@ RaderNetwork.com	(214) 794-9615
Martin Carmichael	MartinCarmichael@ RaderNetwork.com	(214) 794-9510



Questions and Answers (Product Demonstration)



Joint Institute for
Computational Sciences



Thank you

Secure Coding Standards

Robert C. Seacord
CERT/CC
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213 USA
+1-412-268-7608
rcs@cert.org

Jason A. Rafail
CERT/CC
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213 USA
+1-412-268-6305
jrafail@cert.org

ABSTRACT

Secure coding standards define rules and recommendations to guide the development of secure software systems. Establishing secure coding standards provides a basis for secure system development as well as a common set of criteria that can be used to measure and evaluate software development efforts and software development tools and processes. This paper describes plans by the CERT/Coordination Center at the Software Engineering Institute at Carnegie Mellon University to establish, through a coordinated community effort, a set of secure coding standards for commonly used programming languages.

Keywords

Security, Standardization, Programming languages.

1. INTRODUCTION

Society's increased dependency on networked software systems has been matched by an increase in the number of attacks aimed at these systems. These attacks—directed at governments, corporations, educational institutions, and individuals—have resulted in loss and compromise of sensitive data, system damage, lost productivity, and financial loss [19].

Software vulnerability reports continue to grow at an alarming rate [1] and a significant number of them result in technical alerts [2]. To address this growing threat, the introduction of software vulnerabilities during software development and ongoing maintenance must be significantly curtailed.

An essential element of secure software development is well documented and enforceable coding standards. Coding standards encourage programmers to follow a uniform set of rules and guidelines determined by the requirements of the project and organization, rather than by the programmer's familiarity or preference. Once established, these standards can be used as a metric to evaluate source code (using manual or automated processes) to determine compliance with the standard.

There are numerous available sources, both online and in print,

containing coding guidelines, best practices, suggestions, and tips. For example, the following books have been published containing C/C++ programming languages rules and guidelines:

- C++ Coding Standards: 101 Rules, Guidelines, and Best Practices [21]
- Effective C++ : 55 Specific Ways to Improve Your Programs and Designs (3rd Edition) [10]
- More Effective C++: 35 New Ways to Improve Your Programs and Designs [11]
- Effective STL: 50 Specific Ways to Improve Your Use of the Standard Template Library [12]
- C++ Programming Guidelines [16]
- C Programming Guidelines [17]

Industry-specific standards such as the Motor Industry Software Reliability Association (MISRA) Guidelines for the use of the C language in critical systems [13] have also been published. Additionally, many companies have internal coding standards. An example of a publicly released coding standard is the Joint Strike Fighter Air Vehicle C++ Coding Standards [9].

Many online sources of coding practices and coding rules also exist, including the Build Security In web site [4] sponsored by the U.S. Department of Homeland Security (DHS) National Cyber Security Division. The SAMATE Reference Dataset (SRD), maintained by NIST [15], provides a set of programs with known weaknesses in code, design, or architecture that can lead to exploitable vulnerabilities. The Common Weaknesses Enumeration (CWE), maintained by MITRE, is a dictionary of known security weaknesses in code, design, and architecture that can lead to exploitable vulnerabilities [14].

With all these sources of information, it might seem that a secure coding standard for these languages would be unnecessary. However, none of these sources provides a prescriptive set of secure coding standards that can be uniformly applied in the development of a software system. This conclusion is reinforced by the Secure Software Assurance Common Body of Knowledge [18] published by the U.S. Department of Homeland Security, which laments the “lack of public standards as such for secure programming.”

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

2. SCOPE

At one extreme, a secure coding standard can be developed for a particular release of a compiler from a particular vendor. At the other extreme, the standards can be designed to be not only compiler independent but also language independent.

A coding standard for a particular compiler release has the largest possible benefit to the smallest group of users. Targeting a particular compiler allows for the definition of rules and guidelines that deal specifically with the peculiarities of that implementation, including defects in the implementation and non-standard extensions. At the other extreme, a language-independent coding standard has the least possible benefit to the largest possible group of users, as the rules and guidelines specified at this level of abstraction are largely notional.

The secure coding standards proposed by CERT are based on documented standard language versions as defined by official or *de facto* standards organizations. For example, secure coding standards are planned for the following languages:

- C programming language (ISO/IEC 9899:1999) [5]
- C++ programming language (ISO/IEC 9899:1999) [6]
- Sun Microsystems' Java2 Platform Standard Edition 5.0 API Specification [20]
- C# programming language (ISO/IEC 23270:2003) [7]

Applicable technical corrigenda and documented language extensions such as the ISO/IEC TR 24731 extensions to the C library [8] will also be considered.

The scope allows specific guidance to be provided to broad classes of users. Programming language standards, like those created by ISO/IEC, are primarily intended for compiler implementers. Secure coding standards are ancillary documents that provide rules and guidance directly to developers who program languages defined by these standards.

3. GOALS

The goal of each coding standard is to define a set of rules that are necessary (but not sufficient) to ensure the security of software systems developing in the respective programming languages.

A secure coding standard consists of *rules* and *recommendations*. Coding practices are defined to be rules when all of the following conditions are met

1. Violation of the coding practice will result in a security flaw that may result in an exploitable vulnerability.
2. There is an enumerable set of exceptional conditions (or no such conditions) where violating the coding practice is necessary to ensure the correct behavior for the program.
3. Conformance to the coding practice can be verified.

Rules must be followed to claim compliance with a standard unless an exceptional condition exists. If an exceptional condition is claimed, the exception must correspond to a pre-defined exceptional condition and the application of this exception must be documented in the source code.

Recommendations are guidelines or suggestions. Coding practices are defined to be recommendations when all of the following conditions are met

1. Application of the coding practice is likely to improve system security.
2. One or more of the requirements necessary for a coding practice to be considered a rule cannot be met.

Compliance with recommendations is not necessary to claim compliance with a coding standard. It is possible, however, to claim compliance with one or more verifiable guidelines. The set of recommendations that a particular development effort adopts depends on the security requirements of the final software product. Projects with high-security requirements can dedicate more resources to security, and are thus likely to adopt a larger set of recommendations.

4. DEVELOPMENT PROCESS

The development of a secure coding standard for any programming language is a difficult undertaking that requires significant community involvement. To produce standards of the highest possible quality, CERT is implementing the following development process:

1. Rules and recommendations for a coding standard are solicited from the communities involved in the development and application of each programming language, including the formal or *de facto* standard bodies responsible for the documented standard.
2. These rules and recommendations are edited by senior members of the CERT technical staff for content and style and placed in the Secure Coding area of CERT web site for comment and review [3].
3. The user community may then comment on the publically posted content using threaded discussions and other communication tools. Once a consensus develops that the rule or recommendation is appropriate and correct the final rule is incorporated into the coding standard.

Various groups, including the ISO/IEC JTC1/SC22/WG14 international standardization working group for the C programming language have expressed an interest in supporting this model.

5. USAGE

These rules may be extended with organization-specific rules. However, the rules contained in a standard must be obeyed to claim compliance with the standard.

Training may be developed to educate software professionals regarding the appropriate application of secure coding standards. After passing an examination, these trained programmers may also be certified as secure coding professionals.

Once a secure coding standard has been established, tools can be developed or modified to determine compliance with the standard. One of the conditions for a coding practice to be considered a rule is that conformance can be verified. Verification can be performed manually or automated. Manual verification can be labour intensive and error prone. Tool verification is also problematic in that the ability of a static analysis tool to detect all violations of a rule must be proven for each product release, to detect regression errors. Even with these challenges, automated validation may be the only economically scalable solution to validate conformance with the coding standard.

Software analysis tools may be certified as being able to verify compliance with the secure coding standard. Compliant software systems may be certified as compliant by a properly authorized certification body by the application of certified tools.

6. SYSTEM QUALITIES

Security is one of many system attributes that must be considered in the selection and application of a coding standard. Other attributes of interest include safety, portability, reliability, availability, maintainability, readability, and performance.

Many of these attributes are interrelated in interesting ways. For example, readability is an attribute of maintainability; both are important for limiting the introduction of defects during maintenance that could result in security flaws or reliability issues. Reliability and availability require proper resources management, which contributes also to the safety and security of the system. System attributes such as performance and security are often in conflict requiring tradeoffs to be considered.

The purpose of the secure coding standard is to promote software security. However, because of the relationship between security and other system attributes, the coding standards may provide recommendations that deal primarily with some other system attribute that also has a significant impact on security. The dual nature of these recommendations will be noted in the standard.

7. CONCLUSIONS

The development of secure coding standards is a necessary step to stem the ever-increasing threat from software vulnerabilities. Establishing secure coding standards allows for a common set of criteria that can be used to measure and evaluate software development efforts and software development tools and processes. Once established, secure coding standards can be incrementally improved, as a common understanding of existing problems and solutions allows for the development of more advanced security solutions.

8. ACKNOWLEDGMENTS

Thanks to Thomas Plum for suggesting this idea, John Benito for supporting this effort, and Hal Burch for his insights. Thanks to Jason Rafail, Jeff Gennari, Allen Householder, Chad Dougherty, and Claire Dixon for their review and thoughtful comments.

9. REFERENCES

- [1] CERT/CC. See http://www.cert.org/stats/cert_stats.html for current statistics.
- [2] CERT/CC. US-CERT's Technical Cyber Security Alerts. <http://www.us-cert.gov/cas/techalerts/index.html>
- [3] CERT/CC. Secure Coding web site. <http://www.cert.org/secure-coding/>
- [4] DHS. Build Security In web site. See <https://buildsecurityin.us-cert.gov/>
- [5] INCITS/ISO/IEC 9899-1999. Programming Languages — C, Second Edition, 1999.
- [6] INCITS/ISO/IEC 14882-2003. Programming Languages — C++, Second Edition, 2003.
- [7] INCITS/ISO/IEC 23270-2003. Information technology - C# Language Specification, 2003.
- [8] ISO/IEC WDTR 24731. Specification for Secure C Library Functions, 2004.
- [9] Lockheed Martin. Joint Strike Fighter Air Vehicle C++ Coding Standards for the System Development and Demonstration Program. Document Number 2RDU00001 Rev C. December 2005.
- [10] Meyers, Scott. Effective C++ : 55 Specific Ways to Improve Your Programs and Designs (3rd Edition). Addison-Wesley Professional. (September 2, 1997)
- [11] Meyers, Scott. More Effective C++: 35 New Ways to Improve Your Programs and Designs. Addison-Wesley Professional. (December 29, 1995)
- [12] Meyers, Scott. Effective STL: 50 Specific Ways to Improve Your Use of the Standard Template Library. Addison-Wesley Professional. (June 6, 2001)
- [13] MISRA C: 2004 Guidelines for the use of the C language in critical systems. MIRA Limited. Warwickshire, UK. October 2004. ISBN 0 9524156 4
- [14] MITRE. Common Weaknesses Enumeration (CWE). See <http://cve.mitre.org/cwe/>
- [15] NIST. SAMATE Reference Dataset (SRD). See <http://samate.nist.gov/SRD/srdFiles/>
- [16] Plum, Thomas. C Programming Guidelines. Plum Hall; 2nd edition (June 1989). ISBN: 0911537074.
- [17] Plum, Thomas. C++ Programming. Plum Hall (November 1991) ISBN: 0911537104.
- [18] Redwine, Jr. Samuel T, Editor. Secure Software Assurance: A Guide to the Common Body of Knowledge to Produce, Acquire, and Sustain Secure Software Draft Version 0.9. January 2006.
- [19] Seacord, R. *Secure Coding in C and C++*. Addison-Wesley, 2005. See <http://www.cert.org/books/secure-coding> for news and errata.
- [20] Sun Microsystems. Java2 Platform Standard Edition 5.0 API Specification, 2004. <http://java.sun.com/j2se/1.5.0/docs/api/index.html>
- [21] Sutter, Herb. Alexandrescu, Andrei. C++ Coding Standards: 101 Rules, Guidelines, and Best Practices. Addison-Wesley Professional (October 25, 2004). ISBN: 0321113586.

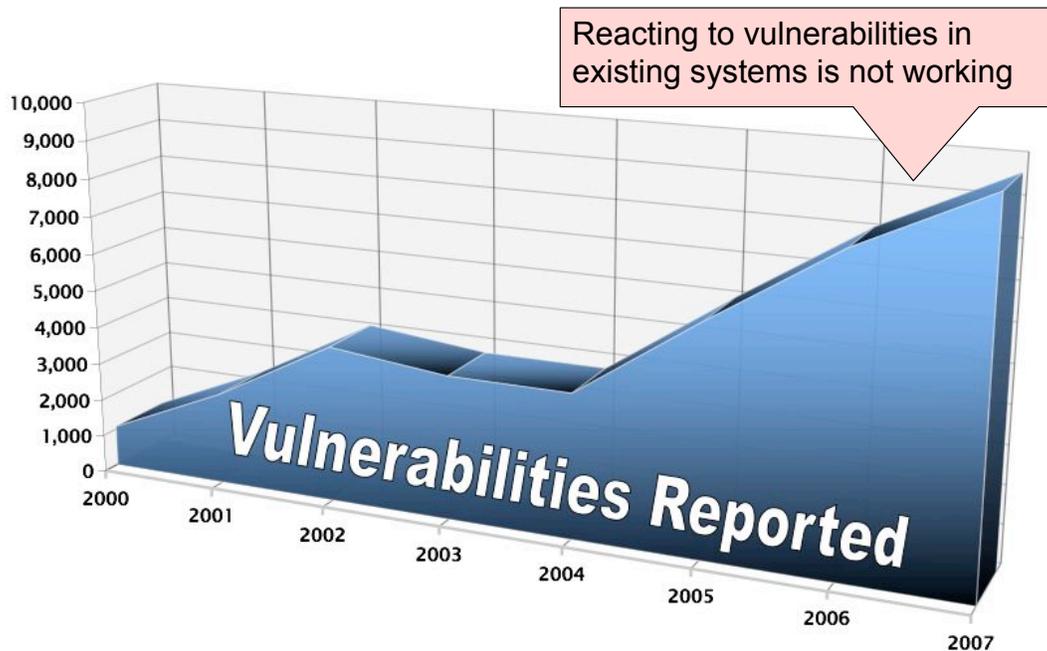


Secure Coding Initiative

Jason A. Rafail
Monday, May 14th, 2007



Increasing Vulnerabilities



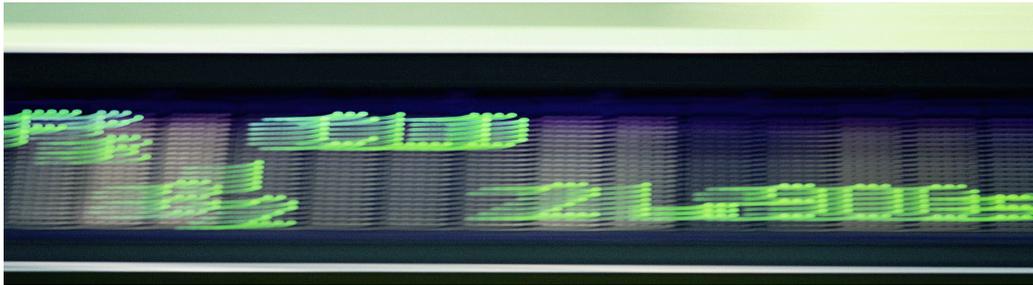
Vulnerabilities Cost Vendors

A study based on real vulnerability announcements in 1999-2004 revealed: an average **drop** of the concerned vendor's **stock price** of **0.6%** after each vulnerability announcement

– Tehang / Wattal, Carnegie Mellon University, 2004

"Impact of Software Vulnerability Announcements on the Market Value of Software Vendors – an Empirical Investigation"

... not to mention the damage to the vendor's reputation



Most Vulnerabilities caused by Programming Errors

64% of the vulnerabilities in NVD in 2004 are due to programming errors

- 51% of those due to classic errors like buffer overflows, cross-site-scripting, injection flaws
- *Heffley/Meunier (2004): Can Source Code Auditing Software Identify Common Vulnerabilities and Be Used to Evaluate Software Security?*

Cross-site scripting, SQL injection at top of the statistics (CVE, Bugtraq) in 2006

"We wouldn't need so much network security if we didn't have such bad software security"

--Bruce Schneier

Unexpected Integer Values



An **unexpected value** is not one you would expect to get using a pencil and paper

Unexpected values are a common source of **software vulnerabilities** (even when this behavior is correct).



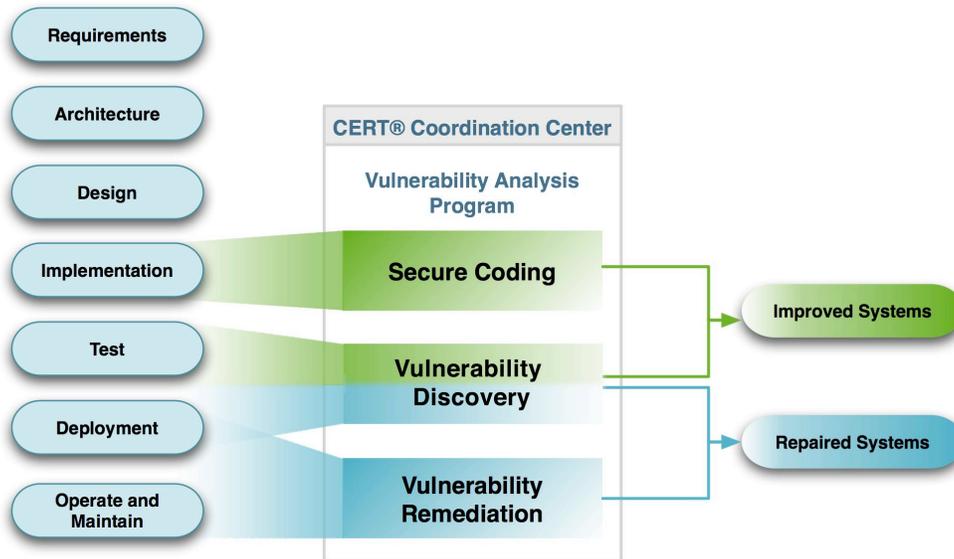
Fun With Integers

```
char x, y;  
x = -128;  
y = -x;
```

```
if (x == y) puts("1");  
if ((x - y) == 0) puts("2");  
if ((x + y) == 2 * x) puts("3");  
if (x != -y) puts("4");
```



CERT Vulnerability Analysis



CERT Secure Coding Initiative

Work with [software developers](#) and [software development organizations](#) to eliminate vulnerabilities resulting from coding errors before they are deployed.

- **Reduce** the number of vulnerabilities to a level where they can be handled by computer security incident response teams (CSIRTs)
- **Decrease** remediation costs by eliminating vulnerabilities *before* software is deployed

Overall Thrusts

Advance the **state of the practice** in secure coding

Identify common programming errors that lead to software vulnerabilities

Establish standard secure coding practices

Educate software developers

Current Capabilities

Secure Coding in C and C++

- Addison-Wesley book
- Training

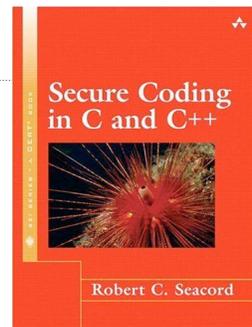
Secure coding web pages

www.cert.org/secure-coding/

Secure string and integer library development

Involvement in international standards activities:

- ISO/IEC JTC1/SC22/WG14 C programming language international standardization working group
- ISO/IEC JTC1/SC22 OWG Vulnerabilities



Current and Planned Efforts

CERT Secure Coding Standards

- C and C++ Programming Language
- Community development process

Training courses

- Direct offerings
- Partnered with industry

Software Validation and Verification

- Partner with software tool vendors to validate conformance to secure coding standards
- Partner with software development organizations to evaluate the application of secure coding standards

CERT Secure Coding Standards

Identify coding practices that can be used to improve the security of software systems under development

Specific objectives include

- avoiding undefined behaviour
- avoiding implementation defined behaviour
- improving clarity for review and maintenance
- providing a consistent style across a program or set of programs
- avoiding common programmer errors
- incorporating good practice, particularly with regard to 'future proofing'

Scope

The secure coding standards proposed by CERT are based on documented standard language versions as defined by official or *de facto* standards organizations.

Secure coding standards are under development for:

- C programming language (ISO/IEC 9899:1999)
- C++ programming language (ISO/IEC 14882-2003)

Applicable technical corrigenda and documented language extensions such as the ISO/IEC TR 24731 extensions to the C library are also included.

Secure Coding Web Site (Wiki)

Secure Coding

CERT Secure Coding Standards

Added by Confluence Administrator, last edited by Robert Seacord on Feb 28, 2007 (view change)

Labels: (None)

Welcome to the Secure Coding Web Site

This web site exists to support the development of secure coding standards for commonly used programming languages such as C and C++. These standards are being developed through a broad-based community effort including the CERT Secure Coding Initiative and members of the software development and software security communities. For a further explanation of this effort please read the [Rationale](#).

As this is a development web site, many of the pages on this web site are incomplete or contain errors. If you are interested in furthering this effort you may comment on existing items or send recommendations to [secure-coding at cert dot org](mailto:secure-coding@cert dot org). You may also apply for an account to directly edit content on the site. Before using this site, please familiarize yourself with the [Terms and Conditions](#).

The [Top 10 Secure Coding Practices](#) provides some language independent recommendations.

Secure Coding Standards

[CERT C Programming Language Secure Coding Standard](#)

[CERT C++ Programming Language Secure Coding Standard](#)

We would like to acknowledge the contributions of the following [folks](#), and we look forward to seeing your name there as well.

Rules and Recommendations

Coding practices are classified as either rules or recommendations

- Rules need to be followed to claim **compliance**.
- Recommendations are **guidelines** or **suggestions**.

Rules

Coding practices are defined as **rules** when

- Violation of the coding practice will result in a security flaw that may result in an exploitable vulnerability.
- There is an enumerable set of exceptional conditions (or no such conditions) where violating the coding practice is necessary to ensure the correct behavior for the program.
- Conformance to the coding practice can be verified.

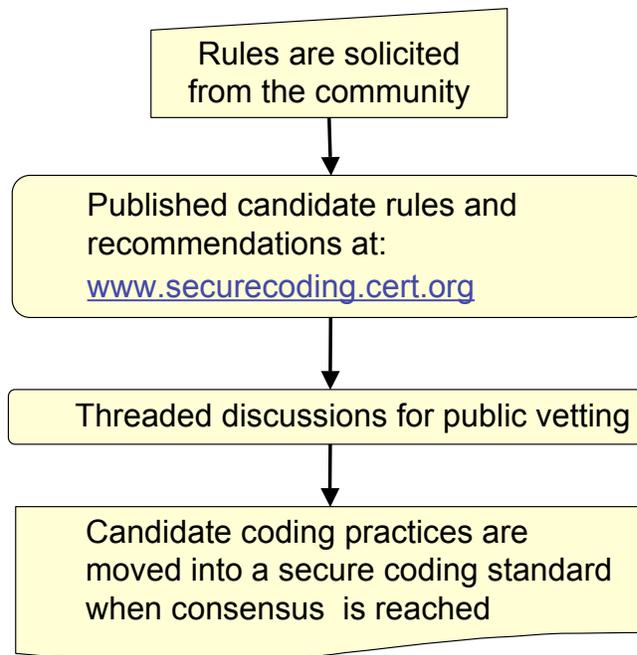
Recommendations

Coding practices are defined as **recommendations** when

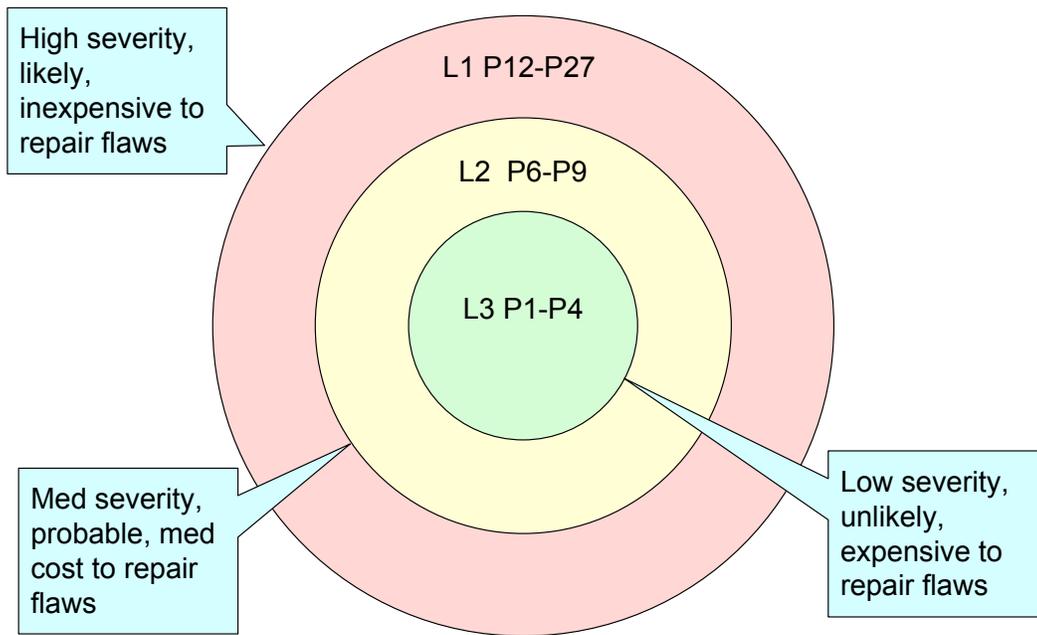
- Application of the coding practice is likely to improve system security.
- One or more of the requirements necessary for a coding practice to be considered as a rule cannot be met.

Community Development Process

Secure coding standards development is a **community effort**



Priorities and Levels



Rules

Rule	Severity	Likelihood	Remediation Cost	Priority	Level
FIO30-C	3 (high)	3 (probable)	3 (low)	P27	L1
FIO32-C	3 (high)	2 (probable)	1 (medium)	P6	L2
FIO33-C	1 (low)	1 (low)	3 (medium)	P3	L3
FIO34-C	2 (medium)	2 (probable)	2 (medium)	P8	L2
FIO35-C	1 (low)	1 (unlikely)	2 (medium)	P2	L3
FIO36-C	1 (low)	1 (unlikely)	3 (low)	P3	L3
FIO37-C	3 (high)	1 (unlikely)	2 (medium)	P6	L3
FIO38-C	2 (medium)	2 (probable)	2 (medium)	P8	L2
FIO39-C	2 (medium)	2 (probable)	2 (medium)	P8	L2
FIO40-C	2 (medium)	2 (probable)	2 (medium)	P8	L2
FIO41-C	2 (medium)	2 (probable)	2 (medium)	P8	L2
FIO42-C	2 (medium)	2 (probable)	2 (medium)	P8	L2
FIO43-C	3 (high)	3 (likely)	2 (low)	P18	L1

Applications

Establish secure coding practices within an organization

- may be extended with organization-specific rules
- cannot replace or remove existing rules

Train software professionals

Certify programmers in secure coding

Establish base-line requirements for software analysis tools

Certify software systems



Questions

For More Information

Visit CERT® web sites:

<http://www.cert.org/secure-coding/>

<https://www.securecoding.cert.org/>

Contact Presenter

Jason A. Rafail jrafail@cert.org

Contact CERT Coordination Center:

Software Engineering Institute

Carnegie Mellon University

4500 Fifth Avenue

Pittsburgh PA 15213-3890

USA

Hotline: **+1 412 268 7090**

**CERT/CC personnel answer 24x7, 365
days per year**

Fax: **+1 412 268 6989**

Mailto: cert@cert.org



Software Engineering Institute | Carnegie Mellon

23

Security in the Context of Dependability

Tacksoo Im

John D. McGregor

Clemson University

{tim,johnmc}@cs.clemson.edu

Abstract

Security, as an architectural quality, is often thought to be measured in terms of availability, confidentiality and integrity. These qualities are part of a broader quality - dependability. There are inherent tradeoffs among the qualities that define security and dependability. Architectural tactics, or architectural design decisions, that enhance one aspect of dependability can decrease security and vice-versa. In addition, this is a multi-scale problem in that different quality attributes are measured on different scales of reference including some that are not quantitative. In this paper we present a qualitative approach to managing interactions among the attributes used to define security.

Introduction

Defining a system architecture to support products that are secure requires the architect to address both functional and non-functional requirements. Quality-driven techniques require the architect to explicitly consider these non-functional requirements – referred to as quality attributes - during even the earliest architecture decisions. Techniques such as quality attribute workshops (QAW) and the architecture tradeoff analysis method (ATAM) [Kazman 00] are used to identify the desired quality attributes of a system but it is hard for the architect to design the system when several of the quality attributes interact with each other. That is, an architecture decomposition that enhances one attribute may degrade another. Managing the tradeoffs among qualities is hard for several reasons. It is hard for an architect to be knowledgeable about all the quality attributes that are involved in a particular system and some of the qualities, such as security, are not measured on quantitative scales while others are quantitative. There are reasoning frameworks that assist the architect in quantitatively analyzing quality attributes such as performance. But there are few techniques for reasoning about attributes that are qualitatively represented, such as security. In this paper, we present a qualitative approach to reasoning about security at the architectural stage.

Background

Dependability is defined as the degree to which trust can be justifiably placed on a computer system. This is usually taken to include the qualities of reliability, availability, safety, integrity, confidentiality and maintainability. [Avizienis 00] This subsumes the definition of security usually taken to include: availability, integrity, and confidentiality. In our work on designing dependable systems we have identified four interactions among the qualities within dependability that involve qualities related to security.

- Availability vs. Confidentiality
Design decisions that increase the availability of the system can decrease confidentiality by introducing prolonged exposure. [Warns 05]
- Availability vs. Integrity
Design decisions that increase the availability of the system can decrease integrity by exposing data for longer periods of time to the possibility of malicious changes. [Warns 05]
- Safety vs. Confidentiality

Design decisions that increase safety can come in direct conflict with confidentiality because safety decisions often require distribution of knowledge. (e.g. replication prevents loss of data at the cost of a higher probability of theft and unauthorized modification) [Schneier 03]

- Safety vs. Integrity

Design decisions that increase safety can come in direct conflict with integrity because safety decisions often require distribution of knowledge. [Schneier 03]

The architect must evaluate the tradeoffs between qualities when designing a dependable system. Analytically reasoning about composite qualities, i.e. qualities defined in terms of other qualities, that are measured on different scales, some of which are not quantitative, is not readily handled by existing techniques. Since security is not measured on a scale, a goal based scale will be used to support design reasoning. [McGregor 07] The goals on which these scales are based are called softgoals because there is no precise, objective definition of the goal or exact criteria for satisfying them. [Chung 00] A softgoal will not capture the level of detail found in performance models based on queuing theory but it will provide qualitative “indicators” that guide the architect. A softgoal is satisfactory for our purposes because qualitative reasoning techniques will allow us to make decisions about satisficing a softgoal but not optimizing it.

Qualitative Reasoning about Security

Qualitative reasoning [Iwasaki 97] provides a means of making decisions involving attributes that can not be expressed quantitatively. Qualitative techniques do assume some type of ordinal scale. The reasoning rules use two fundamental characteristics: a current position on an ordinal scale and an indication of whether the attribute is changing its value and if so in which direction along the scale. For example, the security attribute of a piece of software might be rated on an ordinal scale as “very” secure and that recent architectural changes are making the software “more” secure. Qualitative reasoning supports building models that represent these relationships between qualitative values. These models support inferences about how the values change over time and how they cause other values to change.

For qualitative reasoning about security, the model must consider the direction of change for each quality and the inequality relationship that exists among tactics in relation to how each tactic influences the qualities that comprise security. This matrix of relationships is the minimum required to evaluate the impact of the choice of tactics on the system being designed. For example, as shown in Table 1, consider two tactics that improve security [Steel 05] and a tactic that improves availability. It is difficult to assess the net effect of these three tactics on the degree to which the resulting system is secure since relative magnitudes of the “-“ effect of replication and the “+” effect of a validator can not be compared. (++ is strong positive satisficing, + is weak positive satisficing, -- is strong negative satisficing, - is weak negative satisficing) [Chung 95]

Table 1 - Effect of Tactic on Quality Attribute [Warns 05, Steel 05]

	Availability	Confidentiality	Integrity
Implementing a secure pipe	No change	++	++
Implement a intercepting validator	++	++	++
Replication of modules	++	--	--

The complexity in reasoning about these tactics is present partially because these attributes are not quantitative and partially because the measures are on different scales. It can be overwhelming to keep track of how each tactic influences each sub-quality of security and how each tactic relates to other

tactics. For this reason, we are developing a modeling technique to assist the architect in reasoning about security.

Satisficing security requirements: An Example

The qualities that are of most importance to a financial web-service are the following:

1. Confidentiality
2. Integrity
3. Reliability
4. Availability

How will the use of the two tactics, implementing secure pipes and introducing replication affect the overall quality goals of the system? As shown in figure 1, the two tactics influence confidentiality and integrity in different directions, but the confidentiality and integrity of the overall system would have decreased after the application of the two tactics. This is because there is an inequality relationship between the tactics as it was determined that “percentage of replicated elements” has a greater impact on confidentiality and integrity than “percentage of secured pipes.” (The application of these tactics results in an increase in the “percentage” of the replicated elements and secured pipes)

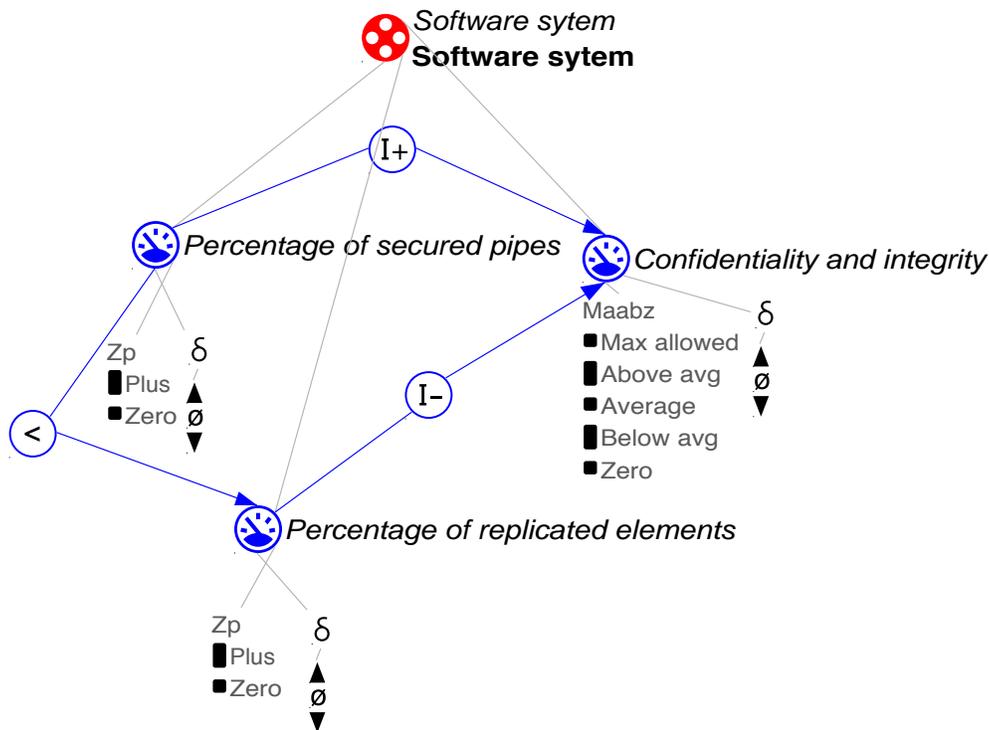


Figure 1

The inequality relationships between tactics are subject to change depending on the context of how they are applied and the architect has to decide on the relationship. The Garp3 tool used for qualitatively reasoning generates all possible cases if no inequality is specified. The power of the qualitative method comes when we combine these model fragments (such as the two shown in figure 1) that contain these inequalities to reason how the overall security of the system changes in response to the tactics that have been applied to the system. In our example, the financial web-service firm has confidentiality and integrity as its top priority and therefore it is not advisable to implement replication as an architectural tactic for the system.

To facilitate the application of qualitative reasoning to security a causal/qualitative model of security is needed. This model will describe how each tactic influences the sub-qualities of security (availability, confidentiality and integrity) and provide a knowledge base for qualitatively reasoning about security. The knowledge base will also contain the necessary data for reasoning about security in a broader context such as dependability. Capturing the causal/qualitative model will be very much like capturing an ontology but with relationships such as relative orders of magnitude and inequalities.

Conclusion

The qualitative method used to reason about security may, on the surface, seem too simplistic to be useful but research [Hastie 01] indicates that simple linear models are very accurate in supporting decision making and predictions. In this paper, we have presented how security architectural tactics can influence dependability and vice-versa. Certain security tactics can hinder the dependability goals of a system. We have also presented a reasoning method to choose architectural tactics that can help the architect achieve the quality goals of a system.

References

- [Avizienis 00] Algirdas Avizienis, Jean-Claude Laprie, and Brian Randell. Fundamental Concepts of Dependability.
- [Chung 95] Chung, L.K. Nixon, B. and Yu, E. "Using Non-Functional Requirements to Systematically Select Among Alternatives in Architectural Design", Proc., 1st Int. Workshop on Architectures for Software Systems, Seattle, April 24-28, 1995., pp. 31-43.
- [Chung 00] Chung, L.K. Nixon, B. and Yu, E. "Non-functional Requirements in Software Engineering", Kluwer Academic Publishers, 2000.
- [Hastie 01] Hastie, R. and Dawes, RM. "Rational Choice in an Uncertain World: The Psychology of Judgment and Decision Making", Sage Publications Inc , 2001 Thousand Oaks CA.
- [Iwasaki 97] Iwasaki, Y. Real-world applications of qualitative reasoning, Expert, IEEE [see also IEEE Intelligent Systems and Their Applications], 1997, pp. 16—21.
- [Kazman 00] Kazman, R. Klein, M. and Clements, P. ATAM: Method for Architecture Evaluation. CMU/SEI-2000-TR-004.
- [McGregor 07] John D. McGregor and Tacksoo Im. A Qualitative Approach to Dependability Engineering, Proceedings of Dahstuhl Seminar #07031, January 2007.
- [Schneier 03] Schneier, B. Beyond Fear: Thinking Sensibly About Security in an Uncertain World, Copernicus Books, 2003.
- [Steel 05] Steel, C. Nagappan, R. and Lai, R. Core Security Patterns: Best Practices and Strategies for J2EE(TM), Web Services, and Identity Management, Prentice Hall, 2005.
- [Warns 05] Warns, Timo Engineering Intrusion-Tolerant Software Systems. In: Dagstuhl Workshop, 22 - 25 May 2005, Dagstuhl, Germany.

Security in the Context of Dependability

Tacksoo Im
John D. McGregor
School of Computing
Clemson University
ORNL Presentation

1

Security in Context

- ❑ Security is not a directly measurable quantity.
- ❑ The level of security is usually described in terms of levels of availability, integrity and confidentiality.
- ❑ The level of security is used to help define the level of dependability.

2

Security in Software Architecture

- ❑ Security should be considered from the earliest point of the development process.
- ❑ Desired level of security is stated as a non-functional requirement.
- ❑ Trade-off between security and other non-functional qualities should be considered.
- ❑ How does an architectural design decision (tactic) influence security and other qualities?

3

What is an Architectural Tactic?

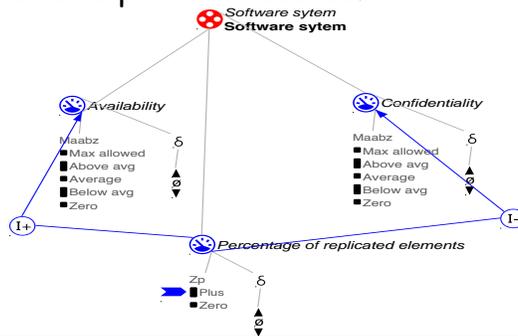
- ❑ An architectural tactic is a design decision that results in a desired change in the quality under consideration.
- ❑ Some examples are reducing the computational overhead and hiding information.
- ❑ Architectural tactics often influence one or more qualities.

4

Security Tradeoff 1

Availability vs. Confidentiality

- When availability is increased, confidentiality decreases because of prolonged exposure.
- The longer the system is available, the longer it is exposed to potential access.

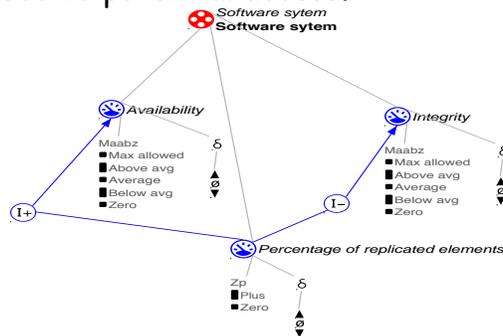


5

Security Tradeoff 2

Availability vs. Integrity

- When availability is increased, integrity decreases because of prolonged exposure.
- The longer the system is available, the longer it is exposed to potential access.

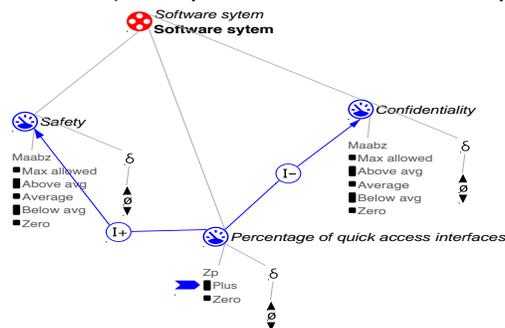


6

Security Tradeoff 3

□ Safety vs. Confidentiality

- In some cases, safety has to be relaxed to increase confidentiality and vice versa.
- Disabling confidentiality measures to ensure the data can be quickly accessed for safety.

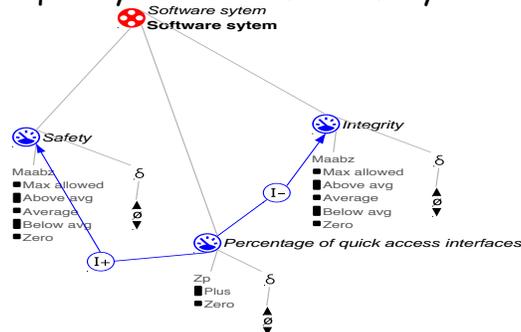


7

Security Tradeoff 4

□ Safety vs. Integrity

- In some cases, safety has to be relaxed to increase integrity and vice versa.
- Disabling integrity measures to ensure the data can be quickly accessed for safety.



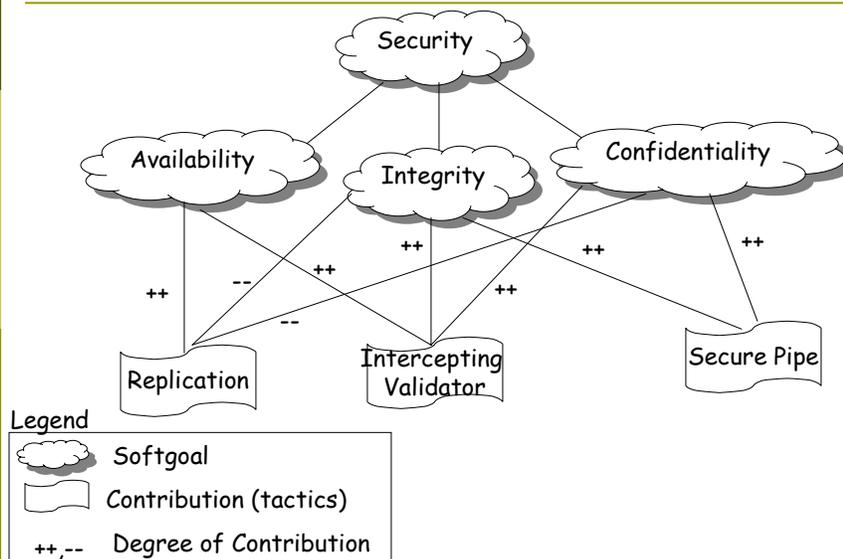
8

Security as a Non-functional Requirement

- ❑ Non-functional requirements such as Security can be seen as a *softgoal*.
- ❑ Softgoals do not have a clear definition and a criteria for satisfaction.
- ❑ A softgoal interdependency graph captures the interdependencies of softgoals.
- ❑ Softgoals are *satisfied*. (achieved within satisfactory boundaries)
- ❑ An architectural tactic can contribute positively, negatively, fully or partially to satisficing softgoals.

9

A Softgoal Interdependency Graph



10

What is Qualitative Reasoning?

- Qualitative Reasoning is reasoning with imprecise data.
- Often used to model tacit (implicit) knowledge.
- **Influences** model processes that cause changes within a model.
- **Proportionalities** propagate the effects of a process.
- **Model Fragments** describe the structure and behavior of the system in a general way.

11

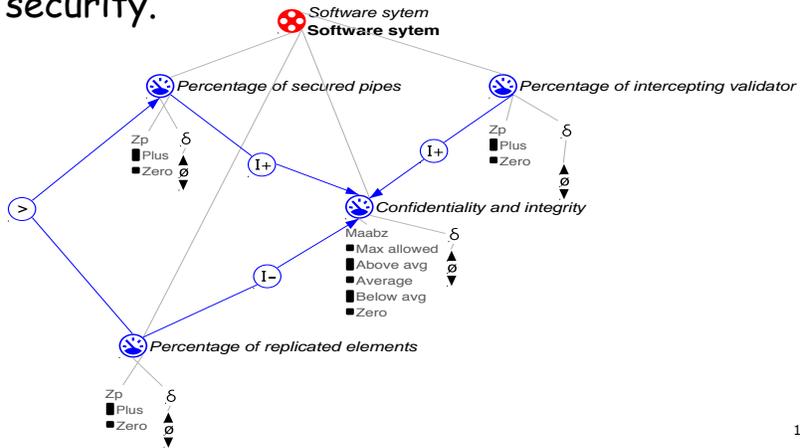
Qualitative Reasoning about Security

- Softgoals can be mapped to a qualitative scale. (i.e. Max, Exceeds goal, meets goal, does not meet goal, Min)
- QR can be used to determine if a softgoal is satisfied.
- Positive, negative, full and partial contributions to the softgoal can be seen as *influences*.

12

Qualitative Reasoning about Security : An Example

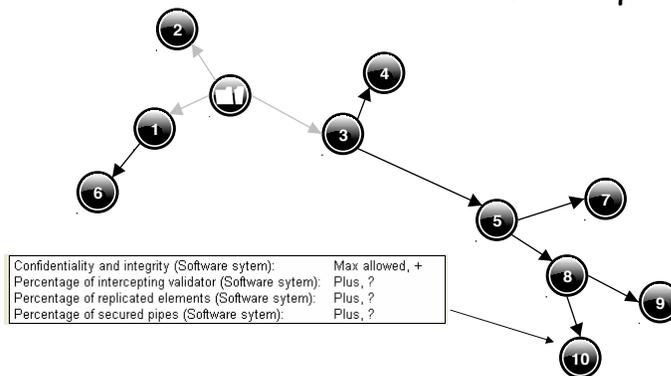
- Model Fragment from a QR model for security.



13

Qualitative Reasoning about Security : The Results

- Garp3 (a workbench for QR modeling) gives the result of the application of the tactics.
- The circles shows the state of the qualities.



14

Satisficing Security Requirements

- Combining Garp3 model fragments will help us reason about the result of applying a set of tactics.
- How does a change in security influence overall system dependability?

15

Conclusion

- Qualitative Reasoning can be used to find out the overall effects of an architectural tactic on software security.
- QR model of security can be a part of model that covers dependability.

16

Hardware/Software Security Support

R. R. Brooks, Associate Professor

S. T. Sander, Assistant professor

Holcombe Department of Electrical and Computer Engineering

Clemson University

Clemson, SC 29634-0915

Tel. 864-656-0920

Fax. 864-656-1347

email: rrb@acm.org

1. Introduction

It is necessary to combine cryptographic primitives, compiler optimizations and adaptive hardware to create a truly secure mobile computing environment. Here we consider embedded applications, but this approach can also create secure co-processors for desktop systems. FPGA hardware can use standard symmetric and public key cryptography approaches.

All chip input and output can be encrypted putting a strain on chip performance. Caches on the FPGA alleviate this strain. Data and program security in memory, peripherals, and communications will rely on the complexity guarantees of cryptography algorithms.

Security of the underlying hardware is based on two premises: (i) Fielded devices will be created and/or initialized in a secure facility. (ii) Devices will have physical safeguards against tampering and directly reading the internal state of the processor.

Covert channels, including monitoring resource consumption, are an important vulnerability. To counter attacks using covert channels use a secure instruction set approach, which masks resource usage when processing sensitive information. Extensions can include physically and temporally isolating processing on the chip minimizing information leakage. Physically isolated processes share no common memory.

An interpreter embedded on the chip can execute programs. The interpreter associates keys with users. Access rights and security policies are enforced using in-line reference monitoring. All programs are monitored during execution and terminated before an active policy can be violated.

System integrity can use the approach in [1] to maintain a trust hierarchy. The top layer of the hierarchy is an integrity monitor that uses secure hashing to guarantee that the level beneath it has not been corrupted. The level beneath it includes the interpreter, encryption primitives, and key management functions.

Hardware integrity is assumed. No mechanism will exist on the FPGA for modifying the integrity monitor. The integrity monitor verifies that the level beneath it has not been corrupted. Lower levels in the hierarchy guarantee that changes are made only by authorized entities, and that no corruption occurs after the fact. Trust in the rest of the system rests on the sanctity of the higher levels. Violations of trust result in the system restoring the initial state of the violator, terminating the process, or signaling the violation.

2. Detailed approach

This creates a secure computing environment by using hardware and software co-design. Consider a processor with an embedded interpreter executing arbitrary programs, using a set of cryptographic primitives. We consider the system secure, when the following are true:

- Basic infrastructure is not corrupted.
- The system's security policy is maintained.
- Confidential information is kept secret.
- Covert channels are removed.
- Data storage and communications are either encrypted or physically shielded.

This paper sketches a research agenda for creating this type of secure environment using adaptive hardware.

Table 1 provides initial implementation results for symmetric key applications on a small FPGA. Public key implementation in hardware is possible, but most public key algorithms rely on modular exponentiation. Modular exponentiation implementation in hardware is problematic. An alternative, consistent with our vision, is to use an FPGA containing standard processor sub-units. The public key

algorithms will execute in the standard processor. As we will explain, access to this standard processor can be controlled.

Table 1:

Virtex-II Pro XC2VP50	Frequency	Area (Slices)	Throughput	Percentage
Block Ciphers				
AES (128/128)	123.793 MHz	5539	1.585 Gbps	23.45443767
AES Encryptor	142.349 MHz	2661	1.822 Gbps	11.26778455
AES Decryptor	124.332 MHz	3364	1.591 Gbps	14.24457995
3DES	214.362 MHz	1070	807.009 Mbps	4.530826558
DES	219.394 MHz	441	825.954 Mbps	1.867378049
Secure Hash Functions				
SHA	97.914 MHz	1722	626.650 Mbps	7.291666667
MD5	58.261 MHz	1838	466.088 Mbps	7.782859079

In-line reference monitoring (IRM) will enforce security policies. Policies control issues like key management. The set of policies enforceable using IRM are strictly defined. Violations are detected before they occur, and prevented. The violating process can be stopped or restored to a safe state. The monitor is part of the interpreter embedded in the chip.

The processor will be hardened and all chip I/O encrypted. This lets us assume that the hardware retains integrity and the system's internal state is secret. All chip I/O is routed through the encryption primitives. Code signing authenticates the sources of information and identifies the keys used for decryption. A new cache structure is needed to minimize the impact on processor performance.

It should be possible to verify FPGA circuit consistency with security policies. Each hierarchy layer (Fig. 1) has:

- No access to configuration registers of higher layers
- Read-only access to its own configuration
- Full access to configuration registers of its subregions.

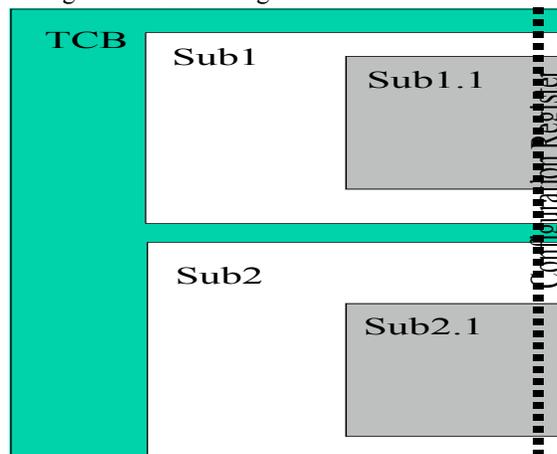


Fig. 1 Each hierarchy level can only access FPGA regions (incl. configuration registers) under its control. Only the integrity monitor has global read access.

The top layer of the trust hierarchy is an integrity monitor. These rules mean that no portion of the FPGA has write access to the integrity monitor. This monitor can verify its own integrity and signal an alarm should it be corrupted. The integrity monitor uses secure hashing to guarantee the integrity of level one components, including: interpreter, key management, and user entity roots. The equivalent of a secure bootstrap is provided by the security guarantees of the hardened processor and lack of support for modification of the integrity monitor. The circuit in Fig. 2 illustrates hardware enforcement of a policy restricting modification of an FPGA region. The hardware and integrity monitor together form the trusted computing base. Both are simple enough to allow exhaustive formal verification.

If the integrity monitor finds corruption at a lower layer, it can either stop the corrupted process or restore it to a trusted initial state. Corruption is thus detected, contained and controlled. It can only flow down the

hierarchy. This provides security guarantees for the system in spite of the fact that many desirable security issues, like the presence of viruses or implementation correctness, are often undecidable.

Note in the following discussion the goal is not to create new key management schemes, but to create adaptive hardware support for key management. Keys are stored in protected on-chip memory regions. Users usually access the clear text information they are authorized to use, with no direct access to their keys.

Keys can be stored indirectly using hardware signatures, generated using physical unclonable functions (PUF). A PUF is a circuit outputting a random bit sequence constant on a given chip, but differing from chip to chip. At key setup time, an XOR of the key value and the FPGA digital signature is computed and stored. The key can be recovered using the stored XOR value and device-specific hardware signature. This ensures security because 1) the PUF cannot be determined without destroying it and 2) the XOR value is useless outside the given device.

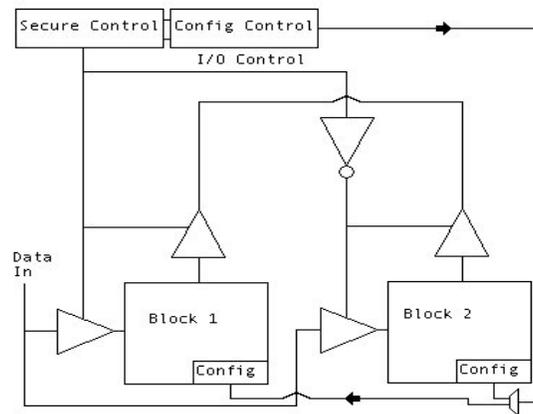


Fig. 2 Configuration memory can only be accessed by the secure control. Uni-directional bus and control multiplexer prevent communication or eavesdropping by other modules. Configuration memory readback is prevented.

Key generation can be external to the system or internal using pseudo-random number generators (like hash functions or block encryption ciphers). Only one key generation server is allowed.

Symmetric key management can use a binary tree structure. Key-encryption keys (KEK) are used to transmit session keys (SK). Each node in a tree of n participants is a leaf with an associated KEK. The root node of the tree provides a global KEK. Intermediate nodes have KEKs accessible to all leaves descended from the node. An SK can then be securely distributed to any group of participants without performing an excessive number of encryptions. In addition, each participant stores at most $\log n$ keys. This reduces the work needed to revoke a user's key.

For public key approaches, key management will use the X.509 standards. Public key approaches are difficult to implement in hardware. FPGA's with embedded processors are suited to PKI technology. We will also test hybrid key management schemes.

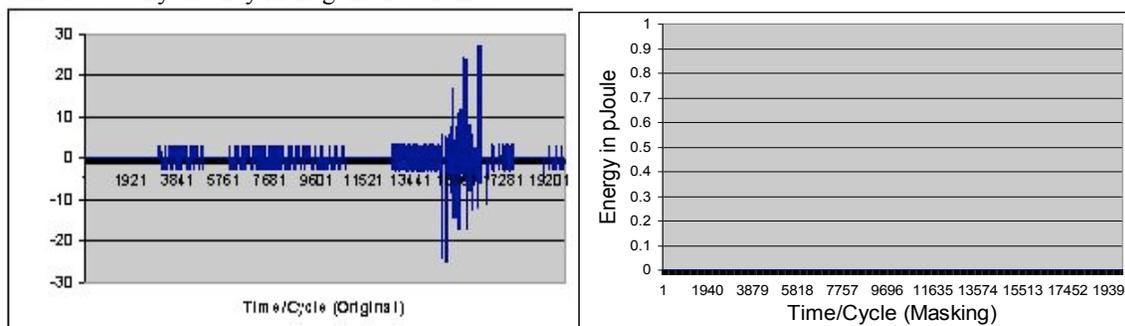


Fig. 3 DES encryption energy consumption difference with keys differing by one bit (top), using secure instructions (bottom) masks this energy consumption difference making it impossible to infer key contents.

To use a secure instruction set, variables and procedures are tagged with a security level. A separate set of hardware instructions are used for handling sensitive information. This approach successfully counters differential power analysis on smart cards. Using dual-rail logic for sensitive variables and masking the lower 6 bits of memory accesses, our DES implementation consumed the same power for any key and consumed 13% more energy than normal implementations (Fig. 3). Extensions are needed to mask covert channels other than power consumption. For example, an extension to the interpreter could mask memory access patterns when the FPGA accesses encrypted data external to the FPGA.

It is also possible to implement spatial and temporal isolation of processes in the FPGA. Spatial isolation restricts processes to a physical region (Fig. 4). If no communications channel exist, direct communications is impossible. Temporal isolation restricts operations from occurring concurrently. It is difficult for inactive processes to find covert communications channels.

Applications of these concepts for secret key algorithms include: (i) Spatial isolation operates on keys in blocks. They exist and are used only as fragments. The entire key never exists as a single entity at any given time. This allows parallelism since key operations are primarily XORs, which are bit-parallel (unlike addition). (ii) Symmetric key algorithms implicitly require temporal isolation, since the key is modified every round. Though the secure hash algorithm (SHA-1) does not have a key, initial constants used to operate on the data are stored independently as well.

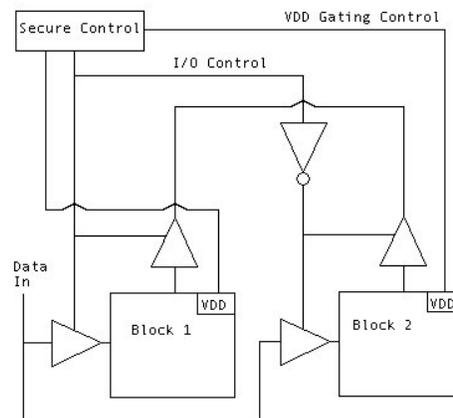


Fig. 4 Bus control is only accessible by a secure region. Blocks cannot control their own I/O. Similarly, the VDD control for each processing block is only accessible by the secure control. The control line's routing channel is accessible only to the secure control module. Bus control is mutually exclusive.

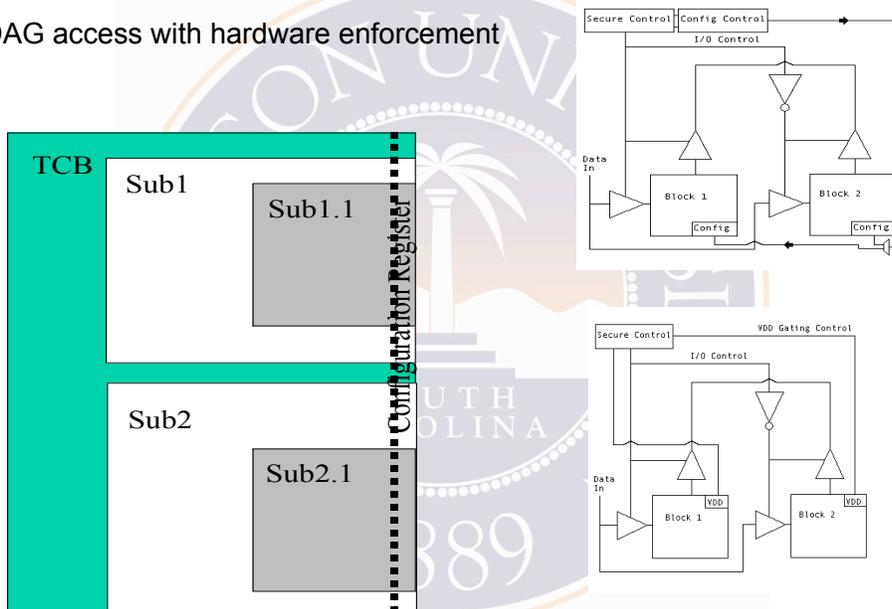
Integrated Hardware/Software Security Support

R. R. Brooks Associate Professor
 Sam T. Sander Assistant Professor
 Holcombe Department of Electrical and Computer Engineering
 Clemson University
 Clemson, SC 29634-0915
 Tel. 864-656-0920
 Fax. 864-656-5910
 email: rrb@acm.org

RRB/STS ORNL Workshop

FPGA security enforcement

DAG access with hardware enforcement



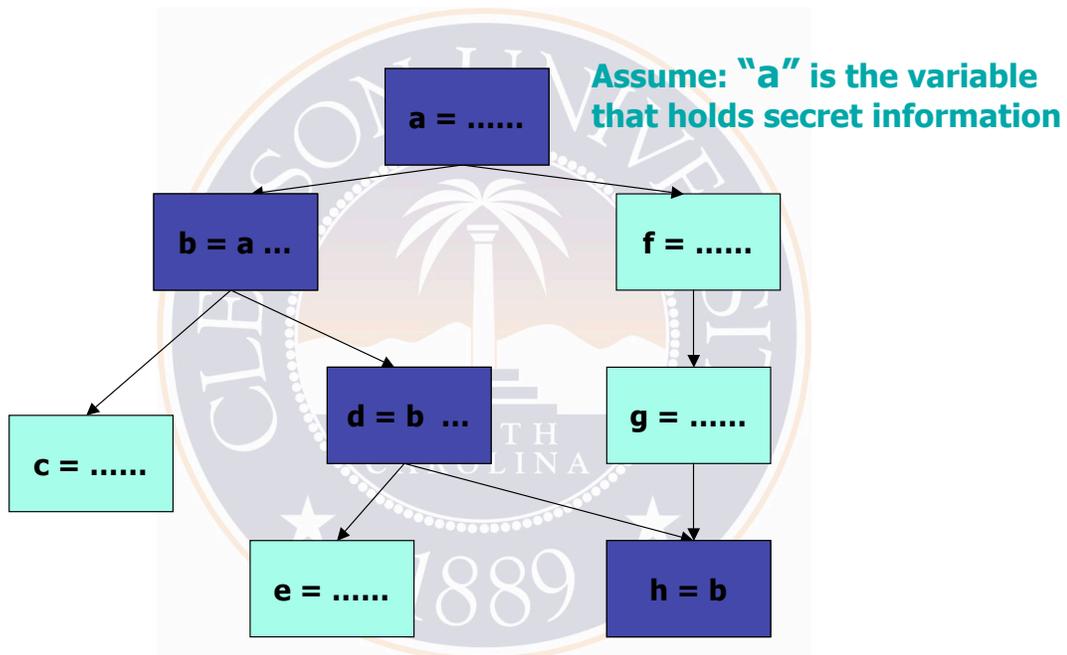
RRB/STS ORNL Workshop

Differential Power Analysis

- Using statistic method in order break the secret key
- Using several runs on several sample inputs, for instance 1000 sample inputs
- An attacker guesses a particular key and based on that key he can determines a theoretical value for one of the intermediate bits generated by the program

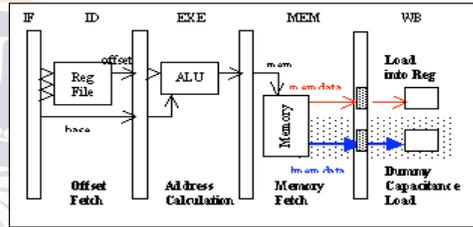
RRB/STS ORNL Workshop

Dataflow Analysis



RRB/STS ORNL Workshop

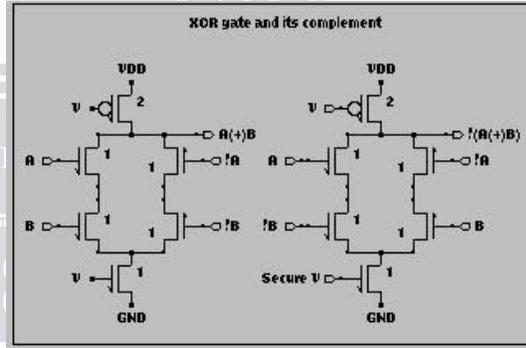
<p>Data Initial Permutation: (L0,R0) = PermuteIP(Data) Key Permutation: (O0,D0) ← PermuteK(Key) ← denotes insecure assignment</p>	<p>Data Initial Permutation: (L0,R0) = PermuteIP(Data) Key Permutation: (O0,D0) ← PermuteK(Key) ← denotes secure assignment</p>
<p>M Rounds Left Side Operation: Lm ← Rm-1 M Key Generation: Km ← Rotate(Cm-1,1) Dm ← Rotate(Dm-1,1) Em ← PermuteK2(Cm,Dm) Right Side Operation: Rm ← PermuteE(Rm-1) (Rm-1,K) ← S(E(R) ⊕ Km) Rm ← Lm-1 ⊕ (Rm-1,K)</p>	<p>M Rounds Left Side Operation: Lm ← Rm-1 M Key Generation: Km ← Rotate(Cm-1,1) Dm ← Rotate(Dm-1,1) Em ← PermuteK2(Cm,Dm) Right Side Operation: Rm ← PermuteE(Rm-1) (Rm-1,K) ← S(E(R) ⊕ Km) Rm ← Lm-1 ⊕ (Rm-1,K)</p>
<p>Output Inverse Permutation: Output = PermuteIP (R16,L16)</p>	<p>Output Inverse Permutation: Output = PermuteIP (R16,L16)</p>



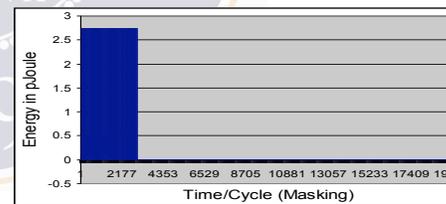
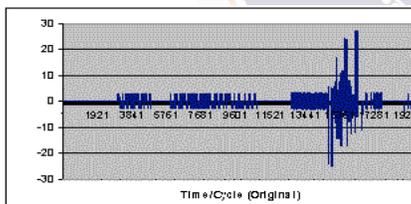
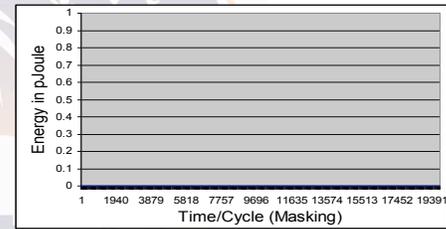
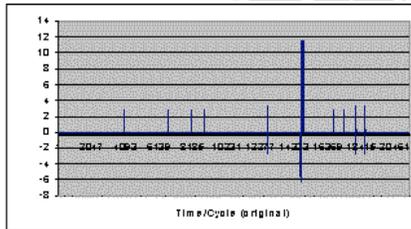
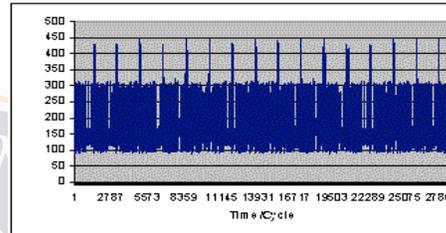
```

// Left Side Operation
for (i=0; i<32; i++)
    newL[i] = oldR[i]
.....
SL12:
.....
SL15:
lw $2,i
la $4,newL
addu $3,$2,$4
move $2,$3
lw $3,i
move $4,$3
sll $3,$4,2
la $4,oldR
addu $3,$3,$4
move $4,$3
lw $3,0($4)
sw $3,0($2)
SL14:
lw $3,i
addu $2,$3,1
move $3,$2
sw $3,i
j $L12
SL13:
.....
(a) Original Assembly Code

SL12:
.....
SL15:
lw $2,i
la $4,newL
addu $3,$2,$4
move $2,$3
lw $3,i
move $4,$3
sll $3,$4,2
la $4,oldR
addu $3,$3,$4
move $4,$3
slw $3,0($4)
saw $3,0($2)
SL14:
lw $3,i
addu $2,$3,1
move $3,$2
sw $3,i
j $L12
SL13:
.....
(b) Modified Assembly Code
    
```



RRB/STS ORNL Workshop



RRB/STS ORNL Workshop

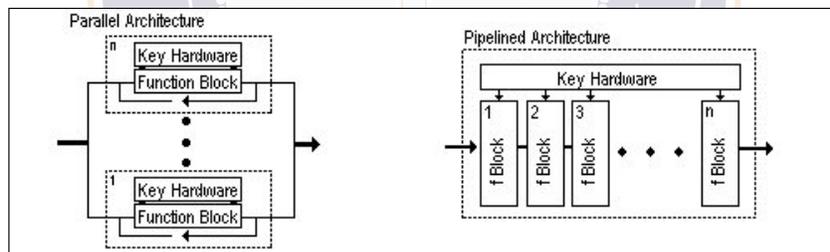
Conclusions

- Since we do not hide the energy behavior of all instructions, our approach consumes less energy overhead than other approaches
- Our approach has 83% less energy overhead than dual-rail logic

RRB/STS ORNL Workshop

Overview of Symmetric Encryption Architectures

- The dotted lines indicate the smallest subset of hardware capable of performing a single encryption.
- Features to note:
 - Parallel architectures have multiple key schedulers. This is both an advantage and a disadvantage.
 - Parallel architectures employ feedback routing.



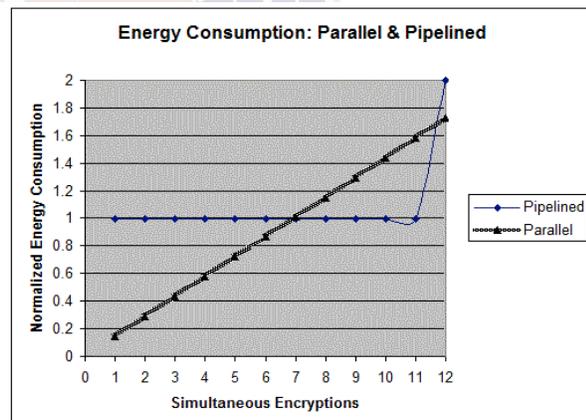
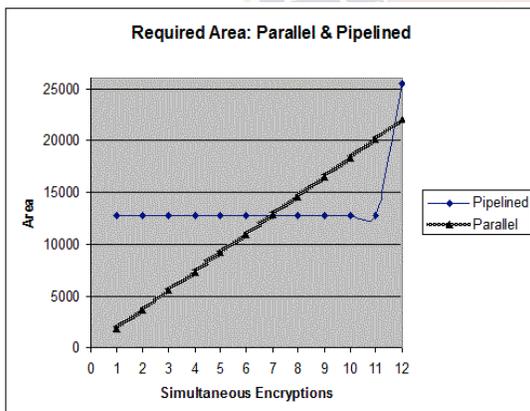
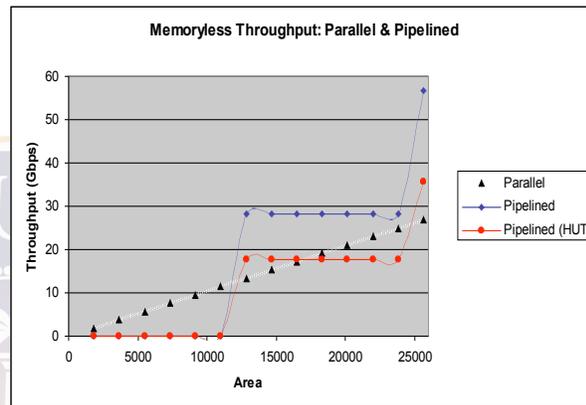
RRB/STS ORNL Workshop

Encryption Implementations – AES

- Industry & Published Results:
 - Helsinki University of Technology:
 - Virtex-II Pipelined: 17.8 Gbps
 - Helion Technology:
 - Spartan-3 Pipelined: 10.0 Gbps
 - Virtex-II Pro Pipelined: 16.0 Gbps
 - ASIC Pipelined: 25.0 Gbps
 - Single Spartan-3: 1.0 Gbps
 - Single Virtex-II Pro: 1.7 Gbps

Algorithm	Target Device	System Frequency	Area (Slices)	Throughput
Iterative Encryption	Virtex-II Pro 50	199.960 MHz	1736	2.327 Gbps
Parallel Processor	Virtex-II Pro 50	179.533 MHz	21997	22.98 Gbps
Pipelined Processor	Virtex-II Pro 50	221.141 MHz	12728	28.31 Gbps
Iterative Decryption	Virtex-II Pro 50	199.960 MHz	2795	2.327 Gbps
Iterative Encryption	Spartan-3 5000	157.873 MHz	1826	1.837 Gbps
Parallel Processor	Spartan-3 5000	130.647 MHz	22670	16.722 Gbps

Encryption Implementations – AES



FPGA Security Concerns

- Differential Power Analysis
 - Published Solutions
 - Selective dual-rail logic
 - Use dual-rail logic to create a uniform power profile for sensitive operations. Only minimal additional energy is consumed as dual-rail operations are not always employed.
 - Power supply noise injection
 - Obfuscates the power profile by adding random noise to the supply voltage within a specific range. Maintains functionality while making differential power analysis practically infeasible.
 - Both are good solutions

RRB/STS ORNL Workshop

FPGA Security Concerns

- Parallel Advantages
 - Irregular power profile
 - Variable number of simultaneous encryptions
 - Variable number of different keys
 - Variable number of active modules
 - Variable number of implemented modules
 - Dynamic key values
 - Within each encryption module, both the key data and the encryption data are changed dynamically
 - Differential power analysis becomes practically infeasible

RRB/STS ORNL Workshop

Conclusions

- A number of physical attacks exist:
 - Power analysis
 - Bit flipping
- Pure software solutions can not address them
- Pure hardware solutions can have prohibitive resource requirements (power, heat).
- Integrated compiler / instruction set support needed
- Hardware support essential for necessary throughput (ex. Symmetric encryption)
- Fixed hardware architecture can not adapt to varying system needs (ex. Number of processes requiring encryption.)
- Reconfigurable hardware architectures are attractive.
- Hierarchical verification possible.
- Isolating processes can take many forms (spatial, logical, temporal).

RRB/STS ORNL Workshop

Models of Models: Digital Forensics and Domain-Specific Languages

(Extended Abstract)

Daniel A. Ray* Phillip G. Bradford*

ABSTRACT

There are numerous and diverse digital forensics models for driving digital investigative processes. To encompass these diverse models we argue that there is need for two Domain-Specific Languages (DSLs) [5]: a static DSL and a dynamic DSL.

This paper motivates research towards building DSLs for proactive digital forensics investigations. In particular, we propose the creation of two descriptive languages for digital forensic models. We elaborate on both static and dynamic DSLs for static and dynamic forensic modeling, respectively. There may be a natural place to integrate these two DSLs as well.

1 Motivation

Good comprehensive models of digital investigation must provide a consistent and standardized framework that supports all stages and levels of an investigation. Digital investigation levels range from detailed technical IT and computer science to high-level procedural methods and best practices. Digital investigation stages range from site identification and preparation to analysis and presentation.

There are numerous digital forensics models, see for instance [9, 7, 4, 1].

The purpose of digital investigation models is to inform, shape, and standardize digital investigations. We are interested in how these models can be used in an integrated fashion to assist investigators. For example, a digital forensics DSL may have a library of the classical digital forensics models. The different models may be used and combined from this library by the DSL. We are particularly interested in having dynamic forensics models for proactive forensics [3, 2].

We differentiate two basic types of computer crimes: *computer assisted* crimes and *computer focused* crimes. In computer assisted crimes, the suspect uses computers for their basic functionality. In computer focused crimes, the suspect uses computers as a primary focus of their criminal activity.

We argue that digital forensics for computer assisted crimes can often be modeled using static models. Where the more dynamic nature of computer focused crimes requires a Turing-complete DSL. For example, an on-going investigation may have

*Department of Computer Science, The University of Alabama, Box 870290, Tuscaloosa, AL 35487-0290. DanielRay@cs.ua.edu , pgeb@cs.ua.edu

to dynamically model criminal behavior. Such dynamic models may use static models intermittently. A possibility would be to integrate such a system into Proactive Forensics systems [3, 6, 2].

DSLs are, "...languages tailored to a specific application domain." [5] DSLs provide a formal means for representing domain specific information that is not easily supported by General Purpose Languages such as C++, Java, Fortran, etcetera. DSLs are often developed because they can support domain-specific notation, constructs and abstractions [5].

DSLs can be Turing-complete and have well-defined execution semantics (i.e. Excel macros, Prolog, *etc.*) or less powerful (i.e. XML, UML, and HTML). In any case, these languages may be used as an "application generator" for a digital investigation. The application may drive the digital investigation and be honed specifically based on the initial circumstances of the investigation. We believe that the domain of digital forensics is one that would benefit from the creation of one or two DSLs at any of these levels, but particularly at the static as well as the Turing-complete level.

2 A Selection of Previous Work on Models of Digital Investigation

A selection of models for digital forensics investigation follows.

2.1 The DFRWS Framework Meta-Model

The DFRWS Framework was developed between 2001 and 2003 at the Digital Forensics Research Workshop (DFRWS)[9]. The framework introduces "Digital Investigation Action Classes." The framework's classes serve to categorize the activities of an investigation into groups.

The framework does not dictate what particular actions must be pursued. Instead, it provides a list of candidate techniques, some of which are required. The specifics of the framework must be largely redefined for each particular investigation.

The framework is represented by a table including columns for each activity class. Each row contains candidate techniques. These candidate techniques may be performed in pursuit of the goals of the associated action class. In addition, the specific goals and purposes of each action class are described in [8].

2.2 The Reith, Carr, and Gunsch Model

The model presented by Reith, Carr, and Gunsch [7] is very similar to the DFRWS Framework [9]. The model includes Preservation, Collection, Examination, and Presentation classes similarly defined as those of the DFRWS.

The model also adds supports for tool preparation and the dynamic formulation of investigative approaches. This model also loosely supports iterations of individual activity classes.

2.3 The Ciardhuain Model

The model suggested by Ciardhuain [4] is based on previous models but exhibits an augmented waterfall architecture. The model's activity classes are doubly linked so that the pursuit of work in one activity class can cause an iteration of some or all of the work in the previous activity classes.

The inclusion of structures known as information flows allows a deeper understanding of the source of evidentiary and other data. These flows must be defined on an organizational basis, but can apply to different investigations within the same organization.

2.4 The Beebe and Clark Model

The Beebe and Clark model [1] provides structure for activities through phases consisting of multiple sub-phases rather than through activity groupings. Sub-phases are objective based rather than strictly activity based. The objective based sub-phases each fall into a particular phase and consist of a hierarchy of particular activities that are subordinate to the particular objective.

In addition, the Beebe and Clark model [1] includes “digital investigation principles” which over arch all the phases and sub-phases and affect how they are performed.

The ultimate goals for each sub-phase are represented as objectives rather than specific tasks. This is an important and unique difference from the task-based models we have discussed thus far. Objectives are goals that can be expected of activities that are similar in nature regardless of the specific case. Tasks are directly related to a specific case, type of crime, platform, etcetera.

2.5 Overview of Models

Each model's distinct characteristics give it inherent advantages and disadvantages. The DFRWS model is somewhat rigid and linear but is particularly suitable where necessary investigative activities are well-understood. The RCG model is more dynamic and supports a limited form of iteration between activity classes. The Ciardhuain model supports iterations of activities between activity classes and provides support for tracking information sources. The objective-based structure of the Beebe and Clark model is most flexible and provides the best support when very little is known about what investigative techniques will be required.

Each model, despite their differences, has quite a lot in common with other models. On one hand, it seems striking how these models are similar. On the other hand,

their differences are driven by varying needs in different investigative situations. The similarities and differences suggests a descriptive language for use in describing the domain for investigative forensics.

Stephenson [8] gives a lisp-like language for post-incident cause analysis. This language allows a structured description of the post-incident forensic analysis.

3 Creating a Digital Investigation Domain-Specific Language

Domain characteristics will change from investigation to investigation. These characteristics influence which model, and which specific activities should be used. Questions must be answered such as: is reuse of investigative techniques important? which investigative results will be highly scrutinized? Will investigators be required to explicitly handle many flows of information both inside and outside the investigation? Are the necessary activities clearly defined? etcetera.

According to Mernick, *et al.* [5], generating a DSL involves executing an analysis, design, and implementation phase. The problem domain is identified and domain knowledge is gained during the analysis phase. In the case of the digital forensics domain, much of this work has already been done. Indeed, we have already covered several of the most prominent domain models, and no doubt more exist. The single “domain model” that is needed for the purposes of designing the DSL will simply need to incorporate the approaches used by all the previous models.

Next, the design phase must be executed. Two considerations must be made during DSL design. Will the new DSL be related to any previous language or not and will the DSL design specification be formal [5]? In designing a new DSL, one may piggyback on a general purpose language, specialize or restrict an existing language, extend the features of a language or even do something completely new [5]. Ultimately, we are looking to achieve gains in expressiveness, ease of use, and enforcement, all the while incorporating domain knoweldge.

Lastly, the implementation phase must be executed in order to determine how the DSL will be implemented. Will the language be interpreted, compiled, pre-processed, etcetera [5].

4 Concluding Remarks and Further Directions

One possibility would be to base the creation of a descriptive level DSL on the Unified Modeling Language (UML). Action classes suggest classes or class hierarchies with individual activities as methods. The iterative nature of some of the models suggests the language must support iteration. The language must also support possible flows of work or computation. Also, the need to store data sources might be fulfilled

by special language supported data types or meta-data. Obviously, these and other considerations will need to be addressed.

The implementation phase must be executed in order to determine how the DSL will be implemented. Will the language be interpreted, compiled, pre-processed, etcetera [5]? It is very important such a DSL is extremely trustworthy.

Our goal is to enhance “best practices” heuristics for digital forensics with “programming patterns” for (dynamic or proactive) digital forensics.

References

- [1] N.L. Beebe and J.G. Clark. A hierarchical, objective-based framework for the digital investigations process. In *Proceedings of the 2005 Digital Forensics Research Workshop*, 2005.
- [2] P. G. Bradford and X. Hong. Proactive computer-system forensics for constrained devices. In *Cyber Security and Information Infrastructure Research Workshop*, pages 159–172. Oak Ridge National Lab, Oak Ridge Tennessee, 2006.
- [3] P.G. Bradford, M. Brown, J. Perdue, and B. Self. Towards proactive computer-system forensics. In P.K. Stemani et. al., editor, *International Conference on Information Technology Coding and Computing*, pages 648–652, Los Alamitos, 2004. IEEE Press.
- [4] S.O. Ciardhuain. An extended model of cybercrime investigations. *International Journal of Digital Evidence*, 4(1), 2004. <http://www.utica.edu/academic/institutes/ecii/ijde/articles.cfm?action=issue&id=9>.
- [5] M. Mernick, J. Heering, and A.M. Sloane. When and how to develop domain-specific languages. *ACM Computing Surveys*, 37(4), December 2005.
- [6] D. A. Ray and P. G. Bradford. An insider threat detection digital forensics system. In *Third WG 11.9 International Conference on Digital Forensics*. To Appear, 2007.
- [7] M. Reith, C. Carr, and G. Gunsch. An examination of digital forensics models. *International Journal of Digital Evidence*, 1(3), 2002. <http://www.utica.edu/academic/institutes/ecii/ijde/articles.cfm?action=issue&id=3>.
- [8] P. Stephenson. Modeling of post-incident root cause analysis. *International Journal of Digital Evidence*, 2(2), 2003. <http://www.utica.edu/academic/institutes/ecii/ijde/articles.cfm?action=issue&id=6>.
- [9] Digital Forensics Research Workshop. A road map for digital forensics research. Technical report, Digital Forensics Research Workshop, 2001. <http://www.dfrws.org/2001/dfrws-rm-final.pdf>.

Models of Models: Digital Forensics and Domain-Specific Languages

Daniel A. Ray and Phillip G. Bradford
The University of Alabama
Tuscaloosa, AL

DanielRay@cs.ua.edu, pgb@cs.ua.edu

Outline

- Summary
- Motivation
 - Process of Evidence Collection
 - Proactive Forensics
- Models for Digital Forensics
 - Different from Classical Forensics
 - Leverage Computer Science
- Domain Specific Languages
- Sequential Statistics & DM techniques
- Conclusions

Summary

- Modeling the investigative process
 - Different investigation processes for different incidents
 - Classical forensics: different tools and procedures for different incidents
 - Digital forensics: different tools and procedures for different incidents
 - “Smoking gun” is very different from classical forensics
- Final objective: make the criminal case obvious to a lay-person
 - Depends on the method and procedure of the model
 - A failure on evidence gathering may damage or destroy the case
 - Can we catch the smoking gun when it is fired, when the smoke starts and when the smoke is wafting through the room?

Motivation: Classical & Digital Forensics

- Computer Security is often *preventative*
 - Focus on preventative measures
 - IDS--anomaly detection may be proactive
- Classical Forensics is *reactive*
 - Post-mortem
- Digital forensics is *reactive*
 - A lot of focus on file recovery from disks
 - Digital Forensics has opportunity to be proactive
- Proactive Forensics!
 - Online Monitoring stakeholders...
 - Internal anomaly detection: easier to get lots of details on behavior and potential evidence locations

Motivation: Proactive Computer-System Forensics

- System structuring and augmentation for
 - Automated data discovery
 - Lead formation
 - Efficient data preservation
- Make these issues proactive
 - How?
- Challenges
 - System resources
 - Exposure
 - Double edged sword...

Motivation: Proactive Computer-System Forensics

- What data should we capture?
 - Different crimes may require different investigative procedures
 - Static: when and where illicit data was placed on a disk
 - Dynamic: what system states do we document when there is an intrusion?
 - What is being written to logs or disks? Which programs are being run? Where is the smoking-gun?
 - Depending on the nature of our online investigation, we may need to secure evidence in several different models

Motivation: Crime Types

- *Computer Assisted Crimes*
 - Computers provide basic help in criminal activity
- *Computer Enabled crimes*
 - Computers are a Primary focus on criminal activity
- **Focus:**
 - Dynamic: computer enabled crimes
 - Range from viruses to spam to sophisticated attacks
 - Static: Computer Assisted Crimes
 - Stolen data, spreadsheets to compute illicit gains, etc.
- **Proceeding backwards: What evidence are we looking for when we are analyzing a crime?**
 - Proactively look in the right places!
 - Ideally, as the crime is being committed

Motivation: Variations on Digital Equipment and Software

- **Mobility & wireless**
 - Cell phones, PDAs, Laptops, etc.
- **Enterprise Level Systems**
 - Database systems, dynamic Internet sites, large proprietary systems,
- **Distributed systems**
 - Virtual private networks, network file systems, user mobility, distributed computation, etc.

Gathering Statistics for Proactive Forensics

- Running sequential statistical procedures
 - What data to save?
 - The data we need may change as things progress
 - Proactive not reactive
 - How much data do we save?
 - How costly?
- Stephenson gives a post-incident DSL using a lisp-like language
- Leighland and Krings give a formalized model for digital forensics analysis

The DFRWS Model

<http://www.dfrws.org/2001/dfrws-rm-final.pdf>

IDENTIFICATION	PRESERVATION	COLLECTION	EXAMINATION	ANALYSIS	PRESENTATION
Event/Crime Detection	Case Management	Preservation	Preservation	Preservation	Documentation
Resolve Signature	Imaging Technologies	Approved Methods	Traceability	Traceability	Expert Testimony
Profile Detection	Chain of Custody	Approved Software	Validation Techniques	Statistical	Clarification
Anomalous Detection	Time Synchronization	Approved Hardware	Filtering Techniques	Protocols	Mission Impact Statement
Complaints		Legal Authority	Pattern Matching	Data Mining	Recommended Countermeasure
System Monitoring		Lossless Compression	Hidden Data Discovery	Timeline	Statistical Interpretation
Audit Analysis		Sampling	Hidden Data Extraction	Link	
		Data Reduction		Spatial	
		Recover Techniques			

Figure 1: DFRWS Digital Investigation Framework [29]

Ciardhuain Model by S. O. Ciardhuain

- Extends DRFWS Model by working on information flows
- Class-based model
 - Authorization activity
 - Planning activity
 - Notification activity
 - Hypothesis activity
 - etc.
- An augmented “waterfall model”
 - supports iterative backtracking between consecutive activities
 - models information flows
 - Feedback critique

Mobile Forensics Platform (MFP) by F. Adelstein

- To remotely perform early investigations into mobile incidents
- Analyze a live running (mobile) machine
- Maintains original evidence which is verifiable by a cryptographic hash
- Connect to same LAN as the suspect machine

DSLs

- DSLs are, “. . . languages tailored to a specific application domain” Mernik, Heering, and Sloane
- Most Digital Forensics Models
 - Have a good deal in common
 - Evidence verification and storage
 - Flow of investigation
- DSLs can be built by: piggybacking, extension, restriction and from scratch
- Pulling together data storage, data modeling and authentication-verification
 - Combining other DSLs: XML, UML, DB Blobs, etc.

DSLs

- May be fairly complex to build a single DSL
 - However, worth investigating
- Must be a very trusted language
 - Numerous cases may depend on the trust-level of the language
- Move from “best practices” to more formal “programming patterns for digital forensics”

Conclusions

- Digital forensics is complex
 - Digital Forensics Models are complex
 - Static and Dynamic
- There may be a need to automatically choose from a diversity of digital forensics models
 - A programming language

Automatic Generation of Certifiable Aerospace Communication Software

Johann Schumann and Ewen Denney
RIACS / NASA Ames, Moffett Field, CA 94035
{schumann|edenney}@email.arc.nasa.gov

The need for reliable, secure and effective methods for communication in the aerospace domain is becoming increasingly important. Communication between a spacecraft and the ground station is central to all space missions, and an optimal design and implementation of the communication subsystems is an important prerequisite for a successful mission, since control of the spacecraft and the effective downlink of mission or science data clearly depend on reliable communication. This is especially the case for deep-space missions, where bandwidth is at a premium.

Also, advanced techniques for air traffic control require digital communication both between the aircraft and the control tower and between multiple aircraft in order to enable a smooth and safe control of the aircraft in a dense national airspace.

Furthermore, heightened needs for operations security add substantial complexity to the communication system requirements. A malicious attack or a simple flaw in the code can put human life at risk or jeopardize the mission.

Although secure communication protocols are in wide use, history has shown that many errors and vulnerabilities do exist and have been actively exploited. Security flaws can be introduced (or fail to be detected) during all stages of the software development cycle and may include

Misunderstanding of protocol requirements: the wrong protocol may be used for a specific application, or specific requirements might be violated (e.g., the existence of a trusted key server).

Weak cryptography: often, cryptographic algorithms are used that are much weaker than originally intended. Thus, attackers can hack or reverse engineer the code to expose vulnerabilities. Sometimes, proprietary encoding schemes are much weaker than published and proven protocols and algorithms.

Coding errors are a major source of vulnerabilities. Most security warnings regarding software like the Windows OS or Internet browsers have been caused by implementation errors like buffer overflow, uninitialized variables, deadlocks, etc.

Errors in protocol optimization: optimizing a complex, layered protocol toward maximal performance can lead to hard-to-detect errors and security vulnerabilities.

Errors during testing and deployment: a bad or incomplete selection of test cases will not exhibit flaws in the protocol. Incorrect testing and deployment procedures can thus lead to serious problems.

It is our contention that reliable and secure communication software can best be developed with a unified approach throughout the entire software life-cycle. We have developed a set of tools that facilitate a unified end-to-end approach to the design, analysis, implementation, and certification of communication software. Our tools are based upon rigorous logical and mathematical foundations, and are capable of automatically generating high quality communication (protocol execution) software from a high-level model using certifiable program synthesis.

As a modeling framework, we use UML, in particular sequence diagrams or scenarios to specify the temporal behavior of the protocol as a sequence of messages between participants (communication partners, key servers, etc.). In order to formalize a deeper semantic content, we augment the sequence diagrams with formal logical annotations in OCL (UML's Object Constraint Language).

The code is automatically generated from this model, and undergoes an automatic, tamper-proof certification process that provides explicit guarantees about important reliability and security properties, as well as the absence of implementation and design errors. These properties include absence of buffer-overflow errors, guarantees for variable initialization and correct usage (i.e., all required data are packed/unpacked and transmitted in the right way), and the correct use of encryption algorithms, although we do not analyze the mathematical properties of such algorithms. Security authentication properties are expressed using the well-known BAN logic. Although this logic is relatively weak, it is amenable to automatic processing and, as our tools can produce readable proofs, allows protocol designers to quickly find and fix flaws in a protocol.

Automatic Generation of Certifiable Space Communication Software



Ewen Denney, RIACS / NASA Ames
Johann Schumann, RIACS / NASA Ames

Secure Communication — Overview

Security is a multi-headed beast...

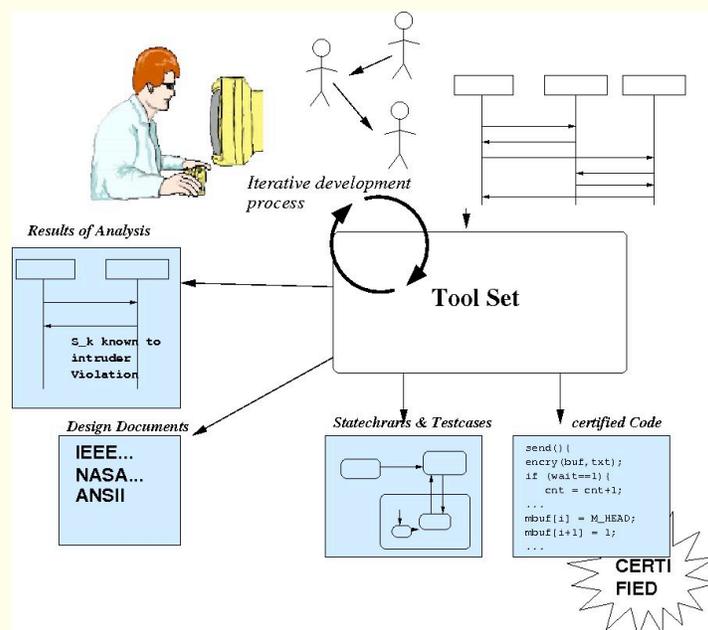
- Security of space communication protocols (software)
- Tool workflow
- The individual tools
- Conclusions

Introduction

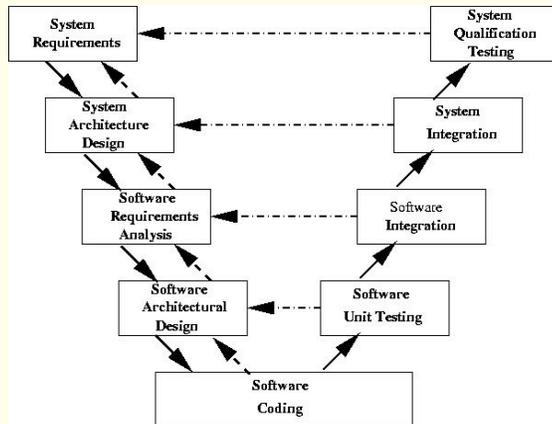
Software for secure communications can become insecure due to *flaws* during any phase of the software life cycle

- Design
 - wrong algorithm or protocol
 - wrong requirements
- Implementation
 - buffer overrun, uninitialized variable, ...
 - sleeper codes
- Verification and Validation
 - wrong tests
 - insufficient test coverage
- Deployment
 - wrong code (e.g., disabled crypto)
 - code tampering

Tool-supported Design and Analysis Process



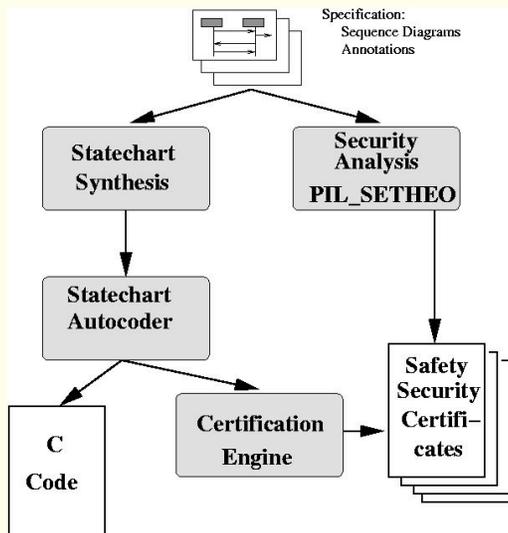
Software Process and the V-shape



Requirements:

- iterative process
- fast turnaround
- security analysis
- reliable, secure code
- support verification
- support validation
- support certification

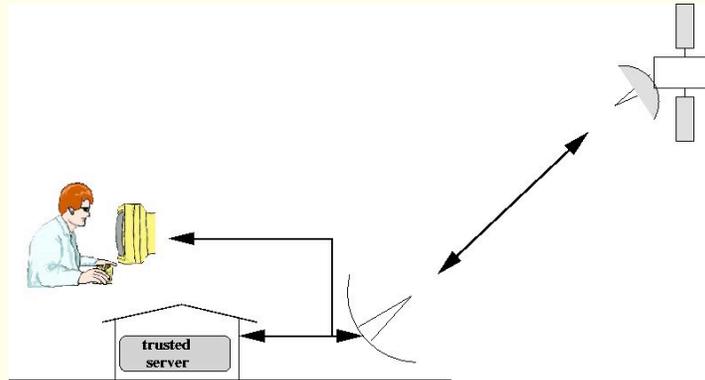
Our Tool Chain



Tools to support

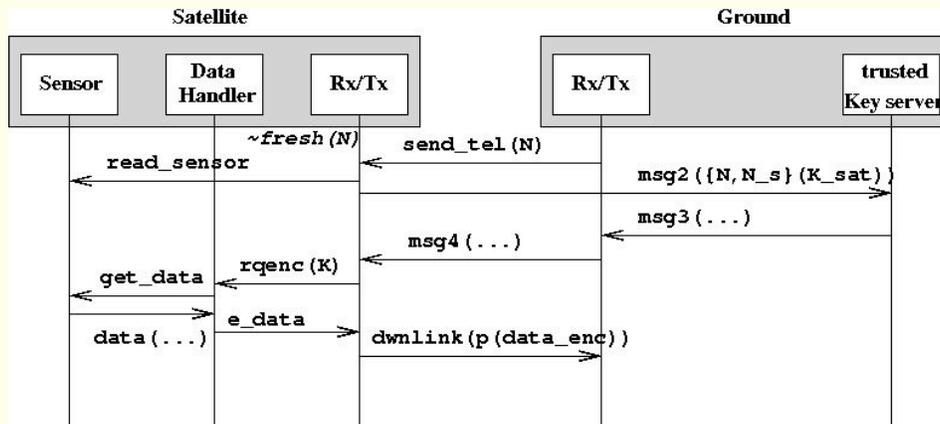
- high-level modeling
- automatic security analysis
- generation of designs
- autocoding
- safety-certification support
- Automatic generation of security cases and documentation

Example Protocol Specification



- Authentication of communication between satellite S and ground G .
- A key server K on the ground is used
- A Yahalom-style protocol shall be used

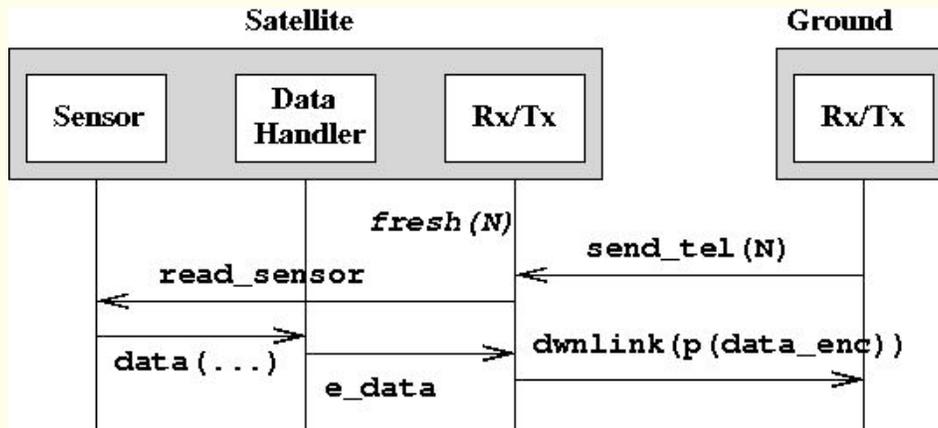
Specification of Protocol using Sequence Diagrams



- describes interaction between the various (sub-)systems
- annotations with actions and logical conditions

A number of different scenarios (sequence diagrams) comprise the specification of the protocol

Specification of Protocol using Sequence Diagrams



- another scenario
- satellite already has fresh key

A number of different scenarios (sequence diagrams) comprise the specification of the protocol

Statechart Synthesis: Annotations

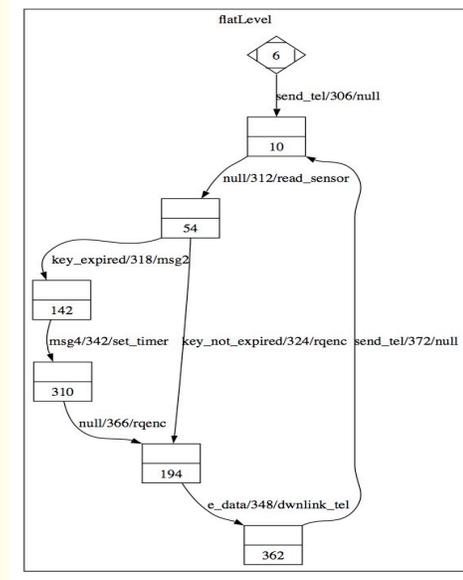
```
<features> <type> SAT_RXTX
  <attributes>
    key_is_valid : Boolean;
    key_recd     : Boolean;
  </attributes>
</invariants>
context SAT_RXTX:: msg4(...) : Void
  post: key_recd = true;

context SAT_RXTX:: rqenc(...) : Void
  pre: key_recd = true;
```

Annotations are used to specify conditions on the *state* of the subsystems and control a *correct and consistent* merge of the sequence diagrams.

Automatic Synthesis of Statecharts

- merge of SDs
- conflict detection
- loop detection
- introduction of hierarchy
- can be used for autocoding



Security Analysis

A formal analysis is necessary to ensure that the protocol is working correctly, i.e., it provides a secure session key and it does not compromise any information.

- We use BAN (Burrows, Abadi, Needham) logic
- Modal logic to express security properties
 - $S \models K \models \text{fresh } K_{gs}$ (satellite “believes” that the keys coming from the key server are valid (fresh))
 - $G, S \models K_{gs}$ (ground and satellite “believe” they have the right session key)
- BAN logic relatively weak (e.g., cannot model confidentiality) but well-used in design/modeling of security software

BAN representation of our protocol

A BAN logic specification of the protocol can be extracted from the sequence diagrams and annotations.

- 1 $G \Longrightarrow S \quad \text{send_tel}(N_g)$
- 2 $S \Longrightarrow K \quad \{N_g, N_s\}(K_{sk})$
- 3 $K \Longrightarrow G \quad \{K_{sg}, \#K_{sg}, N_g, N_s, S \sim N_g\}(K_gk), \{K_{sg}\}(K_{sk})$
- 4 $G \Longrightarrow S \quad \{K_{sg}\}(K_{sk}), \{N_s, K_{sg}, S \equiv \#K_{sg}\}(K_gk)$

Properties that must hold after execution of the protocol:

- $S \equiv K \equiv \text{fresh } K_{gs}$: S “believes” that the keys coming from the key server (K) are valid (fresh)
- $G, S \equiv K_{gs}$: ground and satellite “believe” they have the right session key
- obligations 3 . . . 12 omitted

Our tool PIL-SETHEO uses an automated theorem prover to *automatically* process all proof obligations and produce human-readable proofs

Protocol Analysis: Example Proof

Because of *message_meaning*

$$\vdash A \equiv B \sim C : - \vdash A \triangleleft \{C\}_D \wedge \vdash A \equiv B \stackrel{D}{\triangleleft} A. \quad (14)$$

Because of *lemma_from_task_2*

$$\vdash pB \equiv pA \stackrel{K_{ab}}{\triangleleft} pB. \quad (15)$$

Because of *sees_components*

$$\vdash A \triangleleft B : - \vdash A \triangleleft C \wedge B = \iota(C). \quad (16)$$

Because of *oneof* $A = \iota(\{B, C\}) : - A = \iota(C)$. Because of *oneof* $\{\{T_a, pA \stackrel{K_{ab}}{\triangleleft} pB\}_{K_{ab}} = \iota(\{\{T_a, pA \stackrel{K_{ab}}{\triangleleft} pB\}_{K_{ab}})\}$. Therefore $\{\{T_a, pA \stackrel{K_{ab}}{\triangleleft} pB\}_{K_{ab}} = \iota(\{\{\{T_a, pA \stackrel{K_{ab}}{\triangleleft} pB\}_{K_{ab}}, \{T_a, pA \stackrel{K_{ab}}{\triangleleft} pB\}_{K_{ab}}\})$. Hence by (16) and by (5) $\vdash pB \triangleleft \{\{T_a, pA \stackrel{K_{ab}}{\triangleleft} pB\}_{K_{ab}}$. Hence by (14) and by (15) $\vdash pB \equiv pA \sim \{\{T_a, pA \stackrel{K_{ab}}{\triangleleft} pB\}_{K_{ab}}$. Hence by (11) and by (13) $\vdash pB \equiv pA \equiv \{\{T_a, pA \stackrel{K_{ab}}{\triangleleft} pB\}_{K_{ab}}$. Hence by (8) and by (10) $\vdash pB \equiv pA \equiv \{pA \stackrel{K_{ab}}{\triangleleft} pB\}$. Hence by (7) $\vdash pB \equiv pA \equiv pA \stackrel{K_{ab}}{\triangleleft} pB$. Hence by (6) *querq*. Thus we have completed the proof of (4).

q.e.d.

Proof generated and typeset by PIL-SETHEO

Conclusions and Future Work

- We presented a loosely coupled set of tools that can support the design and implementation of secure communication protocols
- Tools provide assurance with respect to security properties and software safety
- Additional tools for security analysis, e.g., Model Checking, will increase level of assurance
- Combination with other modeling frameworks (e.g., UMLSec)
- Integration of correct optimization of Protocols
- Tight integration of tools into SW development tool chain and with COTS tools

LONG TERM VISION FOR IT SECURITY

Stop focusing on the system and start focusing on the data.

R. Scott Studham

The data.

Intranet



The vandal.



The trespasser.



The thief.



The thief.



The line between personal and business is increasingly blurred.

"You want to contact me in the evening, that is fine. We are going to use my home computer, IP phone, gmail, not yours"

Technology innovation is driven by consumer products

"I can do this at home, why not in the office?"

Business needs greater agility to survive.

"I can't wait for IT to deliver that"

"The seed of revolution is repression"

Woodrow T. Wilson

Digital Natives (younger workers) have new expectations

"You want me to do what? You must be joking, I'll just use my Groove / Flickr / MySpace / del.icio.us for collaboration. Next time IM me."

Consumer products cost less.

"How much? I could have 50 skype accounts for the cost of one of those old ISDN phones"

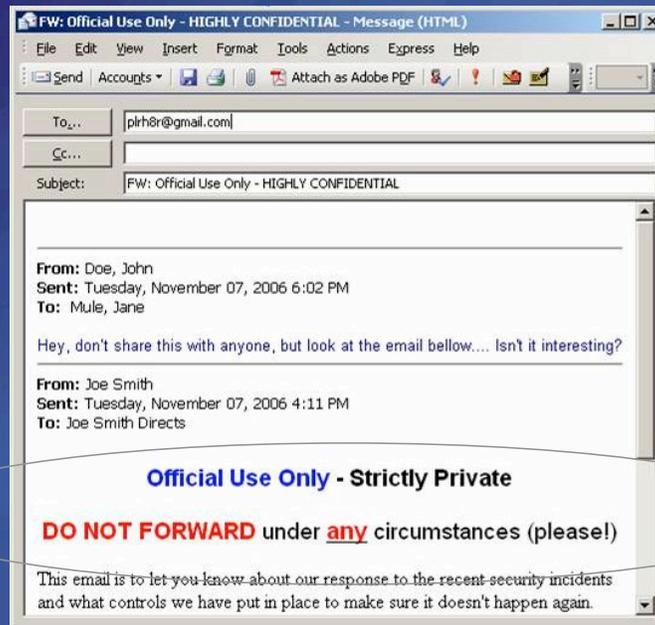
"By 2007, employee-owned notebook plans will be adopted by at least 20% of Type A companies (0.6 probability)." - Gartner

In order to lead the revolution instead of repressing it we must....



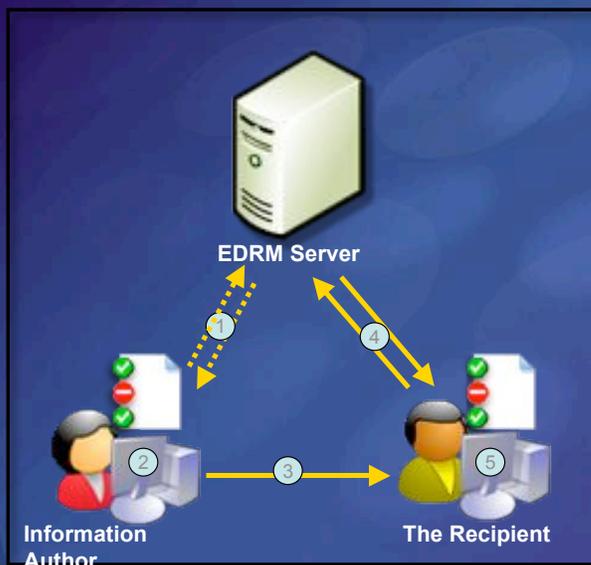
Focus on securing the data, not the device

Typical policy expression...



...lacks enforcement tools

Enterprise Digital Rights Management (EDRM) AKA: Document level security



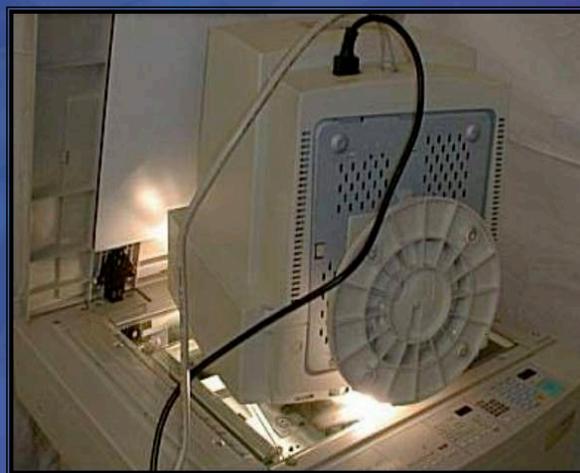
1. Author receives a client licensor certificate the first time they rights-protect information.
2. Author defines a set of usage rights and rules for their file; Application creates a "publishing license" and encrypts the file.
3. Author distributes file.
4. Recipient clicks file to open, the application calls to the EDRM server which validates the user and issues a "use license."
5. Application renders file and enforces rights.

EDRM: This is not the same as DRM (iTunes / RIAA)

Currently EDRM lacks adopted standards

- Applications must have EDRM built in in order to read content.
- Vendors EDRM solutions don't interoperate.
 - Microsoft Rights Management Server clients
 - Office 2003 (Word, PowerPoint, Excel, Outlook)
 - HTML using the RHTML SDK and IE Add-in
 - Via 3rd Party Extensions for: HTML, Visio, Acrobat, Blackberry, etc
 - Custom in-house applications using the RMS SDK
 - Adobe
 - EMC Corporation (Acquired Authentica 3/9/06)
 - Oracle (Acquired Stellant 11/2/06)
 - Liquid Machines (interoperates with Microsoft)

What EDRM does not protect from:



Wouldn't it be cool if the Cyber R&D community helped EDRM with:

- Standards
- Trusted clients
- Classification templates
- Auto classification of data sensitivity.
 - *“It will take three to five years before we begin to see the integration of automated tools within EDRM systems that are effective enough to meaningfully minimize the work of applying EDRM controls to existing documents (0.8 probability).” - Gartner*

TCIP: Trustworthy Cyber Infrastructure for the Power Grid

William H. Sanders
Donald Biggar Willett Professor of Engineering
Director, Information Trust Institute
Department of Electrical and Computer Engineering, and
Coordinated Science Laboratory
University of Illinois at Urbana-Champaign

Abstract

The Information Trust Institute is the home of the TCIP Center, a national center created in August 2005 to address the challenge of how to protect the nation's power grid. The National Science Foundation awarded \$7.5 million over five years to the project, which will be led by the University of Illinois ITI team and also involve researchers at Cornell University, Dartmouth College, and Washington State University. The Department of Energy and the Department of Homeland Security have pledged to join NSF in funding and managing the effort. The center will significantly improve the way the power grid cyber infrastructure is built, making it more secure, reliable and safe.

Our quality of life is dependent on the continuous functioning of our nation's electric power infrastructure, and the functioning of the power infrastructure is dependent on the health of an underlying computing and communication network infrastructure that is at serious risk from both malicious cyber attacks and accidental failures. Industry studies suggest that the risk of future cyber attacks on the electric power grid cyber-infrastructure is significant, and that such attacks, if successful, could have severe consequences. The August 14, 2003 blackout demonstrated how quickly the failure and/or misbehavior of individual components (in that case, partially caused by software failure) can spread across a large geographical area. Furthermore, the constraints of the power system IT infrastructure, which include changing relationships among participants, increasing data volume, and rapid response requirements, are similar to those faced by many other critical networked information systems, so solutions for the power system IT infrastructure will have applicability to cyber infrastructures for other critical systems.

TCIP is working to provide the fundamental science and technology needed to create an intelligent, adaptive power grid that can survive malicious adversaries, provide continuous delivery of power, and support dynamically varying trust requirements. We will do so by creating the necessary cyber building blocks and

architecture, and by creating validation technology to quantify the amount of trust provided by the proposed approach.

Personnel attached to the project include researchers from UIUC, Washington State University, Dartmouth College, and Cornell University. *Industry partners include* ABB, Ameren, AREVA, California ISO, Cisco Systems, Entergy, Exelon, GE, Honeywell, KEMA, Open Systems International, PJM Interconnection, PowerWorld Corporation, Siemens, and TVA.

Biography

William H. Sanders is a Professor in the Department of Electrical and Computer Engineering, Information Trust Institute, and the Coordinated Science Laboratory at the University of Illinois. He is the Director of the Information Trust Institute (ITI) at the University of Illinois. He is a Fellow of the IEEE and the ACM. He serves as the Vice-Chair of IFIP Working Group 10.4 on Dependable Computing. In addition, he serves on the editorial boards of IEEE Transactions on Reliability and Performance Evaluation, and is the Area Editor for Simulation and Modeling of Computer Systems for the ACM Transactions on Modeling and Computer Simulation. He is a past Chair of the IEEE Technical Committee on Fault-Tolerant Computing.

Dr. Sanders's research interests include performance/dependability evaluation, dependable computing, and reliable distributed systems. He has published more than 150 technical papers in these areas. He is a co-developer of three tools for assessing the performability of systems represented as stochastic activity networks: METASAN, UltraSAN, and Möbius. Möbius and UltraSAN have been distributed widely to industry and academia; more than 300 licenses for the tools have been issued to universities, companies, and NASA for evaluating the performance, dependability, security, and performability of a variety of systems. He is also a co-developer of the Loki distributed system fault injector and the AQuA/ITUA middlewares for providing dependability/security to distributed and networked applications.

TCIP: Trustworthy Cyber Infrastructure for Power

William H. Sanders

University of Illinois at Urbana-Champaign

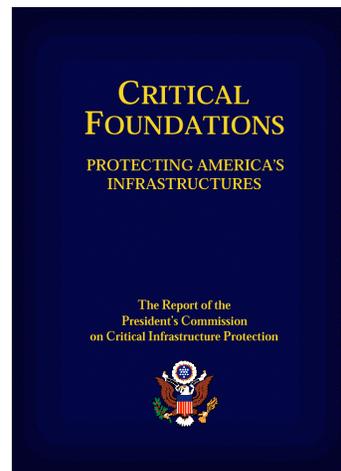
for the TCIP Project Team



The Nation's Power Cyber Infrastructure is at Risk

1997:

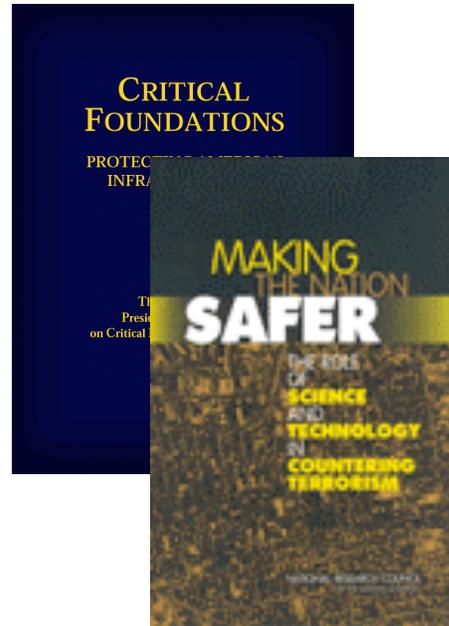
- “The widespread and increasing use of **SCADA** systems for control of energy systems provides increasing ability to **cause serious damage and disruption by cyber means**”



The Nation's Power Cyber Infrastructure is at Risk

2002:

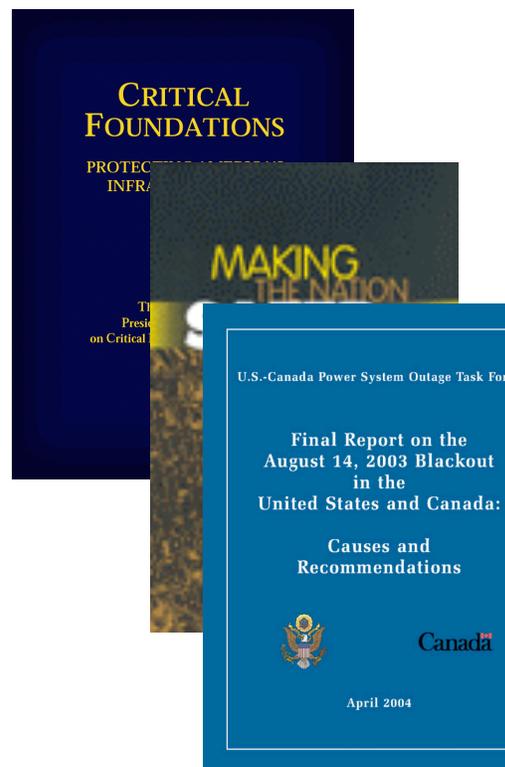
- “Simultaneous attacks on a few critical components of the grid could result in a widespread and extended blackout.”
- “Conceivably, they could also cause the grid to collapse, with cascading failures in equipment far from the attacks, leading to an even larger, longer-term blackout.”



The Nation's Power Cyber Infrastructure is at Risk

2004:

- “A failure in a software program not linked to malicious activity may have significantly contributed to the power outage.”
- “Control and Data Acquisition (SCADA) networks to other systems introduced vulnerabilities.”
- “In some cases, Control Area (CA) and Reliability Coordinator (RC) visibility into the operations of surrounding areas was lacking.”



NERC is Concerned about such Attacks

Critical Infrastructure Protection - Microsoft Internet Explorer

Address: <http://www.nerc.com/cip.html>

NORTH AMERICAN ELECTRIC RELIABILITY COUNCIL
[home](#) | [regions](#) | [committees](#) | [meetings](#) | [search](#) | [site map](#) | [contact us](#)

Critical Infrastructure Protection

[CIP Message System](#)

Industry-Government Interface
 NERC plays a major role in protecting the electric system by serving as the focal point for coordinating information exchange on critical infrastructure issues between the electricity industry and the federal government. Through NERC, government and industry work together to protect the electricity infrastructure from physical and cyber attacks. This coordination ensures that the industry is able to speak with one voice and take action in a consistent and effective manner.

The [U.S. Department of Energy \(DOE\)](#) designated NERC as the electricity sector coordinator for critical infrastructure protection. NERC serves as the Information Sharing and Analysis Center for the electricity sector. NERC also works closely with the [Department of Homeland Security \(DHS\)](#) and the [Public Safety and Emergency Preparedness Canada \(PSEPC\)](#) to ensure that the critical infrastructure protection functions so vital to the industry are fully integrated and coordinated with the department.

Electricity Sector Threat Advisory Levels	
Physical ELEVATED	Cyber ELEVATED
Significant Risk of Terrorist Attacks	Significant Risk of Terrorist Attacks

Electricity Sector Information Sharing and Analysis Center (ESISAC)
 As the designated ESISAC, NERC gathers, disseminates and interprets security-related information between industry and the government and with all the sector entities. The ESISAC website posts advisories, alerts, warnings and the current threat alert levels for the Homeland Security Advisory System, DOE, the Nuclear Regulatory Commission, and the electricity sector.



A Smart, Responsive, and Self-Healing Grid is Needed

“Building the Energy Internet,” The Economist, March 11, 2004

More and bigger blackouts lie ahead, unless today’s dumb electricity grid can be transformed into a smart, responsive and self-healing digital network ...

Economist.com SCIENCE **TECHNOLOGY QUARTERLY** SEARCH

Sunday May 15th 2005 [ED denotes premium content](#) | [Log in](#) | [Free registration](#) | [Help](#)

ENERGY

Building the energy internet
 Mar 11th 2004
 From The Economist print edition

[Printable page](#)
[E-mail this](#)

Energy: More and bigger blackouts lie ahead, unless today's dumb electricity grid can be transformed into a smart, responsive and self-healing digital network—in short, an “energy internet”

“TREES or terrorists, the power grid will go down again!” That chilling forecast comes not from some ill-informed gloom-monger or armchair pundit, but from Robert Schainker, a leading expert on the matter. He and his colleagues at the Electric Power Research Institute (EPRI), the official research arm of America’s power utilities, are convinced that the big grid failures of 2003—such as the one that plunged some 50m Americans and Canadians into darkness in August, and another a few weeks later that blacked out all of Italy—were not flukes. Rather, they and other experts argue, they are harbingers of worse to come.

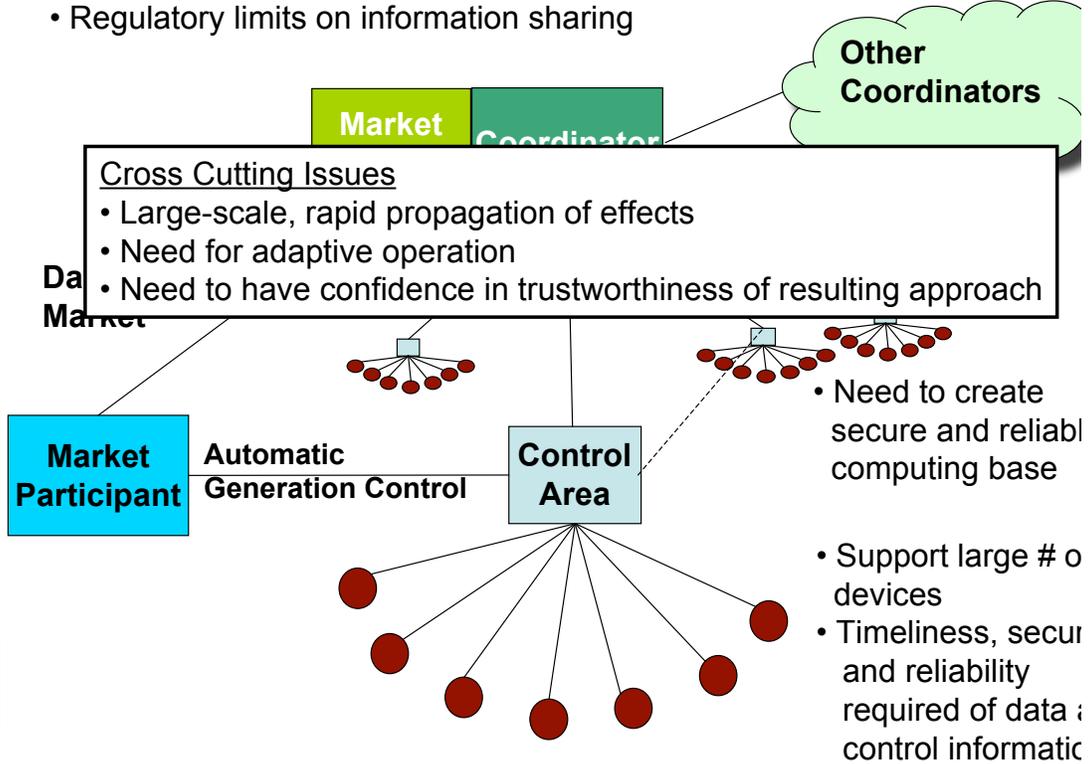
[DETAIL](#)
 Click to enlarge



www.economist.com/displaystory.cfm?story_id=2476988

Next-Generation Power Grid Cyber Infrastructure Challenge

- Multiparty interactions with partial & changing trust requirements
- Regulatory limits on information sharing



TCIP Vision and Strategy

- Provide the fundamental science and technology to create *the cyber infrastructure for an adaptive, available and secure power grid* which
 - survives malicious adversaries and accidental failures
 - provides continuous delivery of power
 - supports dynamically varying trust requirements.
- By:
 - Creating the cyber building blocks and architecture
 - Creating simulation- and experimental testbeds to quantify the amount of trust provided by proposed approach

TCIP: Trustworthy Cyber Infrastructure for Power

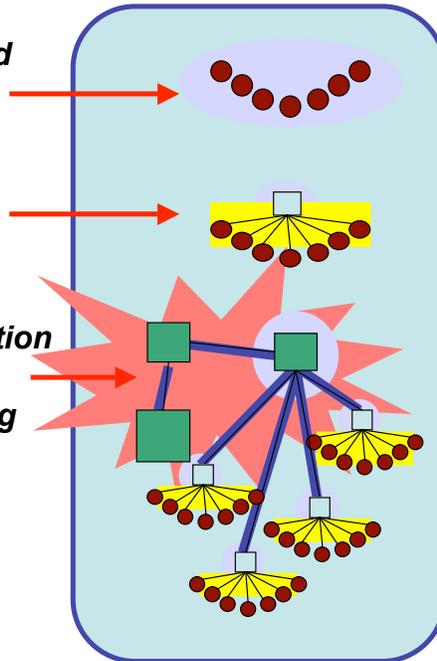
Address technical challenges motivated by power grid problems in

By developing

Ubiquitous exposed infrastructure

Real-time data monitoring and control

Wide area information coordination and information sharing



Secure and Reliable Computing Base

Trustworthy Communication & Control Protocols

Quantitative & Qualitative Evaluation

Education

tcip.iti.uiuc.edu



Technical Approach & Challenge

- 1. Secure and Reliable Computing Base:** Make low-level devices and their communications trustworthy. Challenges:
 - Sheer number of devices to be secured
 - Cost of securing them
 - Performance impacts of security on the devices' functionality
- 2. Communication and Control Protocols (1):** Efficient, timely and secure measurement and aggregation mechanisms for edge device data.
 - Challenge: devising and implementing adaptable policies and mechanisms for trading off performance and security during
 - Normal conditions
 - Cyber-attacks
 - Power emergencies



3. Communication & Control Protocols (2):

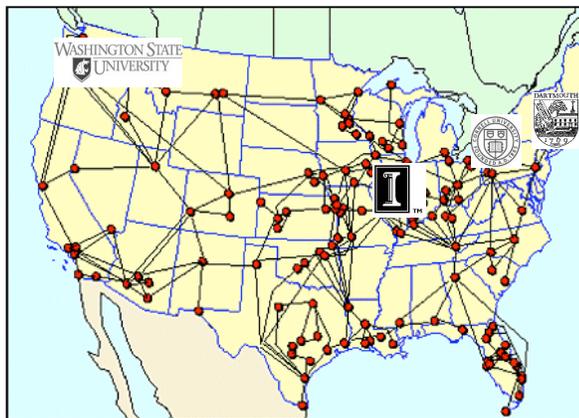
- Mechanisms for scalable inter-domain authorization
- Fundamental principles for security in emergency situations.
- Approaches
 - Dynamic negotiation under normal, attack and emergency conditions
 - Mechanisms to exploit the trusted computing base.

4. Quantitative & Qualitative Evaluation: Validate the TCIP designs and implementations produced in the other areas.

- create security metrics, multi-scale abstractions and attack models
- emulation technology to allow quantitative analysis of real power grid scenarios.



- **Secure & Reliable Base**
 - Gross, Gunter, Iyer, Kalbarczyk, Sauer, and Smith
- **Trustworthy Communication & Control Protocols**
 - Bakken, Bose, Courtney, Fleury, Hauser, Khurana, Minami, Nahrstedt, Sanders, Scaglione, Welch, Winslett
- **Quantitative & Qualitative Evaluation**
 - Anderson, Campbell, Nicol, Overbye, Ranganathan, Thomas, Wang, Zimmerman
- **Education**
 - Kalbarczyk, Overbye, Reese, Sebestik, Tracy



- **Partner Institutions**
 - Cornell
 - Dartmouth
 - University of Illinois
 - Washington State University



TCIP Graduate and Undergraduate Research

Graduate Students:

- Stian Abelsen (WSU)
- Angel Aquino-Lugo (UIUC)
- John Kwang-Hyun Baek* (Dartmouth)
- Scott Bai (UIUC)
- Nihal D’Cunha* (Dartmouth)
- Matt Davis (UIUC)
- Reza Farivar (UIUC)
- Chris Grier (UIUC)
- Joel Helkey (WSU)
- Alex Iliev* (Dartmouth)
- Sundeep Reddy Katasani (UIUC)
- Shruti Kirti (Cornell)
- Peter Klemperer (UIUC)
- Jim Kusznir (WSU)
- Adam Lee* (UIUC)
- Michael LeMay* (UIUC)
- Sunil Murthuswamy (WSU)
- Suvda Myagmar (UIUC)
- Hoang Nguyen (UIUC)
- Hamed Okhravi* (UIUC)
- Karthik Pattabiraman* (UIUC)
- Sankalp Singh* (UIUC)
- Erik Solum (WSU)
- Kim Swenson (WSU)
- Zeb Tate (UIUC)
- Patrick Tsang (Dartmouth)
- Erlend Viddal (WSU)
- Jianqing Zhang (UIUC)

Undergraduates:

- Katy Coles* (UIUC)
- Paul Dabrowski* (UIUC)
- Sanjam Garg (UIUC)
- Steve Hanna* (UIUC)
- Loren Hoffman (WSU)
- Allen G. Harvey, Jr.* (Dartmouth)
- Nathan Schubkegel (WSU)
- Evan Sparks* (Dartmouth)
- Erik Yeats* (WSU)

* Not funded by TCIP, but working on TCIP



Area 1 Approach

- **Focus:** Move from *perimeter security* to *platform security* in the power grid cyber infrastructure
- **Focus:** Secure power *infrastructure by ensuring* security of infrastructure *applications*
 - Derive security *requirements* from *application logic*
 - Derive *hybrid solutions* and *constraints* from application context
- **Project Areas:**
 - Build *new types of platforms* to achieve specific security goals for power applications
 - Make these hardened platforms *reconfigurable and customizable*, so one platform secures multiple power applications
 - Integrate hardened platforms into *comprehensive security architectures* for power grid scenarios



Year 1 Accomplishments / Research Direction

- **Hardening platforms:**
 - Demonstration of automatic tool to secure **high-stakes ISO computation** against dedicated insiders with physical access
 - Design and initial prototype of fast, novel crypto for **control centers and substations**
 - Design and prototype of processor modules:
- **Reconfigurable hardening**
 - Customize and implement, into an FPGA, Illinois Reliability and Security Engine (RSE) for **substations and control center** applications of the power grid infrastructure
 - Incorporation of attack detectors and error detectors within RSE
 - Methodology and associated tools for generation of application-specific assertions for runtime detection of malicious and accidental errors in **SCADA applications**
- **Application Integration**
 - Applied *Trusted Computing (TC)* and *virtualization* technologies to develop an **attested meter**
 - Analyzed security architecture requirements for **relays** in substations understand prospects for individually secured IEDs that can meet timing requirements



Trustworthy Communication & Control Protocols

The past

- Un-secure communication
- Slow communication links
- Lack of inclusion of networking and computing standard technologies

Trends

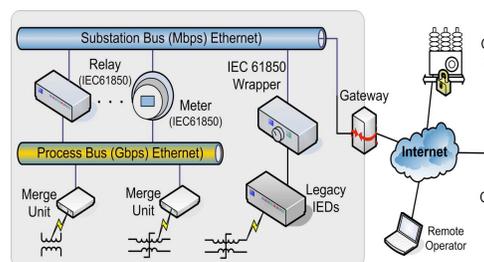
- Data collection at control areas
- High-speed wide area communication and computation solutions available (optical/SONET, multi-core devices, Linux)
- Standard wireless network technologies available
 - 802.11, 802.15, 802.16, Bluetooth
- IP-based protocol solutions available

Challenges

- End-to-end real-time, security, reliability, and QoS guarantees

Approach

- Provision of real-time and reliable monitoring, detection, alert, and control solutions in case of perturbations, vulnerabilities and attacks
- Self-adaptation to new security needs due to long-lifetime installed base (RTUs)
- Handling of adversarial threats to end devices (IEDs), control centers, ISOs, and communication links among them



Communication & Control Protocols Year 1 Accomplishments / Research Direction

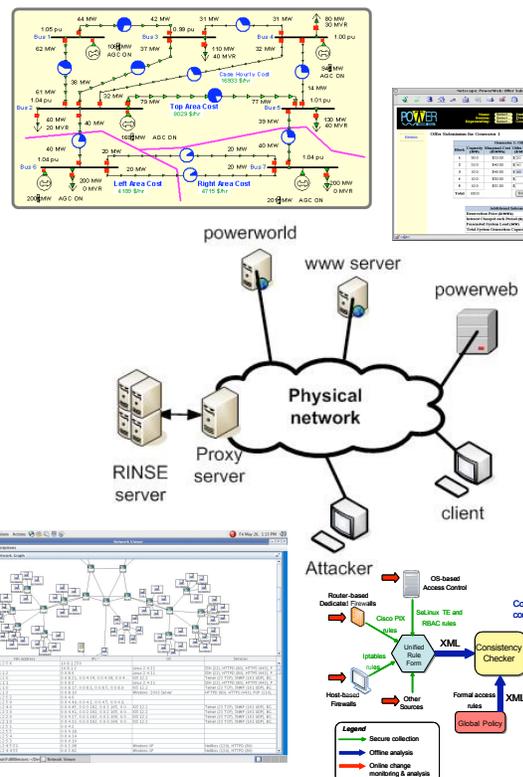
- Evaluated SCADA architectures and protocols for data transmission and aggregation (IEC 61850)
- Identified security threats and attacks in SCADA networks
- Explored mathematical models for QoS/data/alarm aggregations
- Analyzed requirements for generalized trust in pub/sub systems
- Achieved rigorous reasoning about trust negotiation
- **Designed Architectural Innovations**
 - Exploration of selected aggregation functions and algorithms over wireless network technologies
 - Initial design of alert and attack containment to limit spread of unwanted updates
 - Deployment of Real-Time QoS mechanisms in standard IP-based network technologies for QoS-aware dissemination of TCIP information
 - Development of trust management for TCIP components
 - Design of Credentialing for Emergencies at ISO level



Quantitative & Qualitative Evaluation

Approach:

- Developing tools and methodologies for evaluating and validating next-generation power grid designs
- Developing tools and methodologies for evaluating existing system configurations with respect to best practice recommendations and global policies
- Studying the sensitivity of the power grid infrastructure to various kinds of cyber attacks



Evaluation Year 1 Accomplishments / Research Directives

Simulation

- Emulation, transparent integration of IP devices {project,external} servers, routers, clients
- Modbus speaking simulators of power grid, and SCADA control center
- Algorithms for high speed virtual background network traffic
- Cyber-attack models (algorithms/optimizations + implementation)
 - Random scanning worms, flash-worms, packet reflection, packet redirection

Intruder client

- New man-in-middle code attack on Modbus timing
- Database of co-opted traffic

Power Markets

- Experimental design + technical support, co-opting auction information

System Evaluation

- Methodology for analyzing properties of system configuration vis a vis formalized interpretation of best practices
- Tool (APT) for analyzing firewall configurations vis a vis formalized global policies

Integration

- Network simulation/emulation operationally integrated with
 - Simulated power grid and SCADA
 - Simulated power auction server
 - Intruder client
- Conceptually integrated with system evaluation



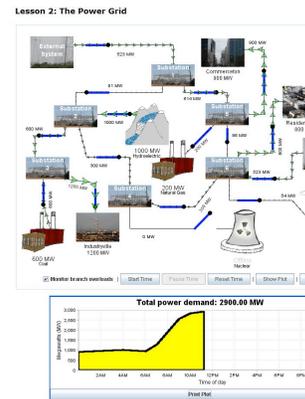
Education Goals

- Facilitate the integration of research, education and knowledge transfer by linking researchers, educators and students
- Connect with K-12 teachers and students
- Share higher education courses and instructional modules across disciplines involved in the project (CE, EE, CS)
- Provide research experiences to undergraduate and graduate students
- Develop hands-on laboratories and tools



Education : Year 1 Accomplishments / Research Directions

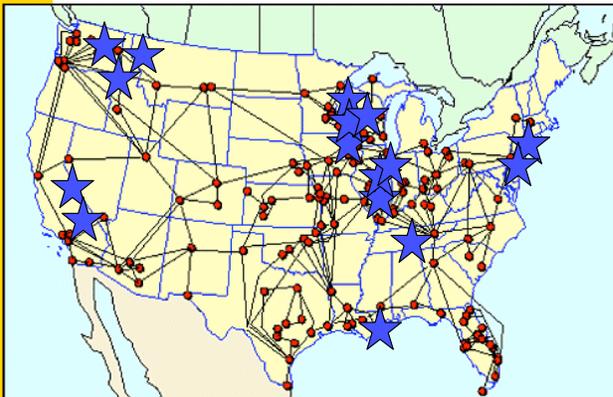
- TCIP Researchers, in partnership with math/science education specialists:
- Developed interactive and open-ended applets for middle-schoolers
- Produced printed activity materials and teacher guides coordinated with the applets
- Aligned lessons to content standards
- Started process of piloting and disseminating educational materials to students and educators in middle schools



5th grade student at Olympia North Elementary School using TCIP applet



Industrial Partnerships – Spanning Stakeholders



Electrical Power Generation, Delivery, and Management

- Ameren** – Major traditional utility in Mo. and IL
- Energy** – Major traditional utility in South
- Exelon** – Major traditional Utility – Midwest & East
- TVA** – Largest public power company

Technology Providers/Research

- ABB** – Industrial manufacturer and supplier
- Siemens** – Industrial manufacturer and supplier
- AREVA** – Major SW vendor for utility EMS systems
- Cisco Systems** – CIP Researchers
- Cyber Defense Agency** – Security Assessment
- EPRI** – Electric Power Research Institute
- GE Global Research** – Research in communication and computing requirements for US power grid
- Honeywell** – Industrial control system provider SCADA researcher
- KEMA** - Supports clients concerned with the supply and use of electrical power
- OSII** – Major SW vendor for utilities including SCADA and EMS systems
- PNNL** – National Lab doing SCADA research
- PowerWorld Corp** – System analysis and visualization tools
- Sandia National Lab** – SCADA research
- Schweitzer** – Industrial control system provider
- Starthis** – Automation Middleware
- CAISO** – Independent system operator for CA
- PJM** – Regional transmission organization (RTO) for 7 states and D.C.



- Comprehensive group of industrial advisors representing industries across the nation
- Industry seminars - ongoing
- Faculty visits and connections - ongoing
- Field trips for TCIP project team
 - MISO and Ameren IP during summer 2006
- Industry kickoff meeting – December 2005
- Industry workshop – December 2006
- Power systems infrastructure tutorial (in progress)
- Directory of industrial contacts (in progress)



Measuring Dependability as a Mean Failure Cost

Ali Mili,

College of Computing Science,

New Jersey Institute of Technology,

Newark NJ 07102-1982 mili@cis.njit.edu

April 2, 2007

1 Challenging the Mean Time to Failure

The Mean Time to Failure (MTTF) is a commonly accepted measure for system reliability. Some variations of it (MTTD, MTTE) have also been adopted to measure other dependability attributes [1]. In this short note we wish to submit a tentative challenge to this measure, propose an alternative, and discuss how the alternative can be deployed.

When we say that a system S has an MTTF M , we mean that the mean time to the failure of the system with respect to some implicit specification R is M . In doing so, we are usually making two implicit assumptions:

- *Independence with respect to subspecifications.* A complex specification is typically the aggregate of many individual requirements; the stakes attached to

meeting each requirement vary from one requirement to another. Yet the MTTF makes no distinction between requirement; failing any requirement counts as a failure.

- *Independence with respect to stakeholders.* Typically the operation of a system involves many stakeholders, who have different stakes in the system meeting any given requirement.

Taking these dependencies into account, we must now consider a substitute for the MTTF, which varies according to which specific requirement is violated, and according to which stakeholder we consider. We propose the concept of *Mean Failure Cost per Unit of Time* (MFC), which reflects how much a particular stakeholder stands to lose, on average, as a result of possible failure, per unit of time. Given a system whose MTTF is M , we can de-

decide whether to rely on this system or not by comparing M to the time T we expect to use the system: If the ratio $\frac{T}{M}$ is small enough, whence the likelihood of failure during the period of usage is small enough, then we can decide to use it. By contrast, given a system whose MFC is M , we can decide whether to rely on this system or not by comparing M to the mean benefit (B) that we gain from using this system per unit of time.

2 Quantification Infrastructure

To estimate the MTTF of a system, we only need to model its probability of failure with respect to its specification, R . By contrast, to estimate the MFC we need the following information:

- The probability of the system to meet various sub-specifications of R .
- A matrix that shows, for each relevant stakeholder, the associated failure cost with respect to each sub-specification.

What makes the estimation of MFC difficult is that the decomposition of a specification R into subspecifications is not orthogonal, in the sense that subspecifications may overlap arbitrarily and that the decomposition is not unique. Also, decompositions may vary from stakeholder to stakeholder. To fix our ideas, we consider a sample/simplistic example of a flight control system for a passen-

ger airplane, for which we list some sample subspecifications, some sample stakeholders, and some sample failure costs.

Sample Subspecifications:

- Ensure a smooth ride.
- Ensure adherence to flight vector within governmental guidelines.
- Ensure timeliness/ adherence to flight schedule.
- Ensure fuel efficiency.
- Ensure adherence to noise pollution standards.
- Ensure responsiveness to Auto Pilot settings.
- Ensure that the reverse thrust is never applied in mid air.
- Ensure that the landing gears are always activated prior to landing.
- Ensure that the aircraft speed does not fall below the stalling speed.
-

This list is neither complete (of course), nor orthogonal (many requirements overlap a great deal). As for stakeholders, we consider:

- The airplane's pilot.
- The passengers.

- The Aviation authorities (e.g. FAA).
- The CEO of the airline.
- The CEO of the aircraft manufacturer.
- The CEO of the insurance company ensuring the airline.
- Environmental activists/ organizations.
- Residents in the neighborhood of the airports (departure, destination of the flight).
- The beneficiary of a passenger's life insurance.

Sample Failure Costs:

- CEO of the Airline, Fuel efficiency: Bottom Line.
- CEO of the Airline, Timeliness: Company's reputation for timeliness.
- CEO of the Airline, Smoothness of the flight: Loyalty of the passengers.
- CEO of the aircraft manufacturer, Fuel efficiency: corporate image, selling point.
- Environmental activists, fuel efficiency: carbon imprint.
- Reverse thrust, passenger: Life.
- Reverse thrust, insurance company: Claims.
- Reverse thrust, airline company: Reputation for safety.

For a given stakeholder, the the Mean failure Cost can be estimated by considering the probability of failure with respect to all relevant subspecifications (i.e. those subspecification whose associated failure cost is non-zero), along with the associated failure costs for the stakeholder. By extension, a party who has no stake in the operation of a system views the MFC of the system as zero (one could argue that for this party the MTTF of the system is as good as infinite).

One of the consequences of this model is that the distinction between reliability and safety is blurred: what is usually referred to as *safety* is merely reliability with respect to a subspecification whose associated failure cost (with respect to some implicit stakeholder) is very high. The proposed model makes no distinction between low failure cost and high failure cost, and integrates a continuum of subspecification with varying failure costs. Hence MFC encompasses not only reliability, but also safety.

What we must consider now is how to estimate the mean failure cost, indeed what is its formula. To begin to understand this question, we consider the refinement structure of specifications, because it is at the core of how specifications are structured as aggregates of subspecifications.

3 Refinement Structure

We approach this problem from the standpoint of relational specifications. The results we present here have been explored in the context of relational specifications, though we suspect that they hold in other refinement calculi as well. Requirements specifications are represented by relations, which map system inputs to correct systems outputs. We define a partial ordering on relational specifications, under the name of *refinement ordering*. This ordering, which we represent by the symbol \sqsupseteq , is reflexive, transitive and antisymmetric. It is relevant to us because a system S is correct with respect to a (relational) specification R if and only if S refines R . Because the refinement ordering is transitive, we infer that if R refines R' , then any system that is correct with R is correct with R' .

If we turn to lattice properties of the refinement ordering, we find that it has lattice-like properties, in the following sense. Any two specifications R and R' which admit a common upper bound have a least upper bound, which we denote by $R \sqcup R'$. We refer to the least upper bound of R and R' as the *join* of R and R' , and we interpret it as the specification that captures all the requirements of R and all the requirements of R' . While the least upper bound exists conditionally, the greatest lower bound of two specifications is defined for any two specifications. We represent it by $R \sqcap R'$, and we find that it reflects requirements information that is common between R and

R' .

Perhaps as a consequence of the conditional nature of the join and the unconditional nature of the meet, the lattice of refinement has a universal lower bound, which is represented by the empty relation, but it has no universal upper bound. Instead, maximal elements of the refinement lattice are total deterministic relations. Figure 1 represents the outline of this lattice-like structure. This refinement structure allows to reinterpret some concepts we have been discussing informally, such as:

- That R_i is a subspecification of R : we can write this as:

$$R \sqsupseteq R_i.$$

- That R is the aggregate of subspecifications $R_1, R_2, R_3, \dots, R_n$:

$$R = R_1 \sqcup R_2 \sqcup R_3 \sqcup \dots \sqcup R_n.$$

- That R_1 and R_2 are independent requirements, in the sense that refinement one of them does not imply refining any part of the other:

$$R_1 \sqcap R_2 = \phi.$$

4 Estimating/ Approximating the Mean Failure Cost

We consider a system S and a specification R , and we are interested in estimating the mean failure cost of sys-

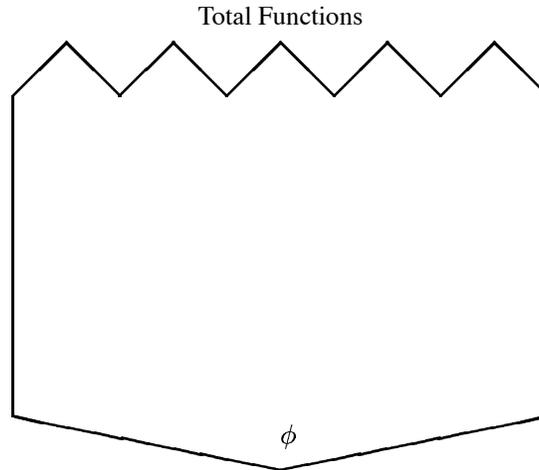


Figure 1: Refinement Lattice Structure

tem S with respect to specification R for some implicit stakeholder K , which we denote by $\Gamma(R)$. We consider a simple case, where specification R is the aggregate of subspecifications $R_1, R_2, R_3, \dots, R_n$, that are mutually independent (i.e. refining any one of them does not imply refining any part of any other subspecification). Then, we argue that the mean failure cost of S with respect to R is:

$$\Gamma(R) = \sum_{i=1}^n Pr(\overline{S \sqsupseteq R_i}) \times \Gamma(R_i),$$

where Pr represents the probability of an event. In other words, the mean failure cost of the aggregate specification is the weighted sum of the failure costs of the various components of the specification, weighted by the probability of failing with respect to each component.

The very unique situation above is the only one that is simple. In general, if we apply the formula above to an arbitrary collection of subspecifications, two things hap-

pen that make the results wrong: Because subspecifications overlap, many failure costs are counted more than once; also, because subspecifications overlap, failing to meet one is not independent from failing to meet another. One way to resolve this is to redecompose the specification into independent subspecifications, but this is rather very impractical, since the failure costs that we know of pertain to overlapping subspecifications.

Another approach, that which we are pursuing, is to explore identities that involve failure costs between of subspecifications. Examples of such identities include:

•

$$\Gamma(R_1 \sqcup R_2) = \Gamma(R_1) + \Gamma(R_2) - \Gamma(R_1 \sqcap R_2).$$

•

$$\min(\Gamma(R_1), \Gamma(R_2)) \leq \Gamma(R_1 \sqcap R_2).$$

•

$$R \sqsupseteq R' \Rightarrow \Gamma(R) \geq \Gamma(R').$$

Generalizations of the formula above are under investigation.

References

- [1] Fabrice Stevens, Tod Courtney, Sankalp Singh, Adnan Agbaria, John F Meyer, William H Sanders, and Partha Pal. Model based validation of an intrusion tolerant information system. In *Proceedings, SRDS*, pages 184–194, 2004.

Measuring Dependability as Mean Failure Cost

Ali Mili, NJIT
Frederick Sheldon, ORNL
CSIIRW, May 2007

Motivation

MTTF of System S with respect to specification R is
 M : mean time to next failure to satisfy R .

Two implicit assumptions:

- Independence wrt Subspecification. R is a monolith. Failing any requirement.
- Independence wrt Stakeholder. All stakeholders have the same stake in failure free operation.

Motivation

Distinction between

- Reliability: failure with respect to low stake requirements.
- Safety: failure with respect to high stake requirements.
- Security: failure with respect to security requirements.

Necessary distinctions?

Proposal

Wish List

- Varies according to which requirement is violated.
- Varies by stakeholder.
- Continuum of failure costs.
- Captures requirements dependencies (subsumptions, redundancies).
- Integrated measure of failure cost, regardless of source.

Tentative formulation

- Mean Time To Failure:

$$MTTF = \sum_{T>0} T \times P(t = T).$$

- Mean Failure Cost:

$$MFC(\sigma) = \prod_{R_i} P(\overline{S} \overline{\sigma} \overline{R_i}) \times C(\overline{S} \overline{R_i}, \sigma),$$

Must be heavily qualified. Not really a sum.

Decision Support

- MTTF: Go iff

$$\frac{OpTime}{MTTF} \times FailCost \ll Benefit.$$

- MFC: Go iff

$$MFC \ll ServiceValue.$$

Quantification Infrastructure

- MTTF: Probability distribution of failure.
- MFC:
 - Probability distribution of failure with respect to various sub-specifications. Depends on sub-spec, and on V&V effort.
 - Cost matrix, stakeholder vs sub-specification.

Difficulty: sub-specifications overlap.

Decompositions vary by stakeholder. MFC is a sum only for orthogonal sub-specifications.

Quantification Infrastructure: An Example

	SubSp1	SubSp2	SubSp3	SubSp4	SubSp5
StHld1					
StHld1			Cost		
StHld1					
StHld1					
StHld1					

Flight Control System, Stakeholders

- Pilot
- Passenger
- Aviation authorities (FAA)
- Environmental organization
- Airport area residents
- CEO, airline
- CEO, aircraft manufacturer
- CEO, insurance company

Flight Control System, Requirements

Requirements (naïve). User driven decomposition \neq V&V decomposition \neq Designer decomposition

- Smooth ride
- Adherence to flight vector
- Timeliness
- Fuel efficiency
- Noise pollution standards
- Responsiveness to AP setting
- Avoid reversing thrust in mid air
- Avoid stalling conditions

Overlap significantly.

Flight Control System, Stakes

- CEO of Airline, Fuel Efficiency: Bottom Line.
- Passenger, Fuel efficiency: 0.
- CEO of Airline, timeliness: corporate image.
- Passenger, timeliness: scheduling inconvenience.
- Safety requirement, passenger: Life.
- Safety requirement, CEO of airline: corporate image.
- Safety requirement, manufacturer: future sales.
- Safety requirement, insurance company: value of insurance claims.

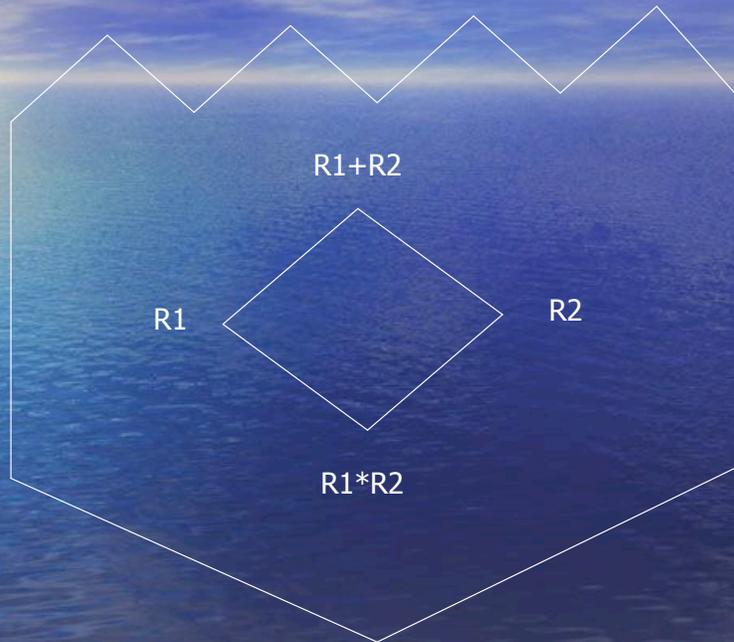
Quantifying MFC

Key Construct: Requirements Structure.

Requirements Specifications: ordered by refinement. Partial ordering.

Lattice Properties: Semi Lattice. Join exists conditionally (adding requirements). Meet exists unconditionally (common requirements). Universal lower bound. No universal upper bound.

Lattice-like Structure



Specification Structure

- Compound/ Composite Specifications:

$$R = \coprod_{1 \leq i \leq n} R_i.$$

- Composition is neither unique, nor orthogonal.
- Illustration:

$$\begin{aligned} &1155 \\ &= 15 \oplus 21 \oplus 33 \oplus 35 \oplus 55 \\ &= 105 \oplus 3 \oplus 11 \oplus 7 \oplus 55 \\ &= 3 \oplus 5 \oplus 7 \oplus 11. \end{aligned}$$

Computing MFC

- Specification,

$$R = \coprod_{1 \leq i \leq n} R_i.$$

- Mean Failure Cost, if R_i 's are independent:

$$MFC(\sigma) = \sum_{R_i} P(\overline{S} \text{ } \tilde{\text{O}} \tilde{\text{O}} R_i) \times C(\overline{S} \text{ } R_i, \sigma).$$

Requirements Independence

Independent Sub-specifications:

$$R_1 \otimes R_2 = 0.$$

Interpretation: Relatively prime.

Implications

$$P(S \text{ is multiple of } (R_1 \oplus R_2)) = P(S \text{ is multiple of } R_1) \times P(S \text{ is multiple of } R_2).$$
$$C(\overline{S \text{ is multiple of } (R_1 \oplus R_2)}) = C(\overline{S \text{ is multiple of } R_1}) + C(\overline{S \text{ is multiple of } R_2})$$

- Interpretation: being multiple of 11 makes no difference whether number is or is not multiple of 15. Being multiple of 9 makes us more likely to be multiple of 15.

General Case

- Stakeholders know failure costs of arbitrary sub-specifications. Needed: identities that allow us to derive failure costs of independent (orthogonal) sub specs.
 - Refinement Identities,
 - Lattice Identities.
- Probabilities: textbook material.

Cost Identities

- Refinement Ordering:

$$R_1 \preceq R_2 \Rightarrow C(\overline{S \mid R_1}) \geq C(\overline{S \mid R_2}).$$

- Refinement Lattice:

$$C(\overline{S \mid (R_1 \oplus R_2)}) = C(\overline{S \mid R_1}) + C(\overline{S \mid R_2}) - C(\overline{S \mid (R_1 \otimes R_2)}).$$

Conclusion/ Summary

Mean Failure Cost

- Acknowledges variance in failure cost by stakeholder.
- Acknowledges variance in failure cost by requirement.
- Provides uniform metric for many dimensions of dependability.
- Consistent with spirit of Value Based Software Engineering.

Very speculative; very preliminary. Urgent: sample application.

Survivability in Wireless Networks: A Case for Overhead Reduction

Axel W. Krings
University of Idaho
Moscow, Idaho 83844-1010, USA

Abstract: *A link scheduling model is presented that utilizes primary-backup scheduling for packet scheduling. The advantage of this scheduling paradigm is that overhead can be suppressed in the fault-free case and overhead only needs to be endured in case of actual faults. The scheduling paradigm significantly increases survivability and can be used to reduce overhead of redundancy-based approaches. The foundation for using primary-backup scheduling in networks is derived. The schemes presented are very effective for multi-path protocols and MIMO and can be applied where watchdog-based algorithms fail or where geographic-centric disruptions render local approaches useless.*

1 Introduction

With the tremendous growth of wireless applications in recent years comes great concern for the lack of reliability, security and survivability. Especially in applications in the area of ad hoc and sensor networks there are many new challenges due to their features and the inherent characteristics of wireless technology. The main considerations have been routing and the overhead resulting from dealing with disruptions of the communication paths. As a result, many protocols have been introduced. However, in critical applications operating in hostile environments the security and survivability requirements may be much higher than usual and fault assumptions should include pathological behavior capable of introducing value faults. Furthermore, most research has focused on operation in benign environments where security considerations were not the driving motivation.

Since this work relates to tolerance of faults of different types under possibly pathological scenarios, we need to explore redundancy mechanisms. As such, any approach utilizing multipath and multiflow communication could have the potential for tolerating faults, if these concepts are exploited for reliability [12]. Many multipath and multiflow approaches have been presented in the literature, but their focus has not been on tolerating diverse faults but have rather been limited to overcome benign link or node faults. For example, the concept of multiflow has been used in [13] in the context of QoS enhancement, however, the focus is on transmission congestion. Multipath routing has been used to increase end-to-end reliability, e.g. the MP-DSR protocol in [8] forwards outgoing packets along multiple paths that are subject to a particular end-to-end reliability requirement; this however raises overhead concerns.

Primary and backup communication paths are considered in [9]. However, disjoint paths are not exploited for data redundancy but discarded as unwanted overhead. In their use of redundant disjoint paths the overhead to resilience tradeoff becomes unfavorable for a larger number of paths [2, 10]. We consider a different approach to primary and backup communication adopted from fault-tolerant multi-processor scheduling with focus on overhead reduction.

2 Network Survivability Model

For our purposes, the term survivability and reliability may be interchanged. Survivability was elected to emphasize that the operating environment may be malicious.

The communication network is represented as a digraph $G = (V, E)$, where computational nodes are the vertices and communication “links” are the edges. An edge e_{ij} is present in E if node v_j receives the signals of node v_i . If a source node v_S wants to establish a communication path with a destination node v_D , then the reliability of the path v_S-v_D is clearly depending on the reliability of the nodes and communication links along the path. To tolerate faults, may they be of benign nature or maliciously induced, one can chose to increase the survivability of the primary communication path v_S-v_D , e.g. using schemes such as presented in [11] or [6], or one can use a multi-path approach, considering alternative paths under the assumption that a certain threshold of “good” paths can mask faults.

Adding path diversity to the communication scheme, one inherits the undesirable overhead of multi-path routing and packet redundancy. In order to reduce overhead, we revert to using a proven mechanism from real-time scheduling. Specifically, we adopt primary-backup (PB) link scheduling, which was introduced in the context of fault-tolerant scheduling in real-time multiprocessor systems [1, 4]. Essentially, non-preemptive computational tasks (consisting of a primary and a backup task) are accepted into the real-time system if a feasibility test guarantees that the task can be scheduled to meet its deadline. Otherwise the task is rejected. If the primary task fails, due to transient or permanent faults, the backup task is executed. To avoid unnecessary overhead in the non-fault case, backup overloading is utilized. Whereas multiprocessor scheduling considers scheduling tasks onto processors, we are concerned with scheduling packets onto communication links. As such, a communication link and a processor are analogous. Similarly, data packets and computational tasks are analogous.

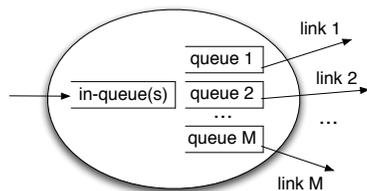


Figure 1: Conceptual Network Node

To make the analogy between links and processors some justification is necessary. We view a network node v_i of G as having separate links, i.e. channels, as shown in Figure 1. Packets are received into one or more input queues and scheduled on links via their associated output queues. This makes perfect sense in fixed networks, but in wireless nodes this view is only conceptual. Only in the case of MIMO (multiple-input-multiple-output), where dual-array multiple-antenna systems are used, is this representation apparent. However, in the absence of MIMO, we can still justify this view using multiplexing. For example, consider code division multiple access (CDMA). In CDMA multiple channels are multiplexed without dividing up the channel by time, thus logically implementing the concept of Figure 1. Time division multiple access (TDMA), on the other hand, allows multiple links to be emulated by sharing the link in a time-division scheme. Again, assuming the time slots are relatively small, the concept in the figure is preserved.

Next, we introduce notation for scheduling packets on links, or practically, their associated queues. Given the abstraction of a wireless node above, let L_j denote link j . We will speak of “scheduling packets on links”, which actually means that packets are scheduled in the respective queues. Associated with each data *packet* P_i are the attributes *arrival time* a_i , i.e. the time at which P_i enters the in-queue of the node, the *ready time* r_i , which is the time the packet is ready to be moved to the outgoing link queue, the *start time* s_i , the time the packet is starting to be transmitted, *transmission time* l_i , which is the time it takes to send out the packet of size l , the *finish time* f_i , the time the last bit of the packet has left the link, and the *deadline* d_i , which defines the latest deliver time as needed to guarantee QoS. Note that $l_i = f_i - s_i$.

For each packet P_i a *primary* Pr_i and a *backup* copy Bk_i are defined. Note that “copy” can refer to redundant pointers to a single data object. The purpose of Bk_i is that, if the transmission of Pr_i fails, it will serve as a backup. The deadline for the acknowledgment of the primary’s delivery in the fault-free case is called *acknowledge time*, $ack(Pr_i)$. Thus $ack(Pr_i)$ constitutes the maximal time up to which one can wait for an acknowledgment. The actual time when Pr_i is acknowledged is denoted by $t_{ack}(Pr_i)$, with $t_{ack}(Pr_i) \leq ack(Pr_i)$ in the fault-free case. Thus, if an acknowledgment of delivery has not been received by $ack(Pr_i)$, it is assumed that a fault

has occurred. However, if Pr_i is successfully delivered, which would be confirmed at $t_{ack}(Pr_i) \leq ack(Pr_i)$, then Bk_i can be discarded from the queue. The backup only requires link resources if the primary fails. Otherwise, the only penalty for utilizing the backup is the overhead associated with queue management. From a practical point of view, the value for $ack(Pr_i)$ is chosen based on the expected transmission time in the no-fault scenario. If the expected time it takes to acknowledge Pr_i is t_a , then $ack(Pr_i) = s(Pr_i) + \alpha t_a$ where $\alpha \geq 1$ is a constant affecting how sensitive the fault detection is. This should be only an expected (pessimistic) value, and thus high accuracy in a minimal $ack(Pr_i)$ may not be meaningful.

An acknowledge $t_{ack}(Pr_i)$ of a packet P_i addresses the round-trip delay of the packet, i.e. the time to deliver the packet plus the time it takes to send and deliver the acknowledge back to the sender. We will assume that the only way we can practically expect that a packet is delivered is at the time of its acknowledge $t_{ack}(Pr_i) \leq ack(Pr_i)$. This way we avoid the issues associated with the case where faults occur during the time of transmission or acknowledge. Note that $ack(Pr_i)$ is a parameter reflecting the expected transmission time in the absence of faults. This should not be confused with timeout parameters of the transport control protocol, e.g. TCP

The packet attributes defined for P_i above will be used for Pr_i and Bk_i as well, e.g. $s(Pr_i)$ is the primary's starting time or $f(Bk_i)$ the finishing time of the backup. We assume that in the schedule of packet P_i the timing relationship between Pr_i and Bk_i is $a_i \leq r_i \leq s(Pr_i) < f(Pr_i) \leq ack(Pr_i) \leq s(Bk_i) < f(Bk_i) < t_{ack}(Bk_i) \leq d_i$. Furthermore, we assume that if Pr_i fails, then backup Bk_i will succeed. Thus, at most one fault is assumed for packet P_i . An important assumption for PB scheduling is that the primary and backup of P_i cannot be scheduled on the same link, i.e. $L(Pr_i) \neq L(Bk_i)$.

In order to minimize the overhead associated with scheduling backup packets the concept of backup overloading is adopted. Figure 2 shows the concept. Packet P_1 has its primary Pr_1 scheduled on link L_1 and its backup Bk_1 on L_2 . Similarly, P_2 has Pr_2 scheduled on L_3 with its backup Bk_2 on L_2 , thus overloading L_2 from $s(Bk_2)$ to $f(Bk_1)$. This has consequences for the assumptions about faults.

In the figure both backup packets overlap. It can eas-

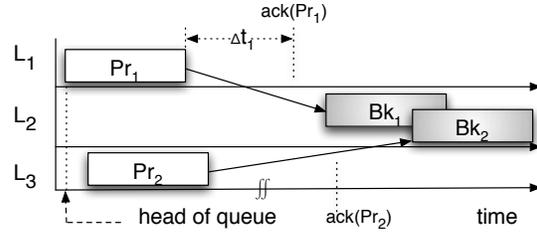


Figure 2: Backup Overloading

ily be shown that if two backups Bk_i and Bk_j are overlapping on a link, then their respective primaries must be scheduled on different links. Conversely, if Pr_i and Pr_j are scheduled on the same link, then their backups must not overload.

The desirable feature of backup scheduling is that given packet P_i , backup Bk_i can be deleted if Pr_i is delivered successfully at $t_{ack}(Pr_i) \leq ack(Pr_i)$. The usage of backup overloading requires the introduction of the notion of *Time to Second Fault* (TTSF), which is the time at which a second fault can occur without risking the loss of a packet due to overloading. Note that the smaller TTSF is, the more resilient the system becomes to second faults. Let $TTSF(L_i)$ indicate the time to second fault with respect to link L_i . It can be shown that TTSF is the maximum $TTSF(L_i)$, which is defined as the maximum time of the acknowledge of the primary whose backup is scheduled on the link and the acknowledge of the backup scheduled on the link.

Whereas the previous discussion considered benign and omission faults, we now turn to the impact of value faults, i.e. the case where the content of a packet is manipulated. To tolerate k such faults, by definition, one needs $2k + 1$ redundant packets, which will guarantee that the good packets are in the majority. This should not be confused with the Byzantine majority of asymmetric faults in distributed agreement [7].

If one wants to detect a single value fault using PB scheduling one can extend the concept to include two primary copies and a backup. Thus, for packet P_i we consider primary Pr_i , secondary Se_i and backup Bk_i . The deadline for the acknowledge of both Pr_i and Se_i is assumed to be $ack(Pr_i)$. Upon acknowledgment of both

Pr_i and Se_i the backup Bk_i is unscheduled. Conversely, if either Pr_i or Se_i fail to acknowledge, Bk_i is required. It should be noted that in principle scheduling of a primary and a secondary on disjoint links allows for correction in the case of a benign and omission fault and for detection of a value fault. In the latter case, the possible tie between packets can be resolved with the backup packet, constituting fault recovery. Thus, logically this scheme corresponds to the so-called hybrid-SCP-TMR [3], where in the case of real-time multi-processor scheduling two copies execute first, implemented as a Self Checking Pair (SCP). If the outputs do not agree, the third copy is scheduled to break the tie, thus implementing Triple Modular Redundancy (TMR).

In the context of link scheduling, the detection mechanism of the hybrid-SCP-TMR requires further explanation. Note, that by the definition of this configuration the detection of a value fault requires that a difference in the packet contents must be observed. In the multiprocessor case of [3] this is done by a comparator, e.g. a voting task, which detects that the results of the two tasks differ. In the network protocol stack the detection of differences in the packet contents can be observed by the receiver of the packets, e.g. by the observation that the signatures (or frame check sequences) of the primary and secondary packets do not match.

If the receiving node detects that the content of Pr_i and Se_i do not match, then an explicit or implicit message $reject_i$ is issued. An explicit reject message identifies the mismatch of the packet content between the two copies of P_i . Alternatively, an implicit reject is realized by simply not acknowledging a packet, thus triggering a timeout at $ack(Pr_i)$. In both cases backup Bk_i is sent to break the tie.

Next, we want to establish the timing relationships of the packets. Assuming $s(Pr_i) \leq s(Se_i)$, the timing relationship between Pr_i , Se_i and Bk_i is $r(P_i) \leq s(Pr_i) \leq s(Se_i) < ack(Pr_i) \leq s(Bk_i) < f(Bk_i) < t_{ack}(Bk_i) \leq d_i$. Furthermore, we have $f(Pr_i) \leq ack(Pr_i)$ and $f(Se_i) \leq ack(Pr_i)$.

To avoid packet loss in the presence of a permanent value fault the primary, secondary and backup of a packet must be scheduled on different links, i.e. $L(Pr_i) \neq L(Se_i) \neq L(Bk_i)$. This follows directly from the function of a TMR, which can handle exactly one value fault under the assumption of independence of faults. Schedul-

ing two or more copies of the packet on the same link would violate this independence assumption.

As in simple PB scheduling we assume that if Pr_i or Se_i fail, i.e. one packet content is corrupted, then backup Bk_i will succeed. Assume that packets P_i are scheduled using backup overloading under a hybrid-SCP-TMR strategy. Furthermore, assume that at time t link L_k experiences permanent value faults. Then another fault can be tolerated at time $t' = \max\{t_1, t_2, t_3\}$, where

$$t_1 = \max\{t_{ack}(Bk_i), \forall Pr_i : L(Pr_i) = L_k\}$$

$$t_2 = \max\{t_{ack}(Bk_i), \forall Se_i : L(Se_i) = L_k\}$$

$$t_3 = \max\{t_{ack}(Pr_i), t_{ack}(Se_i), \forall Pr_i, Se_i : L(Bk_i) = L_k\}$$

If the exact time of $t_{ack}(Pr_i) \leq ack(Pr_i)$ is not known, $t_{ack}(Pr_i) = ack(Pr_i)$ must be assumed. The same holds for Se_i and Bk_i . Contrary to the case of a simple primary-backup scheme, now the overhead associated with the secondary has to be tolerated. However, overhead induced by the backup packets are still suppressed in the non-fault case.

3 Reliability Analysis

The reliability of a communication channel, $R(t)$, is the probability that the communication is failure-free during the entire time-interval $[0, t]$. In order to determine the reliability of communication using PB link scheduling four approaches were analyzed, assuming fail rate $\lambda = 10^{-3}$ per time unit. First, a Single Path was considered, i.e. a communication path without packet redundancy. Second, simple PB scheduling was considered, however, we relaxed the assumptions about a guaranteed delivery of the backup packet, since in a real system no such guarantee can be given. Thus, the results shown are more realistic, but at the same time more pessimistic. Third, we considered Hybrid SCP-TMR scheduling for value faults, again under the relaxation of guaranteed backup packet delivery. Fourth, we used the previous scheme, but only considered benign faults. This effectively changes the hybrid SCP-TMR into a 1-of-3 system. The results of the four different approaches are shown in Figure 3. As can be seen all primary-backup approaches show significant

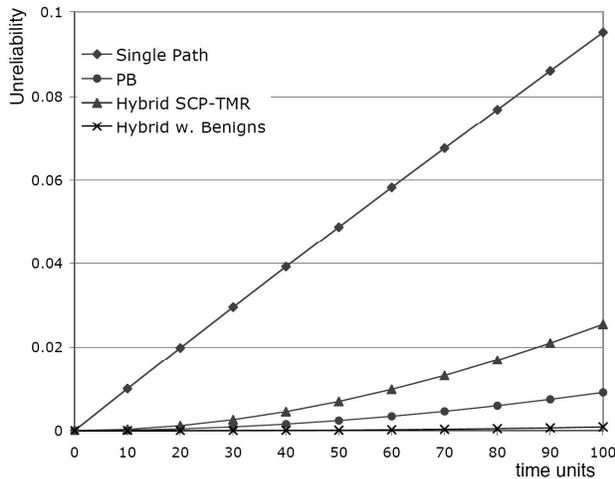


Figure 3: Unreliability for PB Scheduling

improvements over the single path approach. Furthermore, in non-faulty scenarios the improvements come at no communication cost.

4 Conclusions

Primary-backup link scheduling was introduced as a mechanism that significantly increases survivability for wireless networks. The concept was demonstrated for multi-path networks for simple fault models (benign and omission faults) and value faults. The overhead associated with the scheme results in only negligibly small local link scheduling overhead in the fault free case. Thus, the burden of multi-path packet overhead was only induced if an actual fault occurred.

References

[1] R. Al-Omari, et.al, Efficient overloading techniques for primary-backup scheduling in real-time systems, *Journal of Parallel and Distributed Computing*, Vol. 64, Issue 5, pp. 629-648, May 2004.

[2] D. Ganesan, et.al., *Highly-resilient, energy-efficient multi-path routing in wireless sensor networks*, Mobile Comput-

ing and Communications Review, Vol. 4, No. 5, October 2001.

[3] O. Gonzalez, et.al., Adaptive Fault Tolerance and Graceful Degradation Under Dynamic Hard Real-time Scheduling, *Proc. IEEE Real-Time Systems Symposium*, pp. 79-89, 1997.

[4] S. Ghosh, et.al., Fault-Tolerant Scheduling on a Hard Real-Time Multiprocessor System, *Proceedings of the International Parallel Processing Symposium*, pp. 775-782, 1994.

[5] S. Ghosh, et.al., Fault-tolerance through scheduling of aperiodic tasks in hard real-time multiprocessor systems, *IEEE Trans. Parallel Distributed Systems*, 8 (3), pp. 272-284, March 1997.

[6] A. Krings, and Z. Ma, *Fault-Models in Wireless Communication: Towards Survivable Ad Hoc Networks*, Military Communications Conference, MILCOM 2006, pp. 1-7, 23-25 Oct. 2006.

[7] L. Lamport, et.al., *The Byzantine Generals Problem*, ACM Transactions on Programming Languages and Systems, Vol. 4, No. 3, pp. 382-401, July 1982.

[8] R. Leung, J. Liu, E. Poon, A. Chan and B. Li, MP-DSR: A QoS-aware Multi-path Dynamic Source Routing Protocol for Wireless Ad-Hoc Networks, *Proc. 26th Annual IEEE Conference on Local Computer Networks*, LCN 2001, pp. 132-141, 2001.

[9] H. Liu and D. Raychaudhuri, *Label Switched Multi-path Forwarding in Wireless Ad-Hoc Networks*, Proceedings of the 3rd Intl Conf. on Pervasive Computing and Communications Workshops, (PerCom 2005 Workshops), pp. 248-252, 2005.

[10] M. K. Marina and S. R. Das, *On-Demand Multipath Distance Vector Routing for Ad Hoc Networks*, Proc. of the International Conference for Network Protocols (ICNP), Riverside, USA, pp. 14-23, 2001.

[11] Sergio Marti, et.al., *Mitigating routing misbehavior in mobile ad hoc networks*, Mobile Computing and Networking, pp. 255-265, 2000.

[12] S. Mueller, R. P. Tsang, and D. Ghosal, *Multipath Routing in Mobile Ad Hoc Networks: Issues and Challenges*, MAS-COTS 2003, LNCS 2965, Springer-Verlag, pp. 209-234, 2004.

[13] N. Thanthy, et.al., *TCP-M: Multiflow Transmission Control Protocol for Ad Hoc Networks*, EURASIP Journal on Wireless Communications and Networking, Article ID 95149, 16 pages, 2006.

Survivability in Wireless Networks: A Case for Overhead Reduction

Axel Krings
Computer Science Department
University of Idaho

krings@cs.uidaho.edu
<http://www.cs.uidaho.edu/~krings/>

Outline

- Introduction
- Background and Definitions
- Wireless Network Model
- Increasing Path Reliability
- Overload Scheduling
- Reliability Analysis
- Conclusions

Introduction

- Wireless Networks have gained great popularity
- Special focus
 - Ad hoc networks, MANETs
 - Sensor networks
- Wireless has many potential problems w.r.t.
 - Security
 - Reliability
 - Mobility

Introduction

- Problems include
 - Security
 - broadcast, “everybody can see”
 - nodes may be captured/impersonated/... many flavors
 - Reliability
 - nodes may be mobile
 - links and nodes have reliability/availability constraints
 - external interference, benign - malicious

Fault Models

- What are the assumptions about faults?
 - crash faults, omission faults, etc.
 - independence of faults
 - dependence of faults => common mode fault
 - recovery differs greatly depending on the fault model

Recovery needs Redundancy

- Time redundancy
- Information redundancy
- Spatial redundancy

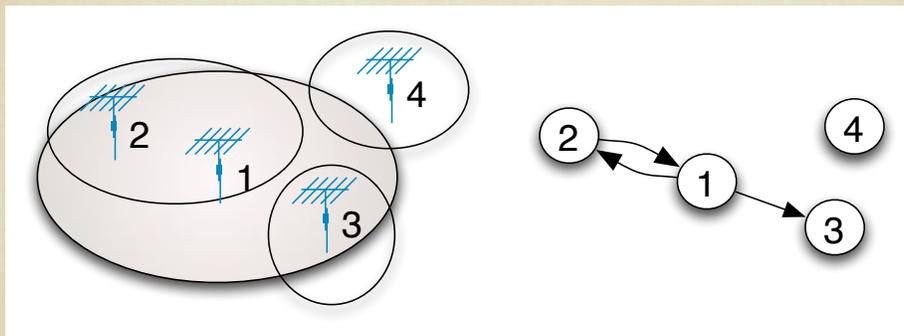
e.g. if one considers s symmetric and b benign faults, then one needs $N > 2s + b$ redundant units to mask the faults

Fault Assumptions

- Faults are seen only in the context of their definition within the fault model under consideration
- Many mechanisms from security & fault-tolerance
 - e.g. encryption, authentication, ...
- 🕒 BUT in the end, their impact on the faults they can produce is what really counts

Network Graph

- Network Graph G is a digraph

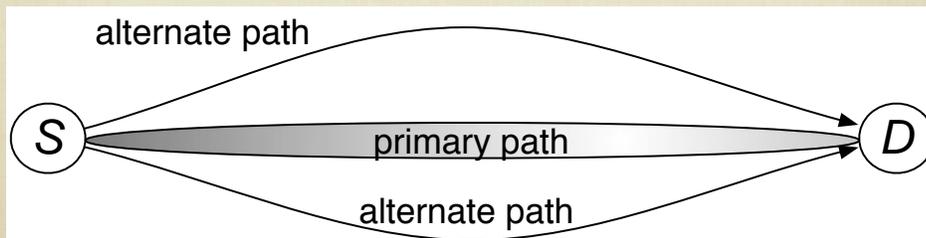


Network Graph

- General Communication Model

- Reliability considerations:

- increase path reliability/security
 - utilize multipath approach

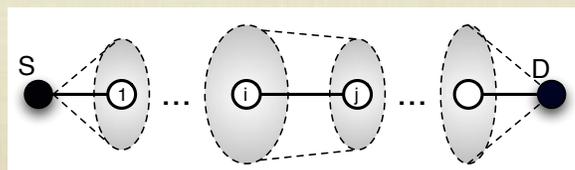


Increasing Path Reliability

- Two dimensional watchdog approach

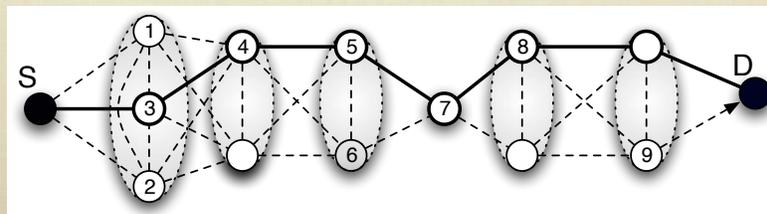
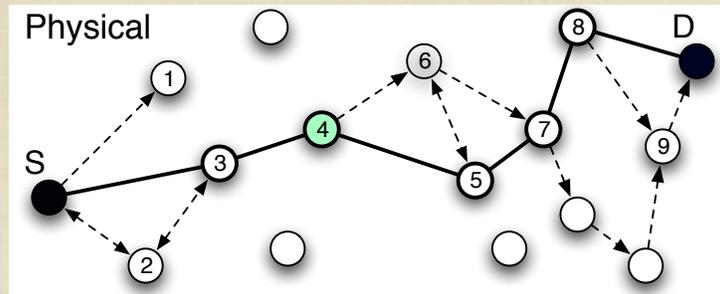
- Krings Axel and Zhanshan Ma, "Fault-Models in Wireless Communication: Towards Survivable Ad Hoc Networks", MILCOM 2006, 23-25 October, 7 pages, 2006.

- Use neighborhood induced by general join graph (GJG)



Example

- Assume nodes are moved to implement the GJG below

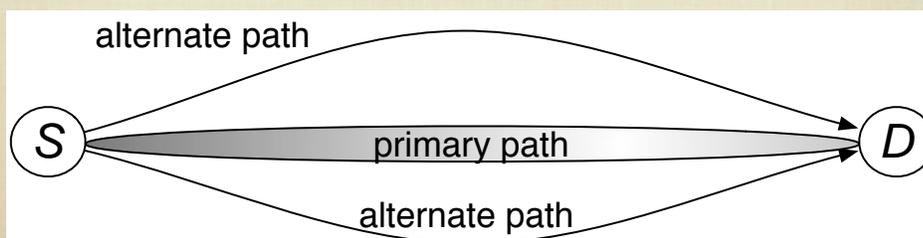


CSIIRW 2007, Axel Krings

11

Multi-Path Approach

- Increased Reliability through Multi-path Routing
 - single path (even if GJG) may be subject to local disturbance
 - alternate paths can serve as multi-path option
 - multi-path is not a new concept, but this is different
 - what about the overhead....?



CSIIRW 2007, Axel Krings

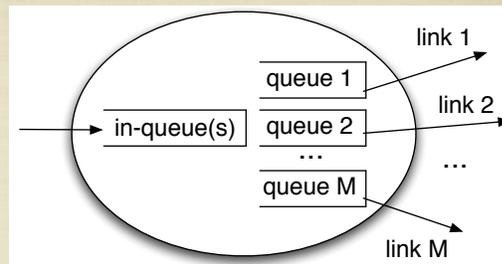
12

Simple Overlay Scheduling

- Used in Real-time Multi-processor Systems
 - Ghosh [1994], Tsuchiya [1995], Ghosh [1997], Manimaran [1998], Al-Omari [2004],...
- Primary-backup scheduling
 - overhead is negligibly small in the fault-free case
 - non-preemptive task consists of primary and backup
 - accept new task into system if feasibility test guaranteed that task can be scheduled to meet its deadline
 - uses backup overloading to avoid unnecessary overhead

Conceptual Network Node

- Node is viewed as having
 - input queue(s)
 - output queues/links
- This makes sense in fixed network, but what about wireless nodes?
 - MIMO
 - CDMA
 - TDMA



Packet Attributes

- A Packet P_j is scheduled on link L_i
- Packet attributes

a_j	arrival time
r_j	ready time
s_j	start time (of transmission)
l_j	transmission time (depends on length and line speed)
f_j	finish time
d_j	deadline

Primary-Backup

- A packet P_i consists of two parts
 - Primary Pr_i
 - Backup copy Bk_i
 - Bk_i serves as backup if primary fails
 - If Pr_i is delivered successfully, Bk_i is “unscheduled”

Primary-Backup

- Acknowledge time $ack(Pr_i)$
 - constitutes the maximum time up to which one can wait for an acknowledge
- Actual acknowledge time $t_{ack}(Pr_i)$
 - actual time when Pr_i is acknowledged

$$ack(Pr_i) = s(Pr_i) + \alpha t_a$$

- alpha is a constant affecting how sensitive the fault detection is
- t_a is the expected time to acknowledge Pr_i

Restrictions on Primaries

- Lemma 1
 - The primary and backup of P_i cannot be scheduled on the same link

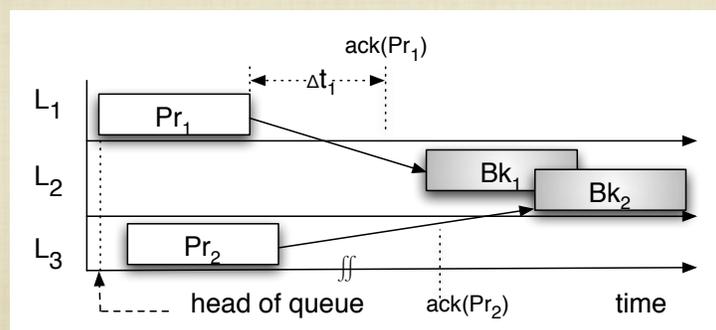
$$L(Pr_i) \neq L(Bk_i).$$

Restrictions on Primaries

- **Lemma 2** Given Lemma 1, if two backups Bk_i and Bk_j are overlapping on a link, i.e. $S(Bk_i) \cap S(Bk_j) \neq \Phi$, then Pr_i and Pr_j must be scheduled on different links, i.e. $L(Pr_i) \neq L(Pr_j)$. Conversely, if Pr_i and Pr_j are scheduled on the same link, then their backups must not overload.

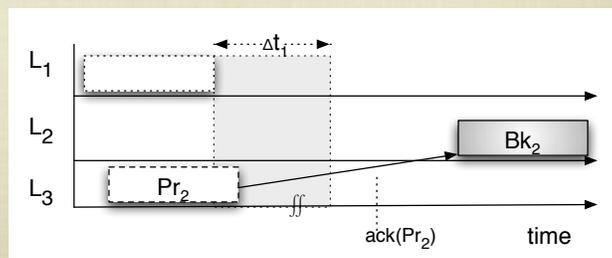
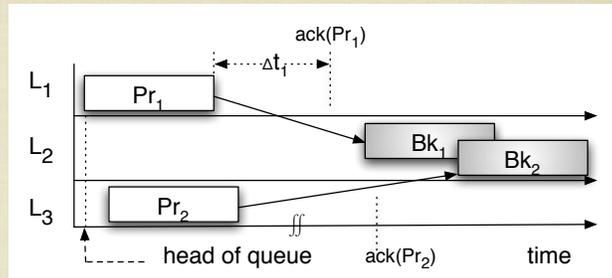
Backup Overloading

- Lemma 2
 - If two backups Bk_i and Bk_j are overlapping on link L_k , then Pr_i and Pr_j must be scheduled on different links, i.e.,



No-Fault Scenario

- If acknowledgment $t_{ack}(Pr_1)$ arrives in Δt_1 then Bk_1 is unscheduled
- Note: at $t_{ack}(Pr_1)$ packet Pr_2 may or may not have been sent out, but acknowledgment may not arrive until $ack(Pr_2)$



Unschedulering

- **Lemma 3** Given packet P_i , backup Bk_i can be deleted only if Pr_i is delivered successfully at $t_{ack}(Pr_i) \leq ack(Pr_i)$.

Time-To-Second-Fault

- Link 1 experiences a permanent fault

$$\text{TTSF}(L_2) = t_{ack}(Bk_1) \leq ack(Bk_1)$$

$$\text{TTSF}(L_3) = t_{ack}(Pr_2) \leq ack(Pr_2)$$

$$\text{TTSF} = \max\{\text{TTSF}(L_2), \text{TTSF}(L_3)\}$$

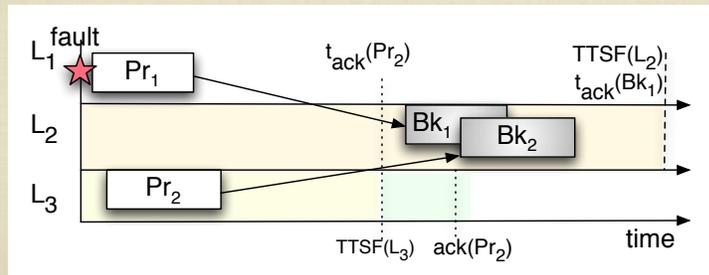
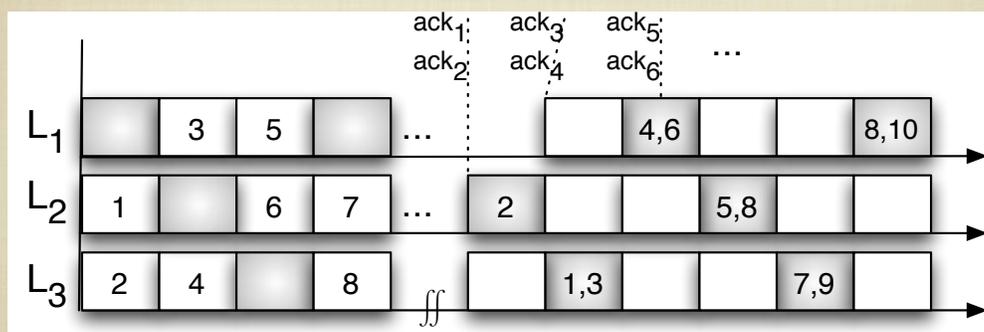


Figure 9. TTSF after link fault

Fixed Packet Link Allocation

- Backup slots are striped

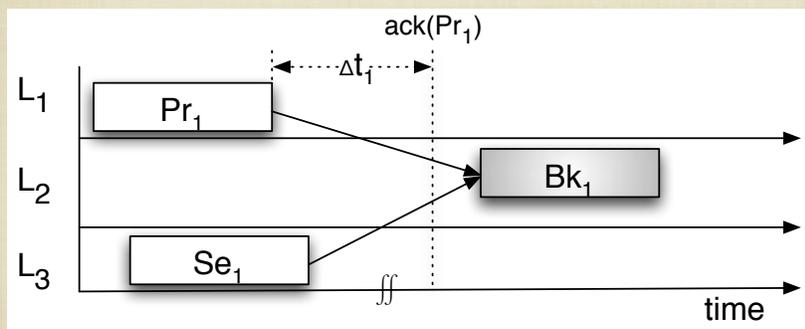


Overlay Scheduling for Hybrid Fault Models

- The concept can be extended to include extensions, analogous to the alternatives in FERTstones
 - [Bondavalli, Stankovic, Strigini 1993]
 - TMR, hybrid-selfchecking-TMR, k-of-N

Hybrid-selfchecking-TMR

- The concept is essentially equivalent to



Permanent Value Fault

Lemma 4 *Assume there is a source for permanent value faults. To avoid packet loss, the primary, secondary and backup of P_i must be scheduled on different links, i.e. $L(Pr_i) \neq L(Se_i) \neq L(Bk_i)$.*

Assume Value Fault

Theorem 2 *Assume that packets P_i are scheduled using backup overloading under a hybrid-SCP-TMR strategy. Furthermore, assume that at time t link L_k experiences permanent value faults. Then another fault can be tolerated at time $t' = \max\{t_1, t_2, t_3\}$, where*

$$t_1 = \max\{t_{ack}(Bk_i), \forall Pr_i : L(Pr_i) = L_k\}$$

$$t_2 = \max\{t_{ack}(Bk_i), \forall Se_i : L(Se_i) = L_k\}$$

$$t_3 = \max\{t_{ack}(Pr_i), t_{ack}(Se_i), \forall Pr_i, Se_i : L(Bk_i) = L_k\}$$

If the exact time of $t_{ack}(Pr_i) \leq ack(Pr_i)$ is not known, $t_{ack}(Pr_i) = ack(Pr_i)$ must be assumed. The same holds for Se_i and Bk_i .

Reliability of PB Scheduling

- Consider again previous example
- Four scheduling approaches
 - Single Path
 - PB Scheduling
 - Hybrid SCP-TMR Scheduling (for value faults)
 - Hybrid with benign faults only

Analytical Model

- Unreliabilities

Communication scenario	Unreliability $F(t) = 1 - R(t)$
Single Path	$F(t) = 1 - e^{-\lambda t}$
PB	$F(t) = 1 - 2e^{-\lambda t} + e^{-2\lambda t}$
Hybrid SCP-TMR	$F(t) = 1 - 3e^{-2\lambda t} + 2e^{-3\lambda t}$
Hybrid with Benigns	$F(t) = 1 - 3e^{-\lambda t} + 3e^{-2\lambda t} - e^{-3\lambda t}$

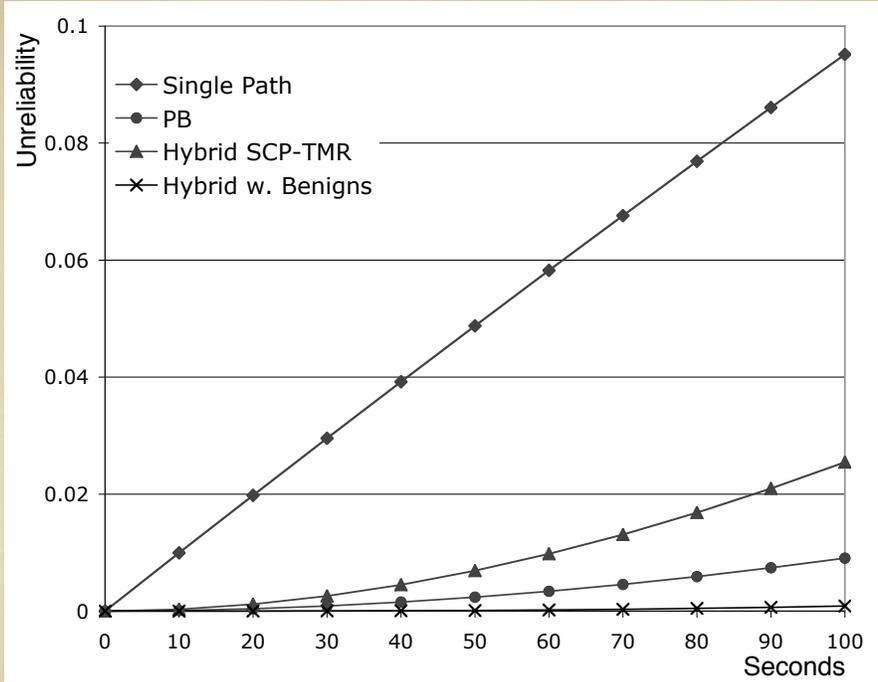


Figure 7: Communication link Unreliability - 100 seconds

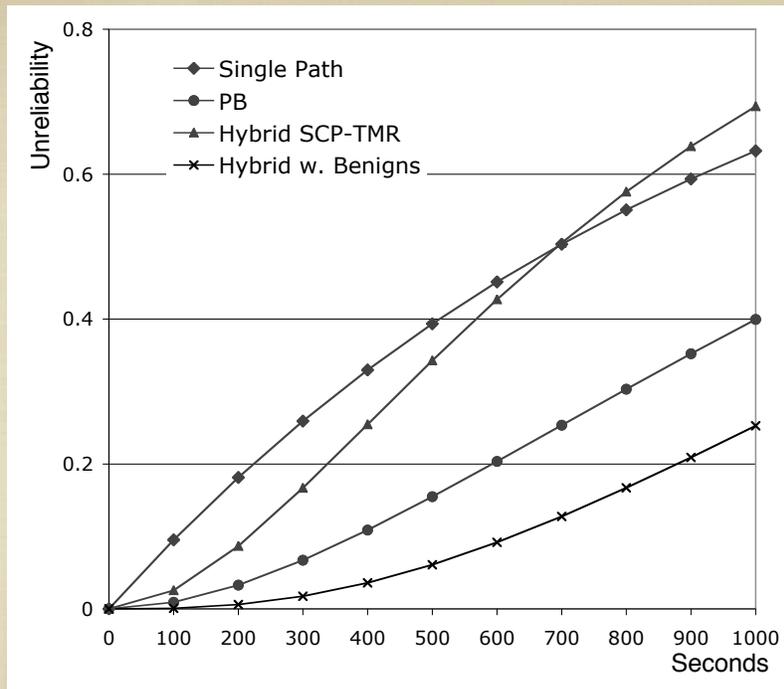


Figure 8: Communication link Unreliability - 1000 seconds

Conclusions

- Reliability and survivability of wireless networks can be greatly improved by using cross-monitoring, i.e. GJG
- PB scheduling reduces overhead, increases network reliability and has potential to drastically reduce delays
e.g. RTO (Retransmission Timeout period) in TCP
- Can be used to adapt network to the required level of reliability

The Layered Security Model and its Representation using Bigraphs to Analyse Critical Infrastructure

Clive Blackwell

*Information Security Group, Royal Holloway, University of London
Egham, Surrey. TW20 0EX. United Kingdom.
C.Blackwell@rhul.ac.uk*

1 Introduction

There is a widening gap between our understanding of systems and their ever increasing complexity, functionality and connectivity. We require more sophisticated functionality for novel applications, and systems to interoperate with each other dynamically and autonomously to meet their different objectives. Piecemeal defences address limited technical problems, rather than tackle system requirements comprehensively. This leads to brittle systems with single points of failure that break easily with unpredictable consequences. We still often rely on the binary distinction between insider and outsider, whereas we need more fine grained measures to cope with a continuum of access rights and to manage the effects of successful attacks.

Some security issues that need to be seriously addressed include emergent system behaviour, effects at a distance, unexpected changes to a system and its environment, and new methods of attack. As Einstein said, “we can’t solve problems by using the same kind of thinking we used when we created them”. We provide an informal architectural model that can be formalised, which can analyse systems that have multiple independent mechanisms operating at different layers and locations with different protective characteristics. This helps to plan, design and build systems to help provide comprehensive protection and assurance that they will complete their missions in the presence of security vulnerabilities and functional defects rather than respond tactically to every little problem.

Only a few systematic models of security can represent systems in their entirety, rather than as technical systems alone. There is the longstanding effort in classifying the important aspects of dependability, including security, which offers a comprehensive taxonomy of the different types of fault and methods to manage them [1]. Neumann [2], and originally Neumann and Parker [3], organised systems into eight layers for security analysis, which are listed starting from the highest layer as the external environment, user, application, middleware, networking, operating system, hardware and internal environment. We consider this as a logical and useful aid to understanding systems, but we have introduced some new organisational criteria to give a simplified model that has only three layers. Howard and Longstaff [4] present a classification system for network security incidents, which shows the different types of entity involved in an attack and their interrelationships. The classification can be extended with dual concepts to model the defence, and by explicitly including the semantic and physical aspects of systems as well as computer and networks.

2 The Layered Security Model

2.1 The Layers

We model systems and their interactions in a three-layer hierarchy, where each layer can have sub-layers when required for detailed analysis (figure 1). The *semantic* or *conceptual layer* is the top layer that includes people and the abstract representation of systems including their requirements. The *logical layer* is an intermediate layer containing entities in an intangible form including data and software that are stored and processed on computers and transmitted between them. The purpose of this layer is to carry out the objectives of semantic layer entities, as they cannot interact directly with logical entities. For example, people are represented by a logical proxy such as an account, a process or a cryptographic key to act on their behalf. In addition, the logical layer contains helper entities such as network routers that aid other logical entities to fulfil their requirements. The *physical layer* is the bottom layer that represents the physical or basic existence that all entities have in the real world. The physical layer includes the physical components of systems and the environment including both tangible objects and electromagnetic radiation.

Every subject, object, relationship or piece of information, except abstract concepts, has a physical representation in addition to its existence at higher layers. Any activity carried out by a system is ultimately on behalf of a semantic subject, but the work must take place at the physical layer. However, higher layer entities

cannot be understood at the physical layer. For example, information is ultimately stored physically, but understanding involves knowing its meaning, purpose, and maybe other attributes such as its origin and correctness that can only be fully appreciated at a higher layer.

This is much simpler than Neumann’s eight-layered model, but it can still provide detailed analysis of systems by allowing each layer to have sublayers. For logical network communication, the best-known model is the seven-layer OSI network model [5]. We would use Tanenbaum’s simplified five-layer network model [6] as sub-layers of our logical layer with the link, network and transport layers as intermediate sub-layers and the upper application and lower physical sub-layers interfacing to the social and physical layers of our model respectively.

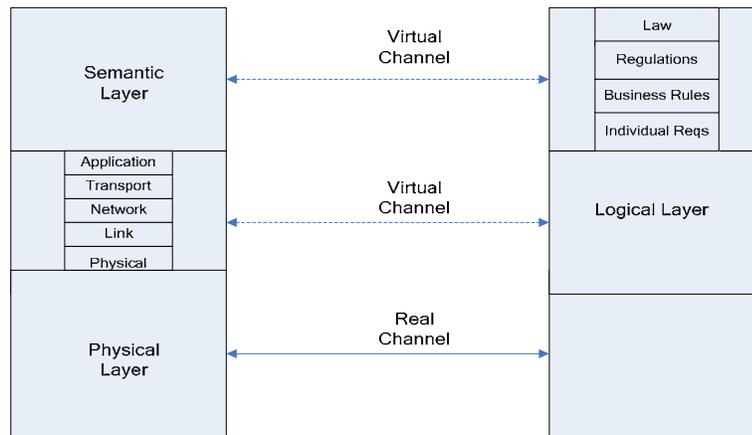


Figure 1-The three-layer model showing the sub-layers of the semantic and logical layers

Each layer has a separate concept of location and distance between entities. Higher layer entities also have a physical existence, so they are represented by different locations at each layer. The entities at each layer have different extents, dependencies and interactions and should meet the system requirements for that layer. A *channel* is an entity that carries flows of information and objects from one location to another at the same layer. Entities in different locations at the same layer use a channel to communicate. The channels at the higher layers are virtual, and must use a physical channel to communicate analogously to communication in the OSI network model (figure 1). The introduction of horizontal scope allows us to remove some of Neumann’s layers [2] such as the networking and middleware layers and represent them as horizontal communication channels for computer and application entities respectively.

A *virtual entity* is a logical entity that uses some controls to limit access, so that it can only be understood, accessed or used with special methods or knowledge such as using cryptographic keys. Application layer resources such as data and services can be virtualised by replicating them, which removes the reliance on protecting single locations and thereby avoids single points of failure. In addition, many systematic controls can be represented using virtual entities including Trusted Platform Modules (TPMs) and virtual machines.

2.2 Protection

All systems have a horizontal scope at every layer. Neumann [2] considers four conceptual locations for compromise at every layer; from outside, above, within and below. Protection from an external entity at the same layer requires horizontal controls, whilst protection from a higher layer entity requires a vertical boundary between the layers. Insiders should be constrained by partitioning the system with additional internal system boundaries they should not be able to breach. However, complete protection from insiders may not be possible, so these controls may use detection and recovery mechanisms such as auditing and redundancy, so that misuse of the system is detected and recovered from, rather than prevented. Replication is an effective defence against insider attacks by backing up data or providing standby services in separate locations that insiders cannot access. Some components that control the system must be trusted and so they should be made simple enough to assure. They should reside in an inaccessible location at the bottom of the system or use a secure control network to stop external interference. In conclusion, all entities, apart from unconditionally trusted entities, should be outsiders relative to one or more controls that moderate their use of the system.

The boundaries can be annotated with their access controls represented by logical predicates or probabilistic estimates of successful defence. All the controls should be analysed together to show that they meet the system policy. For example, a firewall could be annotated by the port numbers of protocols it blocks, and anti-virus software by the signatures of malicious code it detects. This is intended to deal with attacks by partitioning the defence at the network and application layers, but a new virus using an allowed protocol would breach the defence, unless there were additional controls.

3 Modelling Multi-layer Systems

3.1 Coordinate Representation

We can represent the location of logical entities in multiple dimensions as for physical entities. For example, physically separate entities can communicate securely by the creation of a virtual tunnel that can be represented in a higher dimensional space. The location of real entities is represented in real coordinates $(x, y \dots)$, whereas cryptographic entities have additional dimensions as well, which are represented by complex coordinates $(z, w \dots)$. The coordinates indicate the location where an entity can access the data, which is only possible if it has access to the correct keys, indicated by access to the right virtual coordinates, and it can reach the real location represented by the real coordinates. A specific example is an SSL connection, whose real components are the Internet addresses of the path taken, and the virtual component might be an integer identifying the cryptographic channel uniquely amongst current cryptographic sessions. Cryptographic protection offers weaker protection semantics than physical security, as data can be deleted or altered with access to the physical location or communication path alone, without having access to any keys.

The coordinate representation has many applications such as reasoning about possible breaches of security by attackers in different logical locations with different knowledge and abilities. Bigraphs are a more abstract topological representation of this idea that only retains the shapes of entities by discarding the location coordinates, which simplifies the analysis and, in addition have executable semantics. Both methods can represent Neumann's four conceptual locations for compromise more formally [2].

3.2 Bigraphs

Our model can be formalised using multi-level graphs with one level for each layer of our model. We propose formalisation using Milner's bigraph model [7] that can represent physical and logical levels and the interaction between them. It represents the semantic layer indirectly through its actions and effects at lower layers. The model is quite flexible having its origin as a unification theory for process calculi such as Petri nets and the pi-calculus that model communication, together with models such as the ambient calculus that handle physical movement. The model natively incorporates the structure and organisation of the physical and logical layers including their interaction and dependence on each other, which is not considered in most security models. We propose a novel use for modelling security architecture and apply it to critical infrastructure protection.

The bigraphical structure composes two graphs with one to represent logical communication and the other physical mobility. A bigraph can model the security architecture of systems, as security mechanisms can be represented as graph rewriting or reaction rules. The application of a rule in one direction represents the defence, which is reversed by the user to access the system. The defender's objectives can be defined by invariants of the bigraph that hold in secure states of the system, and certain reaction rules representing actions that should only be performed by the defender.

The system is represented by a bigraph with security requirements represented as constraints in the bigraph. Different types of attacker with various powers and locations can be represented using an attacker bigraph occupying a particular kind of node, having particular communication possibilities and accessing certain reaction rules. The model is executed to discover if the attacker can breach or inactivate security controls, access critical assets or reduce system functionality.

4 Critical Infrastructure Protection

There are many applications of our three-layer model as computer systems and physical objects must always be used to meet higher-layer organisational and personal requirements. One important application is modelling critical infrastructure, which are very complex systems that are impossible to analyse manually and the effects of failure could be devastating. These systems have large numbers of people with various degrees of physical or logical access to parts of the system such as buildings, equipment, computers and control systems. The large

horizontal extent at all layers may allow unauthorised remote access to computers, and access to insecure physical components. They can function in unexpected ways with remote effects and complex interaction between the layers with unpredictable consequences.

The model can faithfully represent both physical and logical attacks on control systems and communication links in critical infrastructure. We can formalise the representation using bigraphs and analyse the resulting model for vulnerabilities. It can model dependencies between components, attackers in various locations with various powers, and handle effects in remote locations and other layers. It can model hybrid attacks that use several layers and transitive attacks that operate in several stages. Controls can be proposed to avoid or mitigate undesirable effects by partitioning and isolating systems horizontally, and restricting changes of layer vertically.

In figure 2, the ellipses represent physical locations, whereas the arcs represent logical communication. The locations can be of different kinds, which may have different modes of interaction and communication. The outermost ellipse might be a building, the small circle might represent a computer or machine, and an intermediate size ellipse may represent a room (or possibly a network). The flexibility of the model is demonstrated by the example, as these ellipses could equally represent networks, computers and applications instead.

The physical movement of people and tangible objects is modelled by movement between ellipses, which is controlled by the kinds of the nodes and the available reaction rules. The arcs represent different types of communication, interaction and control including computer, telecommunication and power networks. Communication is controlled by the kinds of communicating entities, the type of the channels, and the available reaction rules. Entities are allowed to move over communication channels as well. A logical entity such as a user account (acting as a proxy for a person) may move over the logical link to the remote computer if it can log on. All links should be protected cryptographically or by enclosure within a secure physical boundary. This interaction between physical location and logical communication is represented natively using bigraphs, but not most formal models that discard location information.

The lower large ellipse represents a building containing a room holding a telecoms switch S that can be accessed and controlled through the administrator's workstation. A defence objective is that there should be no path from the outside to the switch S by either physical or logical means from unauthorised people or malware.

A user can access the administrator's computer remotely if he can use the correct key K required for authentication, which is indicated by the graph reaction rule as shown. The defence initially set up the requirement for external authentication using the reverse reaction rule. Figure 2 shows that the switch S can be accessed in multiple ways from outside both logically through transitive access to S via the administrator's workstation A, and physically by entering the building and then the room. It is also possible to represent hybrid attacks where both logical and physical accesses are combined. The link to the room containing S could be used to send a command to turn off the power supply for example. All these controls can be represented by the kinds of node, types of channel and available reaction rules, so the model can be executed to determine if an external attacker can breach any of the controls to interfere with S.

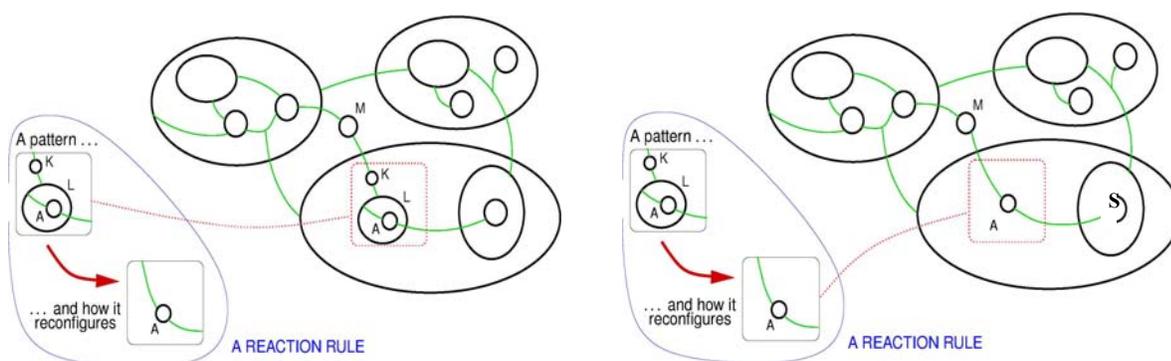


Figure 2 – Use of an authentication key K to remotely access an administrator's workstation A through the boundary L represented by a reaction rule (© Milner (2005) [7])

When compared to our coordinate representation, the diagram can be considered a flattening of the multidimensional structure into two dimensions. The physical entities such as buildings have height one, whereas the logical entities such as computers have height two. Communication is considered to take place at the highest layer of the communicating peers. The cryptographically protected communication channel is considered to occupy a fourth orthogonal dimension not accessible to real entities.

5 Conclusion and Further Work

Comprehensive protection against the different types of threat can be provided by multiple defensive controls that each create a boundary meeting different protection requirements. For example, it was suggested that further internal controls be used to protect against insiders, which allows the treatment of insiders and outsiders to be unified.

We demonstrated an informal three-layer model for modelling security architecture that allows us to reason about the structure and organisation of systems components and their interaction. A coordinate system was provided to represent the location of logical entities that allows the modelling of Neumann's four conceptual locations of attacks at all layers. We formalised the three-layer model using bigraphs and used it in the critical infrastructure example to show how systems can be compromised at the physical and logical layers including multi-layer attacks that use both.

We have used bigraphs to represent cryptographic primitives such as hash functions and digital signatures [8] and intend to use it to analyse Kerberos, which is a complex network security protocol that takes account of physical vulnerabilities such as insecure workstations as well as logical vulnerabilities.

We propose some extensions to the bigraph model to broaden its applicability and to model the security requirements of systems more faithfully. Intermediate layers can be introduced to model different layers of the network stack such as the network and application layer. Additional layers can also model the virtualisation of hardware or the operating system. Users interact with the virtual layer, which is translated to the real activity performed by the layer underneath. This sandwich layer allows, among other things, policy enforcement with additional security checks, or the virtual system acting differently to the underlying layer.

An important application is modelling the interaction between the control elements of a system and its functional components. The control space must interact with the rest of the system through physical proximity or at a distance through logical communication channels. For example, a hardware-based Trusted Platform Module (TPM) has its own separate components in a secure location performing computation with its own processor, communicating with its own dedicated buses and using its own physical storage. The TPM can be represented by a separate bigraph encapsulated within the complete system, which communicates through dedicated control channels to control access to the system resources.

References

- 1 Avizienis, A, Laprie JC, Randell B and Landwehr C, "Basic Concepts and Taxonomy of Dependable and Secure Computing", *IEEE Transactions on Dependable and Secure Computing*, vol 1, no 1 (2004).
- 2 Neumann, PG, "Practical Architectures for Survivable Systems and Networks" (2000), online at www.csl.sri.com/neumann.
- 3 Neumann PG, Parker D, "A Summary of Computer Misuse Techniques", *Proceedings of the 12th National Computer Security Conference*, (1989).
- 4 Howard JD and Longstaff TA, "A Common Language for Computer Security Incidents" (1998), Sandia National Laboratories, online at www.sandia.gov.
- 5 Day JD and Zimmermann H, "The OSI Reference Model", *Proceedings of the IEEE*, vol 71, pp 1334-1340, (1983).
- 6 Tanenbaum, AS, *Computer Networks (4th edition)*, Prentice-Hall, Upper Saddle River, New Jersey, (2003).
- 7 Milner R, "Bigraphs, a Tutorial", (2005), online at www.cl.cam.ac.uk/users/rm135.
- 8 Blackwell C, "Using bigraphs to represent cryptographic primitives", submitted to *ACM New Security Paradigms Workshop* (2007).

Using Bigraphs to Model System Architecture



Clive Blackwell
Information Security Group
Royal Holloway, University of London
Email: C.Blackwell@rhul.ac.uk



Presentation Outline

- The layered security model
- Introduction to bigraphs
- Modelling critical infrastructure with bigraphs
- Further work

Existing Systematic Models

Basic Concepts and Taxonomy of Dependable and Secure Computing

Avizienis, Laprie, Randell and Landwehr

Longstanding work to Classify the important aspects of dependability including security

Practical Architectures for Survivable Systems and Networks

Neumann and Parker organised systems into eight layers for security analysis

The external environment, user, application, middleware, networking, operating system, hardware and internal environment

A Common Language for Computer Security Incidents

Longstaff and Howard present a classification system for network security incidents

Shows the different types of entity involved in an attack and their interrelationships

15/05/2007

CSIIRW

3

The Layered Security Model

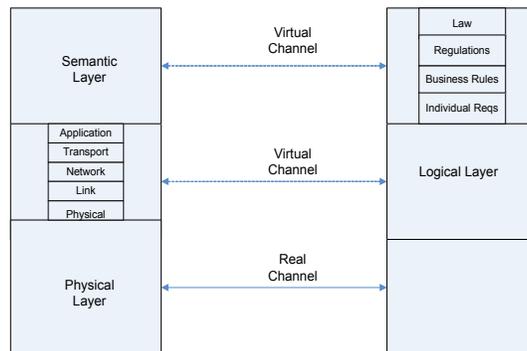
- Systems are modelled in a three-layer hierarchy
- The *semantic* or *conceptual layer* is the top layer
 - Includes people, organisations and system requirements
- The *logical layer* is the intermediate layer
 - Contains intangible entities including data and software that are stored and processed on computers
- The *physical layer* is the bottom layer
 - Represents the physical existence that all entities have in the real world
 - Includes both tangible objects and electromagnetic radiation

15/05/2007

CSIIRW

4

Three-layer model with sub-layers



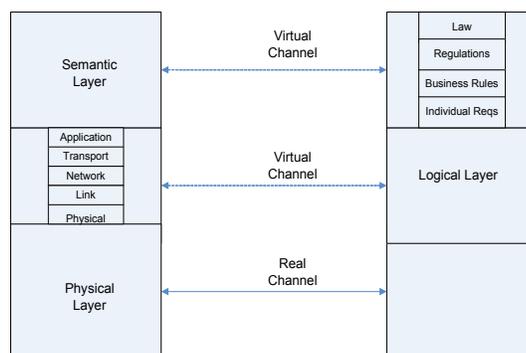
- Each layer can have sub-layers for detailed analysis
- We use Tanenbaum's five-layer network model as sub-layers of our logical layer
 - The link, network and transport layers are intermediate sub-layers
 - The upper application and lower physical sub-layers interface to the social and physical layers of our model respectively

15/05/2007

CSIIRW

5

Locations and communication



- Each layer has a separate concept of location and distance between entities
- Entities at each layer have different extents, dependencies and interactions
- Entities in different locations use channels to communicate
 - Channels at higher layers are virtual, and must use a physical channel to communicate

15/05/2007

CSIIRW

6

Modelling protection requirements

- All systems have a horizontal scope at every layer
- Neumann considers four conceptual locations for compromise at every layer
- From outside, above, within and below
 - Protection from an external entity at the same layer requires horizontal controls
 - Protection from a higher layer entity requires a vertical boundary between the layers
 - Insiders should be constrained by partitioning the system with additional internal system boundaries they should not be able to breach
 - Some components that control the system must be trusted and so they should be made simple enough to assure

Conclusion - All entities, apart from unconditionally trusted entities, should be outsiders relative to one or more controls that moderate their use of the system.

15/05/2007

CSIIRW

7

Representing the 3-layer model using bigraphs

- Any system represented in our model can be transformed into a bigraph
- Use the two levels of bigraphs to represent the lower two layers of our model
 - Physical and logical layers are directly modelled
 - Locations and communication channels can be physical or logical
 - Represented by different types
 - Semantic layer is represented indirectly through lower level activities and effects
- Vulnerabilities are never removed by the application of security mechanisms, but are transformed into other vulnerabilities
 - All vulnerabilities have a physical or logical location or channel
 - The protective mechanism and the channel between the protection and the resource is vulnerable
 - Directly modelled by reaction rules in bigraphs, but not considered in most security models
 - Keys are stored in locations that must be protected with additional controls
 - Cryptographically protected resources exist in a virtual location encapsulated within special nodes that represent the protection
- Attackers have physical and logical scope and powers that can be represented by bigraphs
 - Sites represent their location
 - External names represent their potential communication
 - Rewriting rules specify their powers

15/05/2007

CSIIRW

8

Bigraph Purpose

- Bigraphs are graphs with two constituent graphs representing locality and connectivity separately
- They are an attempt at a unifying theory for program semantics
 - Have been shown to capture the semantics of the π -calculus, Petri nets and mobile ambients faithfully
- Important aid to understanding systems with both physical and logical aspects
 - Ubiquitous systems
- Understanding systems at multiple layers is essential for security

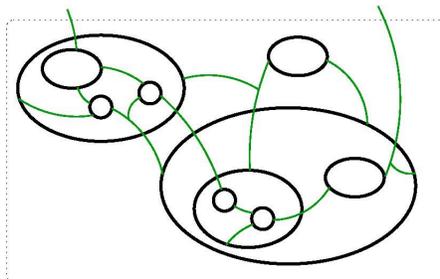
This introduction to bigraphs is based on “Axioms for bigraphical structure” by Robin Milner

15/05/2007

CSIIRW

9

Bigraph description



- The ovals are nodes of the bigraph, which are the common component of the underlying place and link graphs.
- Place graph edges are shown implicitly
 - The nodes are nested inside each other to represent placing one entity inside another
- Link graph edges are shown explicitly
 - Each node has ports which may be linked to other nodes which models communication
- The linking and the placing of nodes is independent, shown by the way links cross node boundaries in the diagram
- Each external link (shown emerging from the top in the diagram) can be joined to some link of a host bigraph to model external communication

15/05/2007

CSIIRW

10

Place Graph

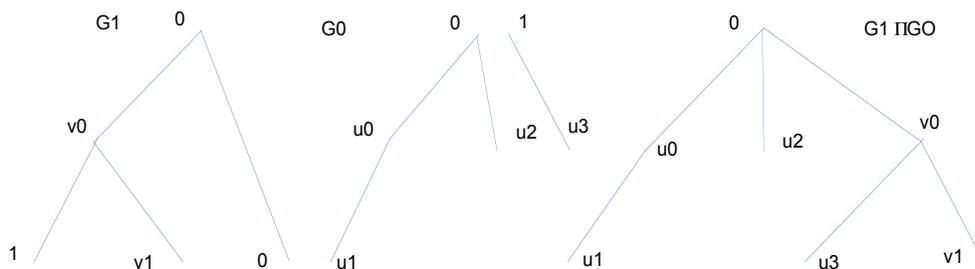
- A place graph $G = (V, \text{ctrl}, \text{prnt}): m \rightarrow n$ has:
 - An inner width m and an outer width n , both positive integers
 - $\text{ctrl}: V \rightarrow \kappa$ represents the kind of nodes
 - A parent map $\text{prnt}: m \cup V \rightarrow V \cup n$
- Represents n locations with a forest of n trees
- Represents m sites where other bigraphs can be placed
- The prnt map is the usual parent function for trees
 - Represents containment
- Places can be real or virtual locations
 - Keys, files and programs inhabit virtual locations

15/05/2007

CSIIRW

11

Composing Place Graphs



- Each root of the place graph G_0 is planted in a site of G_1
 - Identification points disappear in the composition
- If $G_i = (V_i, \text{ctrl}_i, \text{prnt}_i): m_i \rightarrow m_{i+1}$ ($i = 0, 1$) be place graphs,
- $G_1 \circ G_0 = (V, \text{ctrl}, \text{prnt})$ has $\text{prnt} = (\text{Id}_{V_0} \cup \text{prnt}_1) \circ (\text{prnt}_0 \cup \text{Id}_{V_1})$.
 - $\text{prnt}(p) = \text{prnt}_0(p)$ if $p \in m_0 \cup V_0$ and $\text{prnt}_0(p) \in V_0$
 - $\text{prnt}(p) = \text{prnt}_1(m)$ if $p \in V_0$ and $\text{prnt}_0(p) = m \in m_1$
 - new parent node in inhabitant graph
 - $\text{prnt}(p) = \text{prnt}_1(p)$ if $p \in V_1$

15/05/2007

CSIIRW

12

Link Graphs

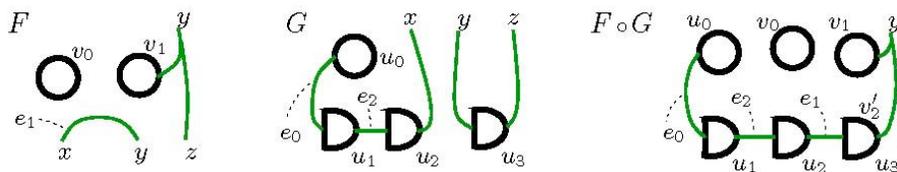
- A link graph $G = (V, E, \text{ctrl}, \text{link}): X \rightarrow Y$
 - Finite sets X of inner names, Y of outer names
 - Function $\text{link}: X \cup P \rightarrow E \cup Y$ called the link map.
 - The inner names X and ports P are the *points* of G , and the edges E and outer names Y its *links*.
 - The link map connects the attachment points to a communication channel
 - The interior link graph has a communication pattern that is enabled by the exterior link graph through the binding between interfaces
- We draw a link graph's inner names below it, and its outer names above
- Outer names can represent local or global communication
 - An outer name is an *open* link, an edge is a *closed* link
- The link graph can represent point-to-point or group communication
 - The link map is a hypergraph, so 'edges' can connect several points

15/05/2007

CSIIRW

13

Composing link graphs



- The outer names of the interior link graph link to an exterior link graph via its corresponding inner names
- Let $G_i = (V_i, E_i, \text{ctrl}_i, \text{link}_i): X_i \rightarrow X_{i+1}$ ($i = 0, 1$) be two link graphs, then $G_1 \cup G_0 = (V, E, \text{ctrl}, \text{link})$ has
 - $\text{link} = (\text{Id}_{E_0} \cup \text{link}_1) \circ (\text{link}_0 \cup \text{Id}_{P_1})$
 - $\text{link}(p) = \text{link}_0(p)$ if $p \in X_0 \cup P_0$ and $\text{link}_0(p) \in E_0$
 - $\text{link}_1(x)$ if $p \in X_0 \cup P_0$ and $\text{link}_0(p) = x \in X$
 - outer name of G_0 links to edge, which makes it local to composed graph or to outer name of G_1 , which potentially makes it global
 - $\text{link}_1(p)$ if $p \in P_1$

15/05/2007

CSIIRW

14

Bigraph Definition

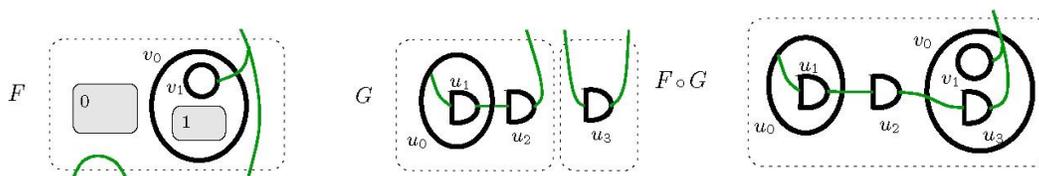
- Define the composition of bigraphs in terms of the underlying operations on their constituent graphs
- An *interface* $I = \langle m, X \rangle$ consists of
 - A positive integer m called a *width*, representing places
 - a finite set $X \subset \xi$ called a *name set*
- A *bigraph* takes the form $G = (V, E, \text{ctrl}, G^P, G^L): I \rightarrow J$
 - The interfaces $I = \langle m, X \rangle$ and $J = \langle n, Y \rangle$ are its *inner* and *outer* faces
- The map from I to J is determined by the underlying place and link graphs
 - $G^P = (V, \text{ctrl}, \text{prnt}): m \rightarrow n$ a *place graph*
 - $G^L = (V, E, \text{ctrl}, \text{link}): X \rightarrow Y$ a *link graph*

15/05/2007

CSIIRW

15

Composing Bigraphs



- The component place and link graphs are our previous examples
- The composition of the two bigraphs are formed from the combinations of the place graphs and link graphs, which have orthogonal operations
- The sites in F indicate where the roots of an inhabitant bigraph G can be placed
- Then the outer names of G are linked to the corresponding inner names of F
- Note that G can be inserted two ways into F
- The two sites of G could represent a communication pattern that is implemented when inserted into F
 - Some of the nodes in G could be keys to protect communication over F , which could represent an insecure medium such as the Internet

15/05/2007

CSIIRW

16

Reaction Rules

- Different controls can participate in different *reaction* or *rewriting rules*
- Each rule consists of a precondition, which may be transformed into a post-condition wherever it occurs
 - Both of these conditions are bigraphs
- Atomic nodes do not have internal nodes
- Complex nodes can be active or passive
 - The control is *active* if reactions can occur inside them
 - It is *passive* if no internal reaction is allowed
 - The only method of reaction possible is when the passive nodes are destroyed

15/05/2007

CSIIRW

17

Critical Infrastructure

- Very complex systems that are very difficult to analyse manually
 - Large numbers of physical and computational entities and communication paths, and people with various powers
 - Large horizontal physical and logical scope
 - Allows pervasive access
 - Subject to both physical and logical attacks on resources, control systems and communication links
 - Intractable problem to avoid attacks
 - Large and unmitigated number of vulnerabilities
 - Goals of model in critical infrastructure protection
 - Finds vulnerabilities
 - Unexpected dependencies difficult to discover manually
 - Suggest remediation measures
 - Remove critical vulnerabilities
 - Continue in face of an attack
 - Avoid catastrophic failure modes
 - Helps in design of system
 - Avoid critical failure
 - Partition systems to limit damage of successful attack

15/05/2007

CSIIRW

18

Critical Infrastructure (2)

- The bigraph representation of our three-layer model can faithfully represent critical infrastructure. It can:
 - Check objectives are satisfied
 - Can give assurance of protection of critical assets or indicate critical vulnerabilities
 - Model architectural protection using defence-in-depth
 - Model dependencies between components
 - Discover linkage between layers
 - Can discover attacks that operate and have effects at several layers
 - Handle remote effects and dependencies
 - Including transitive attacks that operate in several stages
 - Represent attackers in various locations with various powers
 - Including insider attacks
 - Model different types of network and the interaction between them
 - Model different scenarios by changes to attacker and defender

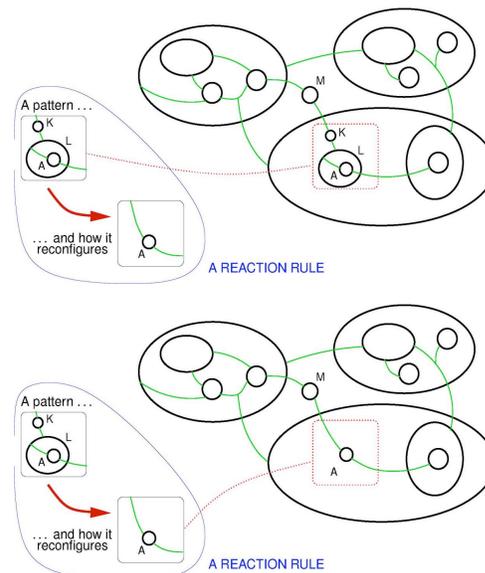
15/05/2007

CSIIRW

19

Critical infrastructure example

- Nodes of different types
 - Outermost ellipses are buildings
 - Circles are computers or machines
 - Intermediate size ellipses may represent rooms
- Representing requirements
 - The lower large ellipse represents a building containing a room with a telecommunication exchange or controller for an electricity substation S that can be accessed and controlled through the administrator's workstation
 - One defence goal is that there should be no path from the outside to S by either physical or logical means from unauthorised people or malware
- Cryptographic channels
 - The administrator's computer can be accessed remotely if the correct key is used for authentication, which is indicated by the graph reaction rule as shown



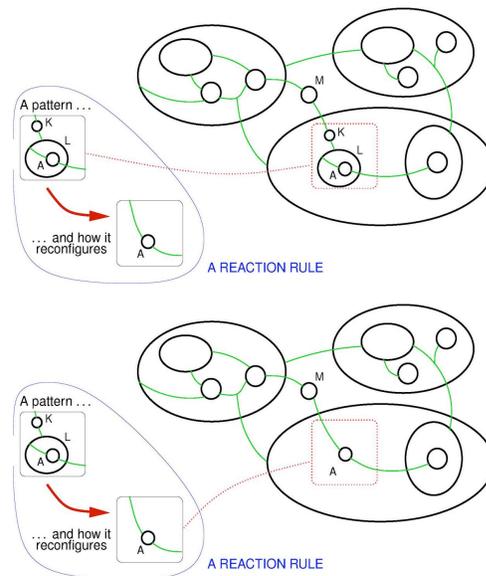
15/05/2007

CSIIRW

20

Critical Infrastructure example (2)

- A user can access the administrator's workstation remotely when he authenticates by proving he has access to the key K represented in the reaction rule
- The defence initially set up the authentication mechanism
 - Represented by the reverse of the reaction rule with the arrow in the opposite direction
- S can be accessed in multiple ways from outside
 - Logically through transitive access via the administrator's workstation A
 - Physically by entering the building and then the room
 - Via a multi-layer hybrid attack where both logical and physical accesses are combined
 - The link to the room containing S could be used to send a command having a physical effect such as cutting the power supply to S
- All these controls can be represented using bigraphs and model checking could determine if the attacker can breach any of the controls to interfere with S



15/05/2007

CSIIRW

21

Conclusion

- We demonstrated an informal three-layer model for modelling security architecture
- We formalised the model using bigraphs
- We used bigraphs to analyse a simple critical infrastructure example
- We suggested how the model unifies the treatment of insiders and outsiders
- We have used the model to represent cryptographic primitives (NSPW 2007 submission)
- We are investigating Kerberos, a complex network security protocol, that considers both physical and logical vulnerabilities

15/05/2007

CSIIRW

22

Model References

Avizienis, A, Laprie JC, Randell B and Landwehr C, “Basic Concepts and Taxonomy of Dependable and Secure Computing”, *IEEE Transactions on Dependable and Secure Computing*, vol 1, no 1 (2004).

Neumann, PG, “Practical Architectures for Survivable Systems and Networks”, (2000), online at www.csl.sri.com/neumann.

Howard JD and Longstaff TA, “A Common Language for Computer Security Incidents”, Sandia National Laboratories (1998), online at www.sandia.gov.

15/05/2007

CSIIRW

23

Bigraph References

Blackwell C, “Using bigraphs to represent cryptographic primitives”, submitted to ACM New Security Paradigms Workshop (2007).

Milner R (2005), “Axioms for bigraphical structure” (revised version of Cambridge Computer Laboratory Technical Report 581) at www.cl.cam.ac.uk/users/rm135.

Milner R (2005), “Bigraphs, a Tutorial”, at www.cl.cam.ac.uk/users/rm135.

15/05/2007

CSIIRW

24

Early Detection and Containment of Worm Epidemics

Tom Chen
Dept. of Electrical Engineering
Southern Methodist University
Dallas, TX 75275
214-768-8541
tchen@enr.smu.edu

Abstract - In the past few years, a number of large-scale worm outbreaks – such as Code Red, Slammer, Blaster – have caused widespread damage and raised concerns that a future outbreak could spread through the Internet faster than current defenses could effectively contain it.

Current detection of worm outbreaks is based mainly on signatures. Signatures allow the most accurate detection but have two serious drawbacks: worms without known signatures will evade detection, and signatures for a new worm requires hours to develop, test, and deploy. Given that a fast worm epidemic might be finished before a signature is available, worm detection must also use behavior-based detection.

We are studying the behavior of worm outbreaks through a “community of households” epidemic model. A household represents an autonomous system in the Internet. We are developing a novel Web-based worm simulator for the community of households. A proof-of-concept prototype of the Web-based simulator written in Java and Perl is available on the Web, although many features have not been implemented yet.

Network simulators are typically written as standalone programs. Users are required to download and install a copy of the simulator program on their computers. This approach has a few drawbacks: programs are platform dependent and often require programming knowledge; graphical user interfaces can be varied and confusing; users are responsible for downloading and installing the latest version.

To the best of our knowledge, our worm simulator is the first written as a Web application. The common Web browser is the interface for invoking the simulator and submitting input parameters. The simulator runs on a Web server, and outputs simulation results back to the browser. This approach offers a number of advantages over standalone simulators: the Web browser is familiar and easy to use interface; it is platform independent (since browsers run on all platforms); users do not have to be concerned with downloading and installing any code; users always access the most current version of the simulator program. An additional advantage is the capability to store simulation results on the server and easily share results between different users.

Early Detection and Containment of Worm Epidemics

Tom Chen
SMU, Dept of Electrical Engineering
Dallas, Texas 75275
tchen@engr.smu.edu
www.engr.smu.edu/~tchen

Early Detection Systems

- Worm outbreaks can spread quickly, e.g., Slammer
- Early detection and warning systems correlate observations from distributed sensors to automatically detect new worm outbreaks, even unknown worms
 - Symantec's DeepSight Threat Management System
 - Internet Storm Center operated by SANS and Incidents.org
- Worm detection depends mostly on signatures

Early Detection and Containment

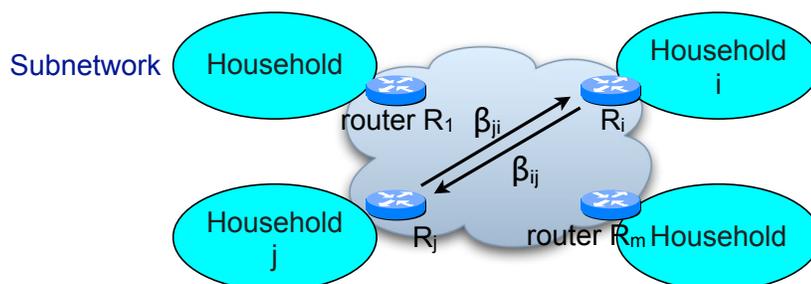
- Signatures allow accurate detection but
 - Worms without signatures may evade detection
 - Signatures for a new worm can take hours to develop
- Behavior-based (anomaly) detection is useful for detecting unknown worms
- After detection, outbreaks can be contained by quarantine (blocking) or rate throttling (slowing down)
- Outbreak behavior can be studied by epidemic modeling and simulation

TC/5-15-07/CSIIRW

SMU Engineering p. 3

Community-of-Households Model

- Population consists of m households (autonomous systems)
 - Hosts are initially *susceptible* (S) state, then change to *infective* (I) state and *removed* (R) state
 - β_{ij} = infectious contact rate from household i to household j (different than intra-household rates)



TC/5-15-07/CSIIRW

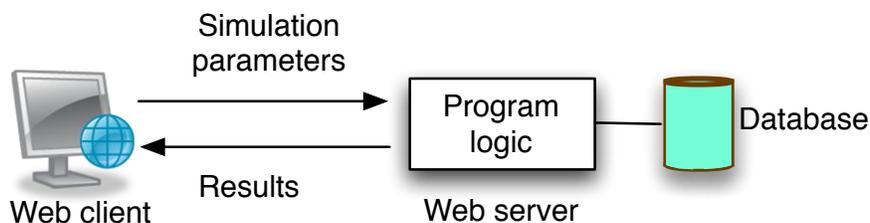
SMU Engineering p. 4

Web-based Simulator

- In addition to mathematical analysis, we are developing Web-based worm simulator
 - Several existing worm simulators are written as applications, requiring users to download and compile
 - Simulators are platform dependent
 - Each copy of simulator and simulation results are tied to a physical machine
 - Users are responsible for maintaining and updating

Web-based Simulator

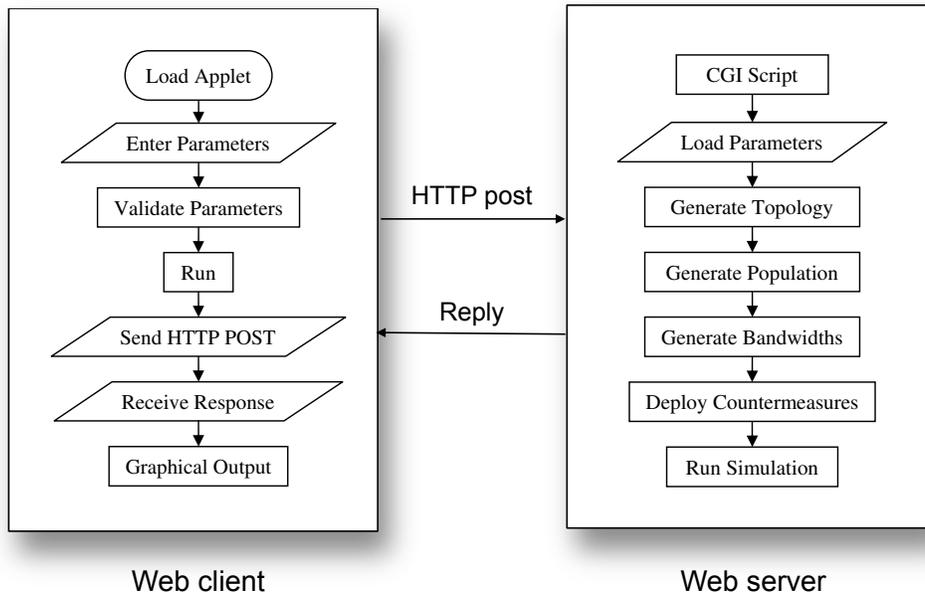
Client-server architecture separates GUI from program logic



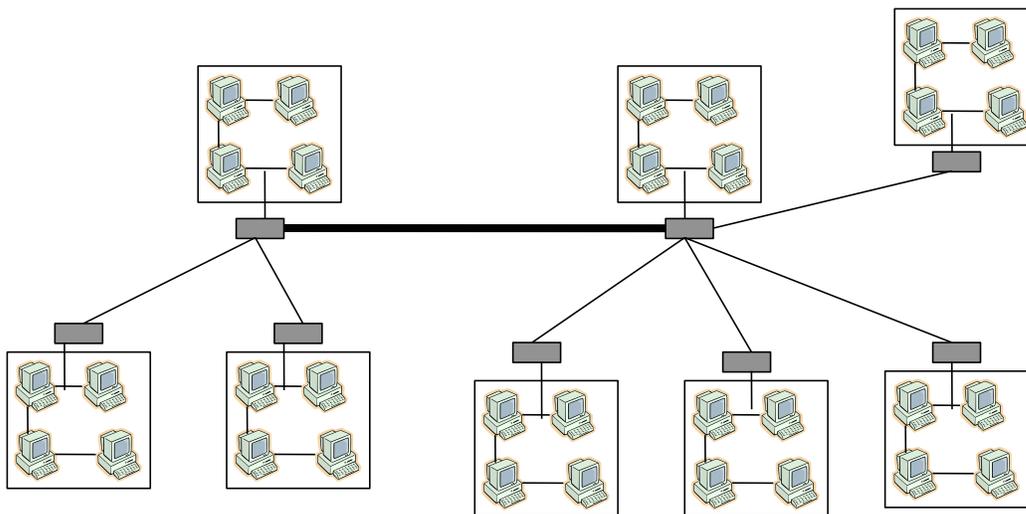
- Web browser provides familiar, consistent, user-friendly GUI
- Users do not have to download and maintain their own simulators

- Web server provides location-independent and platform-independent simulation
- Simulation results can be shared easily

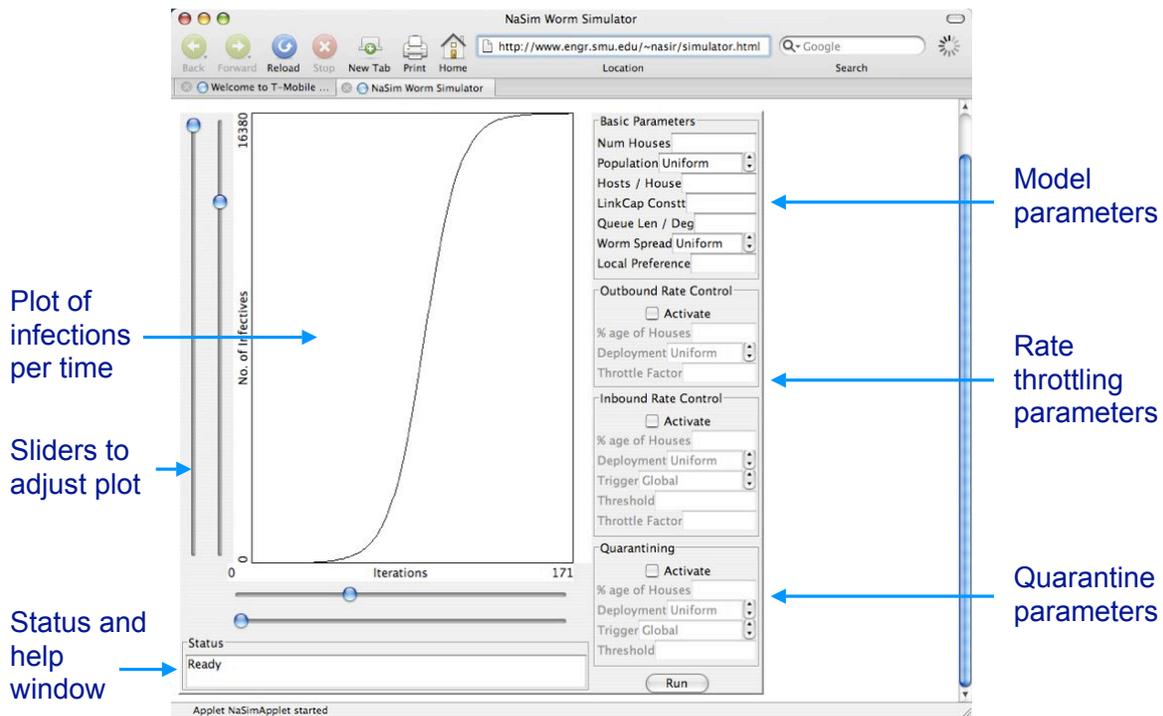
High Level Design



Simulated Network Example



Prototype GUI



TC/5-15-07/CSIIRW

SMU Engineering p. 9

Prototype Current Features

- GUI is Java applet for data entry, data validation, context-based help, graphical display of results
- CGI script is python program to pass input parameters to core simulator program running on server back end
- Core simulator uses U. Michigan's Inet 3.0 to generate network topology
- Simulation results are stored on server (with unique identifiers) for later retrieval or sharing

TC/5-15-07/CSIIRW

SMU Engineering p. 10

Issues and Future Features

- Topology generation and simulation time slows down drastically with model size (number of households)
- Server can keep track of multiple simultaneous simulations (by job scheduling) but number is currently limited to prevent overwhelming
- Currently static routing (shortest routes computed by Dijkstra's algorithm) but dynamic routing more realistic
- Rate throttling not fully implemented yet

Managing Multiple Perspectives on Trust

*Dr. Clifford Neuman
Information Sciences Institute
University of Southern California
(<http://clifford.neuman.name>)*

Trusted computing provides methods for software components to establish confidence in the code with which they communicate. While commonly used for digital rights management, the same underlying mechanisms can be used to protect users from untrustworthy service providers and to provide strong isolation for critical functions running on common infrastructure

This abstract discusses ongoing work to develop trusted computing architectures and policy models supporting multiple perspectives on trust. The TrustView Security Architecture enables strong separation for critical functions. By moving some basic support for separation into the network infrastructure, the architecture enables limited performance isolation across function. The trusted computing reference monitor mediates requirements and obligations for each software component providing mutual protection to all involved.

The TrustView architecture leverages trusted computing technologies to protect multiple, possibly competing, interests within a system, including the interest of the end user against abuse by the companies with which they interact. The architecture supports strong isolation of functions at a coarse level of granularity. Such policies are easier to understand and can be readily implemented by virtualization technologies [3]. Such coarse grained policies are specified in a less dynamic way than traditional fine grained policies: the allowable flow of information between software components is specified through the creation of virtual systems [2]. Protection is provided by limiting the flow of information across virtual system boundaries.

Most systems today allow programs to run in two modes, kernel and application. A system's Trusted Computing Base (TCB) resides in the hardware and kernel, and user applications run untrusted. Early systems like Multics [1] provided more structure, with innermost rings composed of compact, heavily trusted code, and successively less trust required as one moves to outer rings. The problem with this model when it is applied to distributed systems is that it assumes a fictitious hierarchy of trust. Software is either trusted or not, and the software implements whatever policy was decided by its implementer.

In distributed, loosely managed systems like today's Internet, certain processes may be more trusted by some entities, while other processes are more trusted by others. This was illustrated in the past by the installation of root-kits on the PC's of users who played CD's produced by Sony. To Sony, the user's PC was not trusted, but their own software was (trusted does not mean worthy of trust). The users soon discovered that it was Sony who should not be trusted. For a security architecture to protect all users it must provide mechanisms to deal with such mutual suspicion.

Surprisingly, by weakening our trust requirements for modules that are certified for use in a trusted computing environment – i.e. if we accept and certify even partially trusted components – then we can derive significant security benefit for our systems and networks as a whole. We can then use the trusted computing infrastructure to develop a virtual system abstraction that defines policies for interconnecting and managing the flow of information between instances of software modules running in a distributed system. Considering partial trust strengthens security

because modules that providers considered trusted (for example, those used by DRM systems) can be reclassified as only partially trusted since they might not really be worthy of trust from the perspective of the end user.

The TrustView Security Architecture (TVSA) allows software components and protected resources to be placed in overlapping rings of protection. Collections of functions and applications are associated with “virtual systems” that define views of trust from a particular perspective. A process and its associated persistent data may reside in different rings within different virtual systems so that it is considered more trusted by some and less trusted by others. At run time, information does not flow across ring boundaries except through processes that are members of multiple rings. Virtualization techniques are used to provide strong separation between processes running in different virtual systems. Individual processes mediate the flow across the boundaries which they span. When a process joins a virtual system, obligations are negotiated which constrain the process’s ability to participate in other virtual systems based in part on attestation of the process’s ability to protect the flow of information across virtual system boundaries.

In our architecture, the security attributes of applications that communicate across a network (and within individual hosts) are negotiated and communicated by code in the operating system and network stack on the communicating processors. Applications run in ‘virtual systems’, distributed across network

nodes, whose policies for membership are specified during the installation of an application, and managed external to the application. These virtual systems correspond to the rings in our system architecture.

The components of a virtual system are the hardware, OS, and applications on participating nodes. The level of trust placed in each of these components varies according to perspective: thus from the perspective of a node running part of a virtual system, those components running locally may be more trusted, while from the perspective of a server, those same elements of the system may be less trusted.

BIOGRAPHY

Clifford Neuman is director of the Center for Computer Systems Security at the Information Sciences Institute (ISI) of the University of Southern California (USC), and a faculty member in the Computer Science Department at USC. He earned a Bachelor's degree at the Massachusetts Institute of Technology and subsequently worked for Project Athena. He received M.S. and Ph.D. degrees from the University of Washington. Dr. Neuman conducts research in distributed systems, computer security, and electronic commerce and is the principal designer of Kerberos authentication system, the NetCheque and NetCash systems, and the Prospero Directory Service. Dr. Neuman’s current research is focused on trusted computing architectures that support multiple views of trust.

REFERENCES

- [1] M. D. Schroeder and Jerome H. Saltzer. *A hardware architecture for implementing protection rings*. Communications of the ACM, 15(3):157-- 170, March 1972.
- [2] B. Clifford Neuman, *The Virtual System Model: A Scalable Approach to Organizing Large Systems*, Ph.D. Thesis, University of Washington, Department of Computer Science and Engineering Technical Report 92-06-04, June 1992.
- [3] Paul Barham, Boris Dragovic, Keir Fraser, Steven Hand, Tim Harris, Alex Ho, Xen and the Art of Virtualization (2003) .Proceedings of the ACM Symposium on Operating Systems Principles. 2003.

Related information and updates to this paper may be found at:
<http://clifford.neuman.name/publications/2007/200705-neuman-csiirw-managing-multiple-perspectives-on-trust/>



Managing Multiple Perspectives on Trust

Clifford Neuman

Director, USC Center for Computer Systems Security

<http://clifford.neuman.name>

USC Viterbi
School of Engineering

Copyright © 2007 Clifford Neuman

UNIVERSITY OF SOUTHERN CALIFORNIA

**INFORMATION
SCIENCES
INSTITUTE**

Cyber Security and
Information Infrastructure
Research Workshop

Oak Ridge National
Laboratory

May 15, 2007



Today's Systems are Weakly Managed

Our computing environment is federated

- Assets managed by different organizations
- Many assets hardly managed at all (home machines)
- There are natural conflicts of interest in security policies
- Assessment of trustworthiness based on observation and shared reputation

USC Viterbi
School of Engineering

Copyright © 2007 Clifford Neuman

UNIVERSITY OF SOUTHERN CALIFORNIA
INFORMATION SCIENCES INSTITUTE

Trust has meaning in context

Trust has meaning only in a particular context

- A system is not trusted absolutely, but instead it is trusted to operate in a particular way that is dependent upon its intended purpose.
- For a system to be secure, we must consider the different functions, and manage the contexts associated with each function.
- We need basic security functions that can provide separation for each context, allowing a trusted virtual system to be established for each context.
- Finer-grained access control can then be supported within each context.

The Focus of Trusted Computing

Recent work emphasizes mechanism.

- How to provide attestation, isolation, and secure storage.
- Policy is understood in support of the mechanism.

But the mechanism must support policy.

- Policy focus has been limited to applications like DRM and NAC.
- These applications see a system as trusted or not.
- We need to understand how to define and enforce understandable policies that better model real systems that support multiple views on trust.

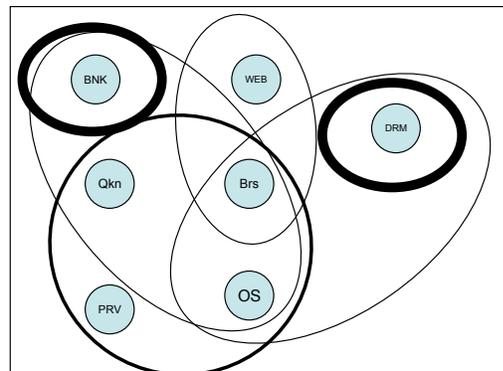
The Trust View Security Architecture

TVSA Policies are separated into:

- First level provides coarse-grained authorization
 - Basic Policies of separation supported by trusted computing functions of attestation of components, isolation, and secure storage.
 - Almost capability like
 - Based on being in the right virtual system.
- Second level enforces fine-grained policies
 - Supported by the trusted components within a virtual system.
- Precomputed policies for managing virtual systems
 - Say how pieces fit together.
 - Trust, ability, and obligation negotiated in advance.

The Trust View Security Architecture

**Rings represent
Precomputed policy
Virtual System
identifiers used to
enforce simple
policies.
Fine grained policies
enforced by the
individual
components
embedded within the
rings to the right.**



Once a Virtual System Formed

Accepted components have access to resources within the virtual system.

- But they have agreed to limits on what they can do.

How to Allow Flow Across Boundaries

Some components trusted to make fine grained decisions which allow data to flow across VS boundaries.

- Component is in multiple virtual systems.
- Data flows to component, in one VS.
- Data flows out of component in other VS.
- Component decides where data can flow.

Joining of Virtual Systems Limited

May not be allowed to join certain other virtual systems.

- Could require approval by other members
- Might carry a policy that says what other components can join.
- Might allow joins if component is known to provide controls on cross VS information flow.
- These are the pre-computed policies that determine what policies are enforced by the basic control mechanisms.

These limits do not apply to new instances

- They can form new Virtual Systems
- But these components do not gain ability to move information across VS boundaries.

Fine Grained Limits Enforced by Component Itself

The negotiation phase required assurances that the component could and would enforce those limits.

Less trusted components end up encapsulated in components that will provide the enforcement.

What Kinds of Policy Work Best

Can standard policy templates be created that correspond to the intrinsic policies that people expect, corresponding to common business, personal, government, or national security interactions.

Can these “templates” be used to structure virtual systems around particular functions that are commonly used in distributed systems.

Many Applications

DRM (or EDRM)

- But protect not just the content owner, also the systems on which the data is accessed.

Lampson’s Red – Green Network

- But really a rainbow of color.
- Examples are NAC, secure VPN from external machines.

SCADA Applications

- Push 1st class of rules into network infrastructure
- Get performance isolation

Toward Mitigating Denial of Service Attacks in Power-Constrained Sensor Networks

O. Arazi^{1,2}, H Qi¹

¹Department of Electrical and Computer Engineering
University of Tennessee, Knoxville, TN 37996-2100

²Cyberspace Sciences & Information Intelligence Research Group (CSIIR)
Oak Ridge National Laboratory, Oak Ridge, TN 37831-6418

Abstract—The challenging characteristics of sensor nodes, including the constrained resources, the ad-hoc nature of their deployment and the vulnerability of wireless media, pose a need for unique security solutions. The advantages of Public Key Cryptography (PKC) for sensor network security are widely acknowledged and include resilience, scalability and decentralized management. Recent work has indicated that PKC is feasible in the wireless sensor network (WSN) environment, paving the way for many new security services and opportunities. However, the computational effort involved in performing PKC operations remains substantial. From an energy consumption perspective, it is imperative that the processing and communication resources be utilized only when required. A malicious party attacks a sensor node by repetitive requests to establish a key, the resources of the attacked node can be exhausted quite rapidly. In this paper, we present an RSA-based framework for combating DoS attacks in WSN by ensuring that the malicious party will exhaust its resources prior to exhausting those of its counterparts. Under the proposed approach, the mathematical operations performed by the malicious party require two or three orders of magnitude more resources than those required by the attacked party.

I. INTRODUCTION

The sensor network, as a network of embedded sensing systems, has been studied extensively since the late 90s. Considerable efforts have been directed towards making them trustworthy [1], [2]. This is particularly true in health and military applications, where critical information is frequently exchanged among sensor nodes through insecure wireless media. In every application, the security of the system, both in terms of safeguarding against malicious attacks and resilience under malfunction, is a vital component. Although the area of network security has been studied for decades, the many unique characteristics of sensor networks have traditionally rendered direct application of existing solutions impractical. A fundamental requisite for security, other than providing data confidentiality and authentication, is Denial of Service (DoS) prevention. The computational effort involved in performing PKC calculations is substantial. From an energy consumption perspective, it is imperative that the processing and communication resources be utilized only when required. To that end, PKC implementations are more vulnerable to DoS attacks, when compared to traditional security methods that require less resources. In particular, if a malicious party attacks a sensor node by futile repetitive requests to establish a joint

secret key, the resources of the attacked node will be exhausted quite rapidly. To address this issue, we present a public key cryptographic approach for mitigating DoS attacks in sensor networks.

II. ECC-BASED KEY GENERATION AND AUTHENTICATION

A. Notation and Formulation

We begin by reviewing the foundations for ECC-based key generation and authentication, as introduced by the authors in [3]. Our mathematical foundations rely on ECC cryptographic techniques pertaining to operations over a finite group of points in which the discrete log problem applies. In order to describe the formalism for efficient two-node Diffie Hellman (DH) key generation, we must first define some notations and terminologies. A group-point is hereby denoted by a capital letter in bold font and a scalar will be presented in regular lowercase letters. Multiplication of a point by a scalar (e.g., $s \times \mathbf{P}$) will be referred to as an exponentiation, where s is the exponent. The intractability of a discrete log operation means that given the points \mathbf{P} and $s \times \mathbf{P}$, the complexity of finding s is exponential. The following notations are employed throughout the remainder of this paper: \mathbf{G} - a generating group-point, used by all relevant nodes; $ord\mathbf{G}$ - the order of \mathbf{G} . (exponents are calculated *modulo* $ord\mathbf{G}$); d - the CA's private key; \mathbf{R} - the CA's public key (where $\mathbf{R} = d \times \mathbf{G}$); x_i - the *private* key of node i served by the CA; \mathbf{U}_i - the *public* key of a node i served by the CA; ID_i - the identification details, or attributes, of node i ; $H(v, \mathbf{W})$ - a scalar obtained by performing a hash transformation on the scalar v and group point \mathbf{W} ; h_i - a random 160-bit scalar generated by the CA (for the purpose of calculating x_i); N_i, N_j - sensor nodes i and j , respectively.

B. Keys Issued to Nodes by the CA

The private and public keys discussed here are issued by the CA to all nodes in the network. We will begin our discussion by focusing only on keys issued to N_i . As indicated above, the CA holds a pair of keys (private (d) and public (\mathbf{R})). By using d, ID_i, h_i , a hash function and \mathbf{G} , the CA establishes the pair of private and public keys issued to node i . We consider two scenarios for issuing the private key (x_i), and the public key (\mathbf{U}_i) of node i . The node's private key x_i , used in the following applications, can be derived by either scenarios

described in this section. In the first scenario, the CA knows the node's secret keys. In this case, N_i 's private key (x_i), and the public value (\mathbf{U}_i) can be generated as follows. First, the CA generates a random scalar h_i and calculates $h_i \times \mathbf{G}$. Next, the CA then generates node i 's public and private keys by performing:

$$\begin{aligned} \mathbf{U}_i &= h_i \times \mathbf{G} \\ x_i &= [H(ID_i, \mathbf{U}_i) \times h_i + d] \text{ mod } \text{ord}\mathbf{G} \end{aligned} \quad (1)$$

The CA issues the values x_i and \mathbf{U}_i to N_i , at which time N_i can establish the validity of the values issued to him by checking whether $x_i \times \mathbf{G} = H(ID_i, \mathbf{U}_i) \times \mathbf{U}_i + \mathbf{R}$. In the second scenario considered, the CA is not allowed to know the node's secret keys and N_i 's private key and public key can be generated as follows. First, the node generates a random value v_i and submits $\mathbf{W}_i = v_i \times \mathbf{G}$ to the CA. Next, the CA generates a random h_i and calculates $h_i \times \mathbf{G}$. The CA then generates the pair of private and public keys by performing:

$$\begin{aligned} \mathbf{U}_i &= \mathbf{W}_i + h_i \times \mathbf{G} \\ p_i &= [H(ID_i, \mathbf{U}_i) \times h_i + d] \text{ mod } \text{ord}\mathbf{G} \end{aligned} \quad (2)$$

and issues the values p_i and \mathbf{U}_i to N_i . At this point, N_i generates his secret key $x_i = [p_i + H(ID_i, \mathbf{U}_i) \times v_i] \text{ mod } \text{ord}\mathbf{G}$ and N_i can establish the validity of the values p_i and \mathbf{U}_i issued to him by checking whether $p_i \times \mathbf{G} = H(ID_i, \mathbf{U}_i) \times (\mathbf{U}_i - \mathbf{W}_i) + \mathbf{R}$. Two important points should be noted here. First, in both cases $x_i \times \mathbf{G} = H(ID_i, \mathbf{U}_i) \times \mathbf{U}_i + \mathbf{R}$, and second since $x_i = [H(ID_i, \mathbf{U}_i) \times (h_i + v_i) + d] \text{ mod } \text{ord}\mathbf{G}$, $x_i \times \mathbf{G} = H(ID_i, \mathbf{U}_i) \times \mathbf{U}_i + \mathbf{R}$, which is identical to the case of the CA being allowed to know the node's secret keys.

III. DoS MITIGATION AND KEY-GENERATION

The ECC-based procedure for key generation, which included certification, as described by the authors in [4] does not include any mechanism for DoS mitigation. The DoS attack considered occurs when a malicious node repeatedly approaches legitimate nodes, requesting to establish a joint secret key. The energy consumed by the legitimate nodes, in the process of key generation, is substantial. Therefore, such an attack strategy can drain their energy. An efficient DoS mechanism should be able to mitigate such attacks. The proposed DoS mitigation approach comprises of two complementing parts. The first pertains to the instigator, Alice, who has to prove her validity to Bob, the party (node) approached. We assume that Alice is a node having limited resources similar to those of Bob. The second part, which takes into effect only if Alice has indeed proven her validity, pertains to Bob, who is required to prove his validity to Alice. We will demonstrate that if the two procedures are successful, i.e., the identity of both Alice and Bob is validated, then an ephemeral key can be issued. The latter implies that each time a certain legitimate node wishes to establish a key with a neighboring node, not only are the chances of a DoS attack diminished, but a different secret key will be generated. We shall refer to the following notations in the context of the proposed DoS

mitigation scheme: n_i is user i 's public key, d_i his private key, CR_i his (CA issued) certificate, and ID_i his public key identification. The following sections describe, in detail, the two stages of the DoS mitigation method.

A. The Instigator Node Proving Its Validity

The specific scenario described in this case pertains to a malicious node who is attempting to drain the energy of a trusted nodes. The first step of a key establishment protocol consists of an instigator node (Alice) initiating communications with another node (Bob). We shall refer to the instigating node as a suspicious node which is required to prove its identity. We thus expect that during the first stage of the key exchange process, the majority of the energy consumed will be on Alice's part. This would mean that if a DoS attack is carried out, whereby a malicious node repeatedly attempts to generate a key with a valid node, the latter will be required to use as little energy as possible. We must assume that most of the nodes are not jeopardized; hence the instigating nodes are to be "presumed innocent until proven guilty". In other words, the amount of energy drained from Alice will be significant, yet not too high so as to deplete her battery too fast. However, if Alice is malicious, and attempts to establish keys with various nodes, she will eventually run out of energy and/or expose her malicious nature. The method described next is based on the notion of key transport [5] using RSA [6] with $e = 3$. We note that $e = 3$ is considered sufficiently secure. (Higher levels of security are satisfied by $e = 2^{16} + 1 = 65537$.) The following four steps constitute an ephemeral key exchange procedure that embeds the DoS mitigation mechanism:

Step 1 - Alice sends Bob her public key, n_A , her identification, ID_A , and her certificate (issued by the CA), CR_A . The certificate is the CA's signature on the association between n_A and ID_A . An example for such an association can be: $n_A \oplus ID_A \equiv H(n_A, ID_A)$. Note that ID_A can be a small number; n_A can be 1024 bits (as in the protocol used here), hence $H(n_A, ID_A)$ depends on the length of n_A . In this case, $CR_A = [H(n_A, ID_A)]^{d_{CA}} \text{ mod } n_{CA}$. Naturally, only the CA can create the CR_A by using its private key d_{CA} .

Step 2 - Bob verifies the validity of the certificate (CR_A) by testing the equality $(CR_A)^e \text{ mod } n_{CA} \stackrel{?}{=} H(n_A, ID_A)$. If the latter holds, Bob knows that n_A and ID_A are undeniably connected. Since $e = 3$, this step requires Bob to compute **only two** modular multiplications. If indeed $(CR_A)^3 \text{ mod } n_{CA} = H(n_A, ID_A)$, Bob can then continue with generating a message m (it will later be shown how this message is utilized as part of the key generation process), compute $t = m^e \text{ mod } n_A$ and transmit t to Alice. Again, since $e = 3$, Bob has to calculate **only 2** modular multiplications at this step. (For $e = 2^{16} + 1$ Bob has to calculate 17 modular multiplications.)

Step 3 - Alice needs to prove that she indeed possesses the private key d_A , proving to her counterpart that her identity is valid. This is true since the CA would have given this private key only to her. Let s_x denote the number of bits in x , the least significant section of m . Alice needs to calculate $t^{d_A} \text{ mod } n_A = m$ and send Bob x . Message m is comprised

out of n bits such that $n \gg s_x$. The rest of the bits in the message will be used for the ephemeral key generation, as will later be described.

It should be noted that, in contrast to Bob (who needs to calculate 2 modular multiplications, or 17 in the worse case), Alice has to perform a computationally heavy task as d_A typically consists of either 512 or 1024 bits. In the latter case she has to calculate 1536 modular multiplications, on the average, using the common square-and-multiply process. To that end, the approach proposed shifts the computational burden on the instigating node.

Step 4 - Bob compares the binary vector x he receives from Alice with the s_x least significant bits in m . If these are identical he determines that Alice's identity is valid. If not, he asserts that Alice is malicious and terminates the key establishment process. It should be noted that this is achieved by performing merely four modular multiplications, two receptions and 1 transmission.

The above process has achieved several key goals. First, the instigating node (Alice) uses more energy than the approached node (Bob) as she calculates $t^{d_A} \bmod n_A$. Yet this is an accepted burden under the assumption that the calculation of $t^{d_A} \bmod n_A$ is performed only once per key generation. As described in [4] there is a need for only two key generations per node. Second, if Alice is malicious and attempts to instigate key generation with more than one node, calculating $t^{d_A} \bmod n_A$ for various types of t 's (different from one correspondent to another) will drain her energy. Third, if the same ID_A is used over and over again then she is bound to be ignored. If Alice is trustworthy, she will need to use her ID_A only twice for both key generations performed [4]. Finally, if Alice tries to impersonate another user by using a different ID_i , then it will immediately be identified since $(CR_A)^e \bmod n_{CA} = H(n_A, ID_i)$ will not hold. In this case, Bob will only have wasted two modular multiplications and one reception.

Two threat models should be considered in this context. First, Alice can attempt to drain Bob's energy by continuously requesting to establish a key, each time using a different ID. Since Bob is only required to calculate $(CR_A)^3 \bmod n_{CA}$ and compare it with $H(n_A, ID_A)$, the computations involved are two Montgomery multiplications alone. Hence the energy consumed in each attempt is relatively small. Moreover, the time Bob spends performing the computations is rather small, thereby not introducing a significant burden in that sense. Second, a malicious node, impersonating Alice, can repeatedly initiate a key establishment process using ID_A . The question is how can Bob know which messages should be ignored? A possible solution would be to maintain a list of IDs of recent nodes that resulted in failed validation (step 2). Bob will then refrain from proceeding with key generation requests originating from these nodes. A time-out mechanism should be employed such that banning of nodes expires after a reasonable duration of time.

B. The Approached Node Proving It's Validity

If the first part of the procedure is successful, i.e., Alice has proven that she is who she claims to be, then Bob will need to do the same. However, if the first stage does not pass, Bob assumes that Alice is not valid, and he will discard the rest of the procedure.

The second stage can be realized in three different ways: (1) using key transport, (2) using the Elliptic Curve Digital Signature Algorithm (ECDSA), and (3) using self-certified fixed key generation [4], [7]. We next describe each of these methods and discuss their respective advantages and disadvantages. Moreover, it will be shown that in each of the cases an ephemeral key is established, which is a primary goal.

1) *Key Transport*: Bob can validate itself to Alice by using the RSA key transport method, similar to that described in section III-A. The random message m , generated by Bob, was encrypted using Alice's public key n_{CA} and e . After sending the encrypted message t , such that $t = m^e \bmod n_A$, Alice can decrypt the message back using her private key, d_A . Eventually, both nodes share the same secret message m . The remaining bits of message m (excluding the s_x least significant bits that were used in stage A) are utilized to establish an ephemeral key. For example, if the length of m is 512 and $s_x = 100$, then there are 412 bits that can be used for authenticating Bob and establishing the ephemeral secret key. In this scenario, y will denote the 200 bits that follow x . The subsequent 212 bits of message m will be labeled z . (The lengths of the components in the message can be negotiated between the two parties.)

The following summarizes the key transport procedure considered:

Step 1 - Bob calculates $S_B = y^{d_B} \bmod n_B$, where y is the next *LSB* portion of message m .

Step 2 - Bob sends Alice his public key, n_B , his identification, ID_B , his certificate (issued by the CA), CR_B , and S_B . As described above, the certificate is the CA's signature on the association between n_B and ID_B . As such, $CR_B = [H(n_B, ID_B)]^{d_{CA}} \bmod n_{CA}$. Only the CA can create CR_B by using its private key d_{CA} .

Step 3 - Alice verifies the following: $(CR_B)^e \bmod n_{CA} \stackrel{?}{=} H(n_B, ID_B)$. If true, Alice knows that n_B and ID_B are undeniably linked. Since $e = 3$, Alice computes only 2 modular multiplications. To check the validity of the certificate, Alice checks the following two equalities

$$(CR_B)^e \bmod n_{CA} \stackrel{?}{=} H(n_B, ID_B) \quad (3)$$

$$(S_B)^e \bmod n_B \stackrel{?}{=} y \quad (4)$$

If true, Alice knows that the corresponding node is indeed Bob, since only he has the same data, y . The ephemeral key resulting will be denoted by $K_{AB-final} = z$, corresponding to the MSB of message m . To complete the authentication cycle key confirmation needs to be preformed.

2) *Elliptic Curve Digital Signature Algorithm (ECDSA)*: Bob can also validate himself to Alice by using ECDSA. The latter is a method for digital signatures, based on ECC. The

elliptic curve employed by the ECDSA can be the same one used in all procedures above. The ECDSA variation proposed, utilizing the components of the message exchanged, m , is:

Step 1 - Bob generates a random number, u , calculates a public value, a point on the curve $\mathbf{V} = u \cdot \mathbf{G}$, where \mathbf{G} is a generating group-point and calculates C , the scalar representation of point \mathbf{V} . Next, he computes $L = u^{-1}(y + d_B \cdot C) \bmod \text{ord}\mathbf{G}$. Finally, he transmits Alice the signature pair (C, L) .

Step 2 - Alice calculates $h = L^{-1} \bmod \text{ord}\mathbf{G}$, $q_1 = y \cdot h \bmod \text{ord}\mathbf{G}$, and $q_2 = C \cdot h \bmod \text{ord}\mathbf{G}$. She next obtains the curve point: $\mathbf{P} = q_1 \cdot \mathbf{G} + q_2 \cdot \mathbf{V}$, where n_B is Bob's public key, and calculates C' , the scalar representation of point \mathbf{P} . The algorithm concludes when Alice validates that $C = C'$. If the latter holds, Bob is validated.

Step 3 - The ephemeral key resulting will be denoted by $K_{AB-final} = z$, corresponding to the MSB of message m . To complete the authentication cycle key confirmation needs to be preformed.

3) *Self-Certified DH Fixed Key-Generation*: One of the methods in which Bob can prove his validity to Alice is by using a self certified method similar to the ephemeral one described in section III. The ephemeral method can certainly be used, but when the primary focus is to minimize energy drainage, a self certified fixed key generation is advisable since it consists of less computations. We now go back to the notations used in section II where self certified ephemeral key generations were described.

A self-certified DH fixed key-generation is achieved by the following two steps: (1) N_i and N_j exchange the pairs (ID_i, \mathbf{U}_i) and (ID_j, \mathbf{U}_j) , respectively, and (2) N_i and N_j generate the session-key,

$$\begin{aligned} K_{ij} \text{ (generated by } N_i) &= x_i \times [H(ID_j, \mathbf{U}_j) \times \mathbf{U}_j + \mathbf{R}] \\ K_{ji} \text{ (generated by } N_j) &= x_j \times [H(ID_i, \mathbf{U}_i) \times \mathbf{U}_i + \mathbf{R}]. \end{aligned} \quad (5)$$

The two keys are expected to be identical, having the value $x_i \times x_j \times \mathbf{G}$. (i.e., N_i calculates: $x_i \times [H(ID_j, \mathbf{U}_j) \times \mathbf{U}_j + \mathbf{R}] = x_i \times [H(ID_j, \mathbf{U}_j) \times h_i \times \mathbf{G} + d \times \mathbf{G}] = x_i \times [H(ID_j, \mathbf{U}_j) \times h_i + d] \times \mathbf{G} = x_i \times x_j \times \mathbf{G}$. Similar logic is applied by the calculations performed at N_j . However, these identities hold only for valid ID's. Therefore, to complete the authentication cycle there is a need for key-confirmation, during which the two nodes either verify that they share an identical key by encrypting and decrypting a test value, or by establishing a communication session and implicitly verify that they share the same key. Verifying that the keys generated by the two nodes are equal then establishes their correct identities.

A primary contribution offered by this method of self-certified fixed key generation lies in the number of exponentiations needed to calculate the value $x_i \times x_j \times \mathbf{G}$. As indicated above, each node (among each pair of nodes) calculates the value $x_i \times x_j \times \mathbf{G}$. Note that the calculations performed by N_i are $K_{ij} = x_i \times [H(ID_j, \mathbf{U}_j) \times \mathbf{U}_j + \mathbf{R}] = x_i \times H(ID_j, \mathbf{U}_j) \times \mathbf{U}_j + x_i \mathbf{R}$. Further note that the calculations have been separated into two parts. The first is a dynamic scalar by point

multiplication executed in an ad hoc manner (as it contains the value \mathbf{U}_j). The second is a scalar by point multiplication that can be calculated and stored "before" the key-generation session commences, thereby avoiding the need for a real-time exponentiation (as it contains information known *a priori* by node i). It is clear that N_i is able to calculate its session-key by a single online exponentiation ($x_i \times H(ID_j, \mathbf{U}_j) \times \mathbf{U}_j$) instead of two. Similar considerations apply to N_j .

We shall refer to the joint fixed key shared by Alice and Bob as $K_{AB-temp}$. In addition, as an integrated part of the key generation process, if the two generated keys are indeed identical, authentication is achieved. Therefore, the approached node has proven its validity to the instigator.

The goal of the entire procedure is to establish a shared joint secret key. It is highly desirable for that key to be ephemeral, i.e., two nodes generate a different key for each session established. Ephemeral key-generation is more secure and is generally preferred when time and resources permit. A self-certified DH ephemeral key-generation is also possible, but would consume three times more energy when compared to the fixed key case. In order to establish an ephemeral key, the two nodes can utilize bits in message m , (generated by Bob) excluding the first x least significant bits. Hence, the final shared ephemeral key can be defined as

$$K_{AB-final} = H(K_{AB-temp}, m'), \quad (6)$$

where H is a hash function and m' is the random message m , excluding the x least significant bits.

IV. CONCLUSIONS

This paper introduced a public key cryptographic method for preventing DoS attacks that target the draining of battery energy in WSN ephemeral key establishment. By exploiting the asymmetry in RSA signature generation, a robust approach to minimizing energy usage at the node being attacked has been proposed. Combining the DoS mitigation with self-certified ECC-based key generation yielded a highly resource-efficient security framework. Moreover, the concept developed can be applied to a wide range of additional security services that are currently not offered in WSN environments.

REFERENCES

- [1] D. W. R. Molva, G. Tsudik, *Security and Privacy in Ad-hoc and Sensor Networks*, vol. 3813 of *Lecture Notes in Computer Science*. 2005.
- [2] A. Perrig, J. Stankovic, and D. Wagner, "Security in wireless sensor networks," *Communications of the ACM*, vol. 47, pp. 53-57, June 2004.
- [3] O. Arazi, I. Elhanany, D. Rose, and H. Q. B. Arazi, "Self-certified public key generation on the intel mote 2 sensor network platform," in *Third Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks, SECON 06*, 2006.
- [4] O. Arazi and H. Qi, "Load-balanced key establishment methodologies in wireless sensor networks," *International Journal of Sensor Networks, IJSN*, vol. 1, April 2006.
- [5] A. M. Eskicioglu and E. J. Delp, "A key transport protocol based on secret sharing applications to information security," *IEEE Transactions on Consumer Electronics*, vol. 48, pp. 816-824, November 2002.
- [6] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120-126, 1978.
- [7] B. Arazi, "Certification of dl/ec keys," in *Proc. of the IEEE P1363 Study Group for Future Public-Key Cryptography Standards*, May 1999.

Toward Mitigating Denial of Service Attacks in Power-Constrained Sensor Networks



Ortal Arazi, Hairong Qi

College of Engineering
Dept. of Electrical & Computer Engineering
The University of Tennessee

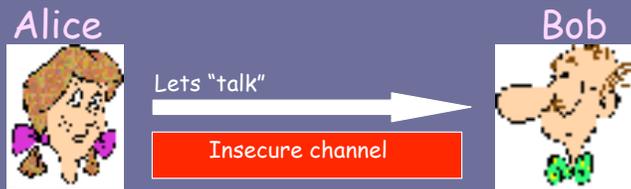
Cyberspace Sciences & Information Intelligence Research Group (CSIIR)
Oak Ridge National Laboratory, Oak Ridge

Background & Motivation

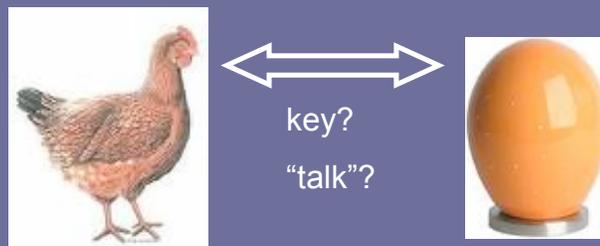
- ◆ Wireless sensor network (WSN) applications are growing
 - Military as well as civilian
 - Many research challenges
- ◆ Typical deployment environment is prone to malicious attacks
- ◆ What makes security for WSN so unique ...
 - Scarce resources
 - Energy, memory, computation, communications
 - **Vulnerability to Denial of Service attacks**



Instigating Communication



Alice and Bob: legitimate users
They would like to start communicating



Denial of Service Mitigation

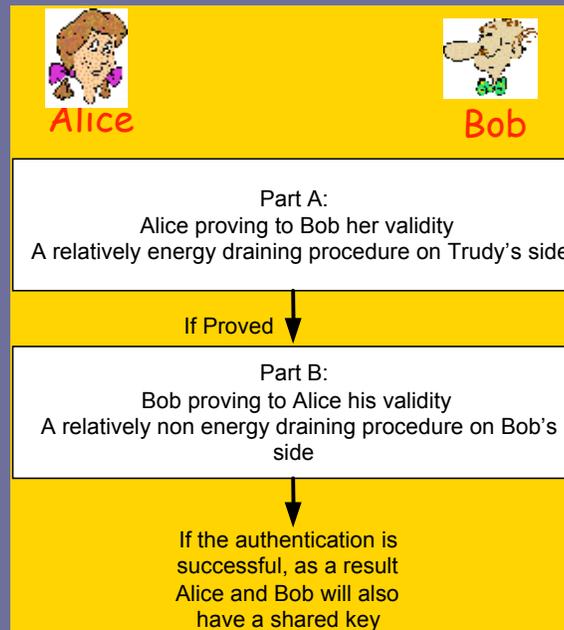


Trudy wants to:

- 1) Drain Bob's energy
- 2) Impede him from talking to other legitimate users

How can we prevent that?

Denial of Service Mitigation Procedure



The Denial of Service Mitigation Part A

(n_A, e) → Alice's public key
 (n_A, d_A) → Alice's private key
 CR_A → Alice's certificate
 ID_A → Alice's identification

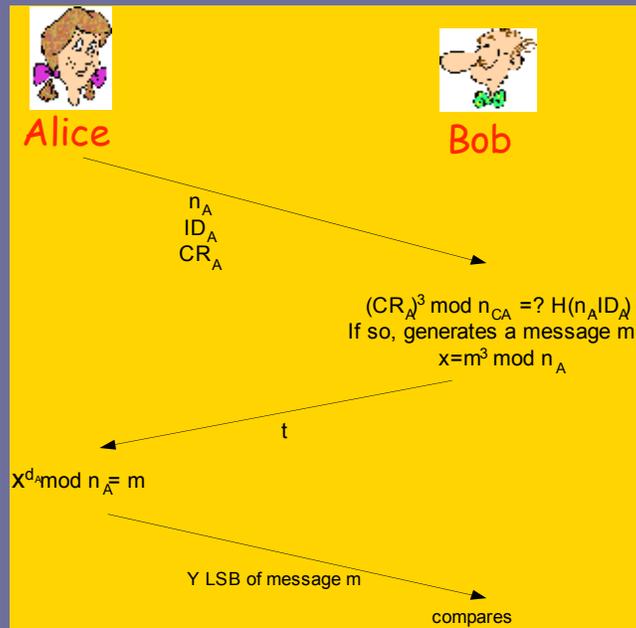
CR- A certificate

The CA's signature on the association between n_A and ID_A

$$CR_A = [H(n_A, ID_A)]^{d_{CA}} \bmod n_{CA}$$

$$H(n_A, ID_A) \equiv n_A \oplus ID_A$$

The Denial of Service Mitigation Part A

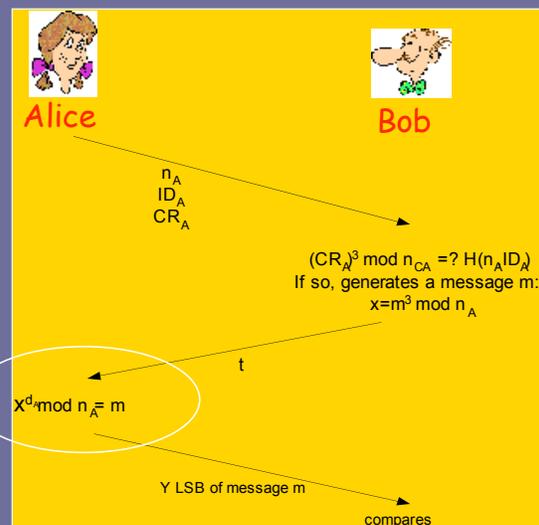


Timing and energy considerations Part A

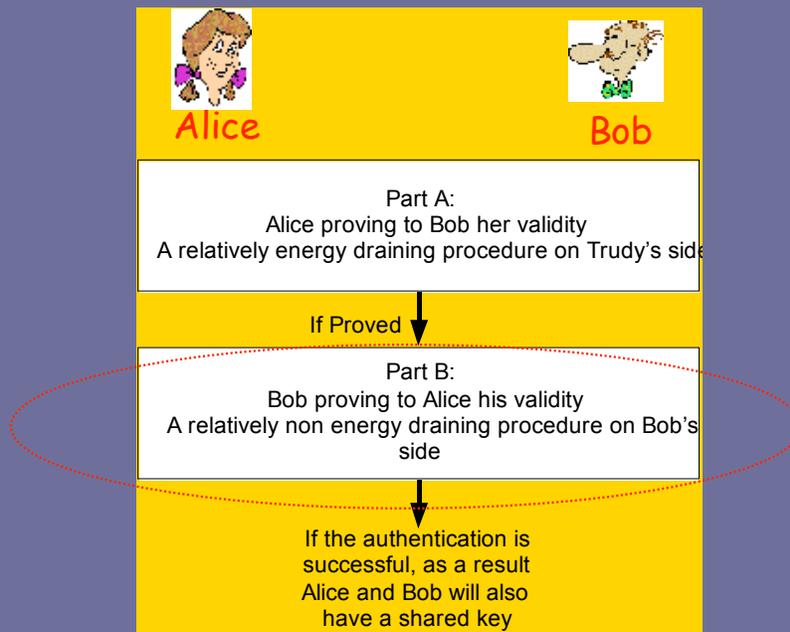
$$t^{d_A} \bmod n_A = m$$

~ 160 mJ
~250 msec
(for a 512 bit key)

All other energy and time consumptions (from the other procedures) are negligible



The Denial of Service Mitigation Procedure



The Denial of Service Mitigation Part B

How can Bob prove his validity?



Several solutions:

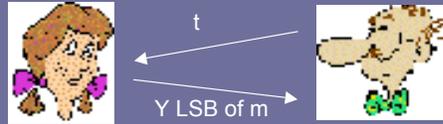
1. Using the self-certified fixed key method
2. Using RSA
3. Using ECDSA

The Denial of Service Mitigation

Part B

(1) Using the self-certified fixed key method

When calculating the self-certified fixed key, Bob is authenticated!



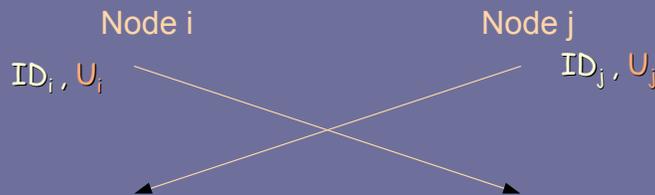
$$X_A[H(ID_B, U_B) * U_B + R] , X_B[H(ID_A, U_A) * U_A + R]$$

Both have the message m Bob sent in part A

The final ephemeral key:
 $f(\text{fixed key}, m/_{\text{without the Y LSB sent on the open channel}})$

Self certified DH key generation: Fixed key

Each node is given by the CA (Certifying authority) a set of public and private keys: (U_v, X_v)



Node i calculates: $X_i[H(ID_j, U_j) * U_j + R]$ = $X_j[H(ID_i, U_i) * U_i + R]$:Node j calculates

- ID_v : identification of node v - scalar
- U_v : node v's public key, generated by the CA - a point on the curve
- X_v : node v's private key, generated by the CA - scalar

Self certified DH key generation: Fixed key

mathematical assertions ...

As given by the CA:

$$U_i = h_i * G$$

$$X_i = [H(ID_i, U_i) * h_i + d] \text{ mod } \text{org } G$$

$$U_j = h_j * G$$

$$X_j = [H(ID_j, U_j) * h_j + d] \text{ mod } \text{org } G$$

Node i calculates:

$$\begin{aligned} & X_i [H(ID_j, U_j) * U_j + R] \\ &= X_i [H(ID_j, U_j) * h_j * G + d * G] \\ &= X_i [H(ID_j, U_j) * h_j + d] * G \end{aligned}$$

$$= X_i * X_j * G$$

Node j calculates:

$$\begin{aligned} & X_j [H(ID_i, U_i) * U_i + R] \\ &= X_j [H(ID_i, U_i) * h_i * G + d * G] \\ &= X_j [H(ID_i, U_i) * h_i + d] * G \end{aligned}$$

$$= X_j * X_i * G$$

R : the CA's public key = $d * G$

d : the CA's private key

G : a generating group-point, used by all relevant nodes

h_v : a random 160 bit number generated by the CA

- a point on the curve

- scalar

- a point on the curve

- scalar

The Denial of Service Mitigation

Part B

(2) Using RSA



Message m, 512 bits		
Z	Y	x
212 bits	200 bits	100 bits

1. Bob calculates:

2. Bob send Alice:

3. Alice calculates:

CR_B
 ID_B
 n_B
 S_B

If so, the final ephemeral key: **Z**

The Denial of Service Mitigation

Part B

(3) Using ECDSA

Bob:

- Generates a random number: u . Calculate $C = u \cdot V$.
C- the scalar representation of point V.
- Calculates $L = (m + H(C)) \cdot u^{-1} \pmod{n}$.
The signature is the pair (C,L)
- Sends Alice (C,L)



Alice:

- Computes: $C' = (m + H(C',L)) \cdot u^{-1} \pmod{n}$
- Obtains the curve point: $P = C' \cdot G$.
C'- the scalar representation of point P

Message m, 512 bits		
Z	Y	X
212 bits	200 bits	100 bits

If so, the final ephemeral key: Z

If $C=C'$, then the signature is valid, it is Bob!

The Denial of Service Mitigation, Part B

Comparing the three methods

The time is measured in: ECC point by scalar multiplications
Approximately: 40 msec

	Method 1 (Fixed key)	Method 2 (RSA)	Method 3 (ECDSA)
Part A Alice proving her identity	Alice: 6 Bob: ~0	Alice: 6 Bob: ~0	Alice: 6 Bob: ~0
Part B Bob proving his identity	Alice: 2 Bob: 2	Alice: ~0 Bob: 6	Alice: 2 Bob: 1
Overall computational overhead	~10	~12	~9

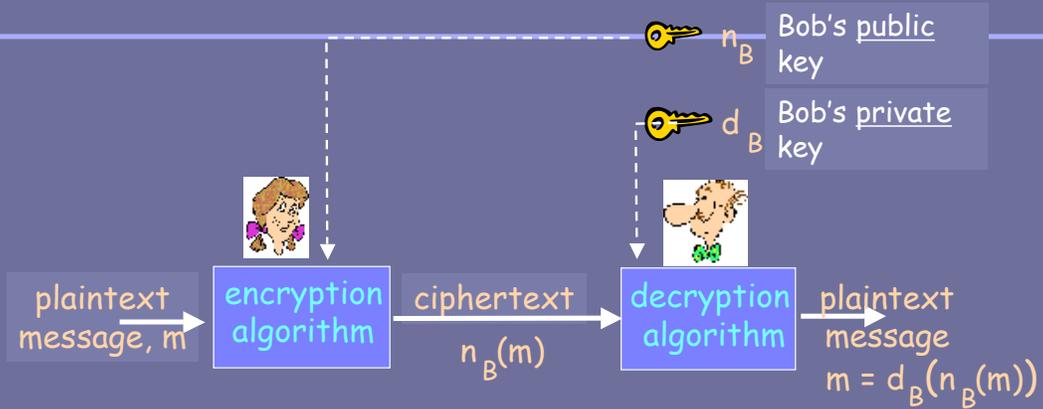
Conclusions

- ◆ PKC implementation in WSN is feasible
- ◆ ECC shows promise as crypto technology
- ◆ DoS is a primary threat
- ◆ Introduced a hybrid RSA/ECC framework for mitigating DoS attacks
- ◆ Using the fixed key approach or the ECDSA approach proved to be highly beneficial

Thank You
Questions ?



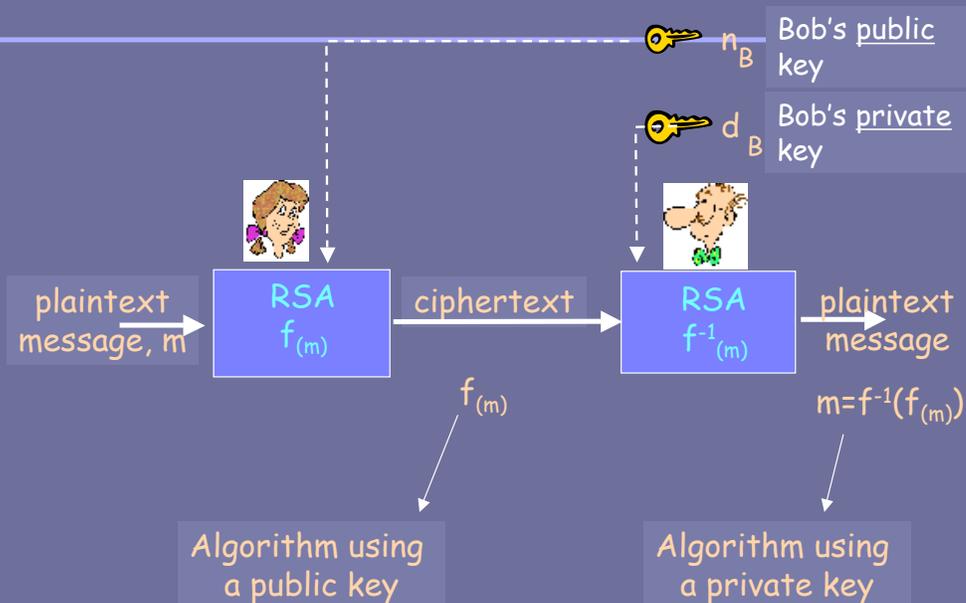
Public Key Cryptography



Requirements:

- 1 $d_B(n_B(m)) = m$
- 2 Given a public key it should be impossible to compute the private key

Public Key Cryptography - RSA



RSA (Rivest-Shamir-Adelman): Choosing Keys

1. Choose two large prime numbers p, q .
(e.g., 1024 bits each)
2. Compute $n = pq$, $z = (p-1)(q-1)$
3. Choose e (with $e < n$) that has no common factors with z . (e, z are "relatively prime").
4. Choose d such that $ed-1$ is exactly divisible by z .
(in other words: $ed \bmod z = 1$).
5. Public key is (n, e) . Private key is (n, d) .

RSA: Encryption, decryption

Given (n, e) and (n, d) as computed above:

1. To encrypt bit pattern, m ($m < n$), compute
 $c = m^e \bmod n$ (i.e., remainder when m^e is divided by n)
2. To decrypt received bit pattern, c , compute
 $m = c^d \bmod n$ (i.e., remainder when c^d is divided by n)

Magic happens

$$m = \underbrace{(m^e \bmod n)}_c^d \bmod n$$

RSA: Why is that $m = (m^e \bmod n)^d \bmod n$?

Useful number theory result: If p, q prime and $n = pq$, then:
 $x^y \bmod n = x^{y \bmod (p-1)(q-1)} \bmod n$
(Fermat's Small Equation)

$$\begin{aligned} \underbrace{(m^e \bmod n)^d}_{C - \text{the encrypted message}} \bmod n &= m^{ed} \bmod n \\ &= m^{ed \bmod (p-1)(q-1)} \bmod n \\ &\quad \text{(using number theory result above)} \\ &= m^1 \bmod n \\ &\quad \text{(since we chose } ed \text{ to be divisible by } (p-1)(q-1) \text{ with remainder 1)} \\ &= m \text{ (since } m < n \text{)} \end{aligned}$$

Calculating the keys

$(n_A, e) \rightarrow$ Alice's public key
 $(n_A, d_A) \rightarrow$ Alice's private key

$$n_A = p_1 \cdot p_2$$

p_i - Pseudo random prime number

$$d_A = e^{-1} \bmod \varphi(n_A)$$

- $\varphi(n_A)$ - Euler's Totient Function

Euler's Totient Function $\varphi(n_A)$ returns the number of Integers less than n_A

$$\varphi(n_A) = (p_1 - 1) \cdot (p_2 - 1)$$

Calculating the keys (cont)

- Public key is (n_A, e)
- Private key is (n_A, d)

$e=3$

$$\implies d_A = e^{-1} \bmod \varphi(n_A) = 3^{-1} \bmod (p_1 - 1)(p_2 - 1)$$

Calculating d_A :

lets choose p_1, p_2 such that $p_1, p_2 = 2 \bmod 3$

$$\implies (p_1 - 1) \bmod 3 = (p_2 - 1) \bmod 3 = 1$$

$$\implies (p_1 - 1)(p_2 - 1) \bmod 3 = 1$$

$$\implies \varphi(n_A) \bmod 3 = 1$$

$$\implies 2\varphi(n_A) + 1 = 3x, \quad \forall x \iff x = 3^{-1} \bmod \varphi(n_A) \equiv d_A$$

\implies

$$d_A = \frac{2\varphi(n_A) + 1}{3}$$

Checking the Certificate

n_{CA} and d_{CA} are calculated using the exact procedure indicated above

$$n_{CA} = p_1 \cdot p_2$$

$$d_{CA} = e^{-1} \bmod \varphi(n_{CA}) = \frac{2\varphi(n_{CA}) + 1}{3}$$

$$CR_A = [H(n_A, ID_A)]^{d_{CA}} \bmod n_{CA}$$

The validity procedure:

$$(CR_A)^e \bmod n_{CA} \stackrel{?}{=} H(n_A, ID_A) \iff (CR_A)^3 \bmod n_{CA} \stackrel{?}{=} H(n_A, ID_A)$$

\implies the calculations is the following:

$$(CR_A)^3 = \left[[H(n_A, ID_A)]^{d_{CA}} \right]^3 \bmod n_{CA} = [H(n_A, ID_A)]^{d_{CA} \cdot 3} \bmod n_{CA}$$

Since $d_{CA} = 3^{-1} \bmod \varphi(n_{CA})$, $d_{CA} \cdot 3 = 1$

$$\implies (CR_A)^3 = [H(n_A, ID_A)]^1 \bmod n_{CA} = H(n_A, ID_A)$$

$$[H(n_A, ID_A) < n_{CA} = p_1 \cdot p_2]$$

Tolerating Change in a Secure Environment: A Visual Perspective

Shawn A. Bohner, Denis Gracanin, and Riham Hassan
Virginia Tech, Dept. of Computer Science, Falls Church VA 22043
sbohner@vt.edu, gracanin, rhabel@vt.edu,

Abstract

With the relentless growth in trusted systems software and governmental mandates for evaluation, automated support for visualizing and navigating software artifacts is no longer a luxury. Much of this growth is in post-deployment and re-evaluations take considerable effort to complete. For those involved in evaluating software security, the job of examining their Target of Evaluation (TOE) for potential security vulnerabilities is daunting and often protracted. Making sense of the relationships between components, the supporting documentation, and the testing can stretch the limits of human capacities. Even seemingly innocuous software changes to the system can result in considerable effort establishing the extent of vulnerabilities that could be introduced. This paper describes research for using visualization technology for supporting trusted system evaluation. Recognizing that the more formal the software artifact representation form, we exploit more opportunities to use automation in the evaluation.

1. Introduction

In an era of heightened security concern, trusted software systems are increasingly growing, evolving, and sustaining changes. As changes are introduced, the process of assuring that security vulnerabilities are not introduced becomes increasingly labor intensive and error prone [1]. As society increasingly depends on software, the size and complexity of software systems continues to grow making them more difficult to understand and evolve. This trend applies equally to trusted systems. Manifold dependencies between critical elements of trusted software now drive the architectures and increasingly sway the overall system architecture [2]. When changes are introduced, it is often difficult to determine the resulting ramifications. Will the change introduce security vulnerabilities? Will the information assurance be compromised? To what degree can we be confident that the system will operate unhindered by outside attacks? These can be answered only if there are mechanisms to evaluate these aspects of the system.

Considerable effort has been expended developing a software security evaluation process and associated criteria [3, 4]. With the Common Criteria Security Evaluation (CCSE) [5-7] [ISO/IEC Standard 15408]

requirements mandated as of July 2002, the backlog of software to comply is immense and requires both process and automated support. This regulation created a sudden demand for understanding software and security impacts [1]. Evaluating software for security issues entails understanding common criteria related security requirements, their design dependencies in the systems under evaluation, and the degree to which their design, implementation, and testing processes and artifacts convey confidence that the security elements have been realized appropriately.

The CCSE process is time consuming and labor-intensive. It involves evaluators wading through large bodies of system and software documentation to determine if there is enough confidence to employ a software product in a secure environment. For a typical software system, several weeks of effort are expended to produce an evaluation that leads to a validation report. The evaluation is largely based on an evaluator's opinion of how well the system meets the security criteria.

While formal specifications of security requirements coupled with effective traceability techniques can provide leverage in the subsequent evaluation of trusted systems [8], the use of visualization technology can offer both and extension and confirmation of these approaches. Moreover, the use of visualization technology can help with the overwhelming amount of information and relationships between information that goes with the various software system artifacts.

The development and maintenance life cycles entail traceability relationship dependencies that extend from early requirements to architecture, design, implementation, and all stages of testing. Source code has data and control dependencies that form program dependency graphs employed in software analysis. Managing the evolution of these systems still entails configuration and version dependencies. And now with the growth of packaged applications and component-based development, interoperability between components must also be incorporated into the dependency network used to understand software. All of these contribute to a security evaluator's understanding of the software and its respective security aspects.

While understanding can be difficult from a development perspective, it is even more challenging from the maintenance or evolution perspective.

Constraints of an existing system with all of its software artifacts (or lack there of) can add significant complexity to the software change situation [9].

While software requirements, architecture, and design methods provide effective means for dealing with some of this challenge, the software community has found it necessary to employ software impact analysis techniques such as traceability and source code analysis to understand and account for relationships between software objects [10-12].

In this research we examine criteria for software security, effective means of organizing the software system information for evaluation, and visualization techniques for developing insights that lead to more effective security evaluations. Since software changes and evaluations are subject to subsequent updates, we pursue how visualization can help support tolerating changes while preserving security.

1.1. Understanding Dependencies

Software systems must be understandable in order to create and change them. However, short of developing systems with formal methods and specifications, most software development methods have significant gaps in dependency information that, when absent for software changes or security evaluations, leads to fragile software products or information assurance vulnerabilities. A situation that occurs time and time again is where a design decision is made without visibility into the potential impacts. For example, an exception handling approach for buffers is considered from the perspective of programming efficiency and left to the system to handle when overflow occurs. From a software change perspective, this might be entangled with many system services, some of which the ripple effect may not be deterministic. From a security perspective, this may (and has) turned out to be a significant security vulnerability where Internet intruders obtain access to system level services and wreak havoc on the system. Had the software engineers had visibility into the issue, a different tactic might have been employed or at least some safety mechanism may have been implemented.

While this example is one that most software engineers face, the real issue rests in the limited visibility that today's technology provides for relevant software dependencies. Analogous to the situation when source code analysis tools were introduced, we now face considerable complexities that dependency analysis can alleviate. With the increased size and complexity, new artifacts have been introduced into the software product bringing more complexities still. The program dependency graphs (PDG) that were then and are now used to represent control and data flow dependencies must be extended to resolve this situation. The semantics of the objects and the relationships between them must be extended beyond programming idioms and include other specifications like requirements and design.

Considerable traction can be achieved in developing an essential dependency model that encompasses requirements, architectural, and detailed design relationships and connects them with implementation dependencies. This would enable software engineers and software security evaluators to reason effectively about software change and security. Since demonstrating this for all software domains would dilute the effort, we focused on the software security area, building upon successful research work accomplished for the Commonwealth Information Security Center [13].

1.2. Common Criteria Security Evaluation

In the CCSE, the product to be evaluated is called the Target of Evaluation (TOE) and the organization that requests the evaluation is called the Sponsor. The certification laboratory is the Evaluator. The TOE can be evaluated to various levels of assurance called the Evaluation Assurance Levels (EAL). The outcome of evaluation is an Evaluation Technical Report (ETR), which is used to generate and publish the Validation Report (VR) by a Validator. The TOE is evaluated according to security requirements conveyed in the Security Target (ST). An application independent set of high-level security requirements for families of products is called a Protection Profile (PP).

CCSEs essentially checks for completeness and correctness of a system's security features. To check for correctness, first the evaluator needs to navigate through the labyrinth of software artifacts. The navigation through all software artifacts (e.g., requirements and design documents, code, tests, and related documents) can be arduous and time consuming. Further, the manual process does not provide the vendor, who is preparing the TOE, any mechanism to show the "gaps" or missing artifacts in the TOE document. Hence the vendor must wait for evaluator to go through the artifacts and inquires for missing or additional information.

2. Formalism in Security Assurance

The Evaluation Assurance Level (EAL) determines the level of formalism or rigor required for a given application (EAL1 is most basic and cheapest, while EAL7 is the most rigorous and expensive).

EAL1– Functionally Tested: Basic assurance of security by analyzing functional specifications and guidance.

EAL2– Structurally Tested: Moderate level of assurance by EAL1 plus high-level design and independent testing of the security functions for vulnerability assessment.

EAL3– Methodically Tested and Checked: Provides moderate level of assurance by including EAL2 plus evidence of sound development practices.

EAL4– Methodically Designed, Tested and Reviewed: Moderate/high level of assurance - highest level economically feasible to retrofit an existing product line.

EAL5– Semiformally Designed and Tested: Provides security engineering based upon rigorous commercial development practices to ensure resistance to attackers.

EAL6– Semiformally Verified Design and Tested: High assurance through security engineering techniques in a rigorous development environment to reduce risks.

EAL7– Formally Verified Design and Tested: Highest assurance level - requires formal design verification.

These indicate a trade-off between the rigor to ensure low security risks and the cost to accomplish it. That is, the investment to ensure security should align with the benefit gained from the rigor. Note that levels 5-7 specify some range of formal representation – the more formal the representation, the higher the odds of identifying security vulnerabilities. A corollary to this is that with more formal representations, the opportunity increases to use automated verification and evaluation technologies such as theorem provers, analysis and modeling, and visualization tools – key research driver.

3. Analytics and Visualization

Analytic solutions offer a means of examining indicators that lead to discovery. The level of confidence goes up when we produce a mathematical equation or proof that supports our assertion or negation. Formulas, however, are an intermediate form of what we believe is true – we often “see” the answer before hand. This is the concept behind “visual thinking” [14]. Kriz outlines this idea in a number of accounts ranging from Albert Einstein to J. Willard Gibbs. Gibbs, a pioneer of thermal dynamics, in his ground breaking work [15], first analytically formulates the equations that form the basis for the

mathematics used to describe the first and second laws of thermal dynamics today. Concluding his work, Gibbs dispenses with the analytics in favor of the visual form as he only used the analytics as an intermediation to the concepts he was trying to communicate. Note that at the time of writing, 1873, visual depictions did not exist as they do today. It makes one wonder what else Gibbs would have discovered if he had today’s tools.

4. Security Impact Analysis Virtual Environment (SIAVE)

Evaluating trusted software systems often entails large volumes of documentation containing related concepts that are not organized as such. There are frequently gaps, disconnects, and ambiguities. In our research we found that most of the evaluator’s effort was expended on organizing and wading through all of the material to gain an acceptable level of understanding. To expedite CCSEs, we prototyped the SIAVE to automate many of the CCSE process’s laborious tasks while retaining the creative part for humans. SIAVE partially automates tasks associated with both the vendor and evaluator. We had three key goals: 1) Improve efficiency of preparation process; 2) Improve evaluation cycle time through better preparation and evaluation process; and 3) Improve evaluation effectiveness through better visibility.

Figure 1 depicts the workflow for the SIAVE. Once the ST/PP report is generated, the vendor opens the SIAVE template document and runs a GenerateTemplate macro which automatically generates the set of CC and requirement elements required for successful evaluation.

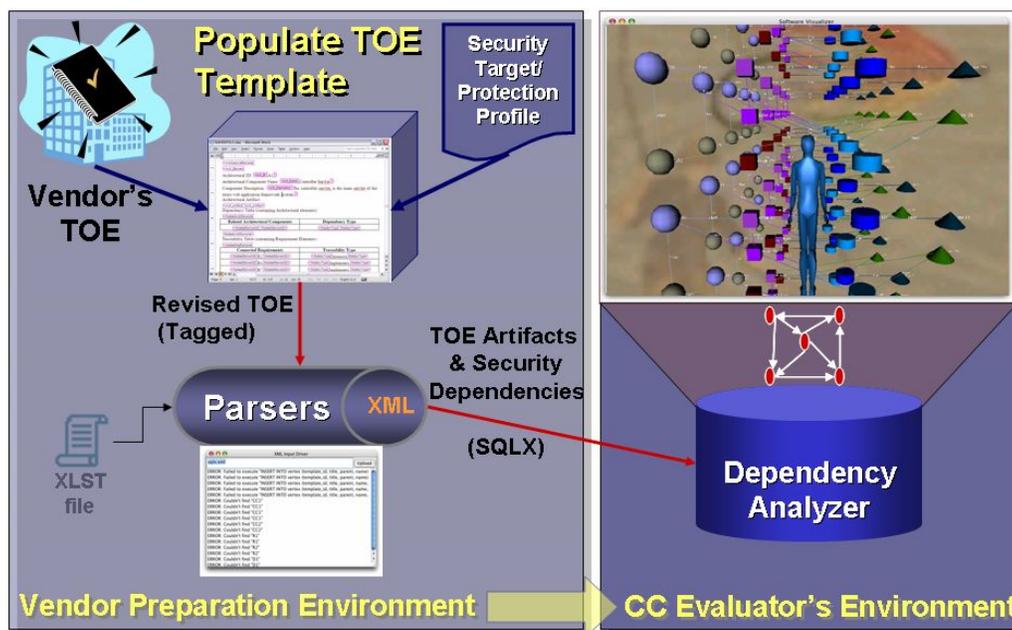


Figure 1: Security Impact Analysis Visualization

The template is designed to help the vendor in revising the TOE to confirm to CC evaluation standards. The SIAVE template lists the CC elements, connected requirements, architecture, design, code and other software artifacts that need to be filled in for a successful CCSE. The vendor then pours into the template the appropriate TOE information which includes image files also. The “fill in process” enables the vendor to understand gaps in the system that needs to be addressed prior to submitting the system for evaluation. Once the template is filled in, the exportdata macro is executed by the vendor to convert the document into a series of SQL statements and stores in a SQLX document. The images in the document are saved as separate files into the directory where the document exists. The SQLX document is then executed using a custom built application called the DB Input Driver. This application does error and document structure checking before storing the contents of the document in the database.

The evaluator by means of the 3D environment then views the contents stored in the database. The 3D environment (two other versions were developed in phase one) is the enabler that helps the evaluator to check for completeness of security aspects (the first view shows the important components and how they are linked) and to check for correctness (helps to navigate through the artifacts faster). The environment allows the evaluator to view the whole system, and then explore parts of the system more deeply. The immersion mechanism allows the evaluator to select an object of interest, read about its details and even get immersed in that object – become the object of interest and view the system from that object’s perspective.

5. Conclusions

This research takes the two proven areas of impact analysis and virtual environments, and applies them to a relevant and growing area of trusted system evaluation. We produced a model of dependency relationships, a basic prototype environment using key impact analysis identification techniques (transitive closure, slicing, and semantic inference) and incorporating the initial 3D visualization interface with improved navigational instruments for security evaluators.

The SIAVE uses two key approaches for the preparation and evaluation process: templates to facilitate ingestion of the TOE and, immersion technology to assist in navigation and visual analytics.

To improve efficiency of the preparation process, we reduced time in creating and revising TOE by providing standard templates into which vendor can “pour in” information about the system. We provide automated customization of the template for a specific TOE and EAL. We reduced effort and time spent in vendor-evaluator interaction cycles – the template served as a checklist, giving the vendor an initial indicator if the system might pass the certification.

To improve evaluation cycle time through better preparation and evaluation process, we reduced the conceptual distance between the various artifact representations and the standardized format used in the SIAVE system by automated generation and partial customization of the template. We reduced effort and time by identifying failing evaluations early via the aforementioned checklist. The map created for navigation has a great feature of providing an initial analysis (if the map is not complete enough for navigation, it is probably an indication of likely failure and a pruning opportunity for an overburdened evaluation process).

6. References

1. Prieto-Diaz, R., *The Common Criteria Evaluation Process: Process Explanation, Shortcomings and Research Opportunities*, CISC Technical Report CISC-TR-2002-003. December 2002, James Madison Univ.: Harrisonburg, VA.
2. Medvidovic, N. and R. Taylor, *A Classification and Comparison Framework for Software Architecture Description Languages*. IEEE Transactions on Software Engineering, 2000. **26**(1): p. 70-93.
3. Anderson, R.J., *Security Engineering: A Guide to Building Dependable Distributed Systems*. 2001 John Wiley & Sons, Inc., New York, NY.
4. Vatterling, M., G. Wimmel, and A. Wisspeintner, *Secure Systems Development Based On The Common Criteria: The PalME Project*. ACM SIGSOFT Software Engineering Notes, 2002. **27**(6): p.129-138.
5. *Information technology -- Security techniques -- Evaluation criteria for IT security -- Part 1: Introduction and general model*. 2005, (ISO) JTC 1/SC 27.
6. *Info. tech. - Security techniques - Evaluation criteria for IT security Part 2: Security functional requirements*.
7. *Info. tech. -- Security techniques -- Evaluation criteria for IT security -- Part 3: Security assurance requirements*.
8. Lamsweerde, A.v. *Elaborating Security Requirements by Construction of Intentional Anti-Models*. in *26th International Conference on Software Engineering*, 2004.
9. Reiss, S. *Constraining Software Evolution*. in *International Conference on Software Maintenance*. 2002.
10. Arnold, R.S. and S.A. Bohner. *Impact Analysis - Towards a Framework for Comparison*. in *International Conference on Software Maintenance*. 1993: IEEE Computer Society.
11. Bohner, S. and R. Arnold, *Software Change Impact Analysis*. 1996: IEEE Computer Society Press.
12. Lee, M., A.J. Offutt, and R.T. Alexander. *Algorithmic analysis of the impacts of changes to object-oriented software*. in *34th International Conference on Technology of Object-Oriented Languages and Systems*. 2000.
13. Bohner, S., et al., *Software Security Impact Analysis Visualization Research: Phase 2 Report*. 2004, Commonwealth Information Security Center Technical Report, James Madison University: Harrisonburg, VA.
14. Kriz, R. *Reports on the Visual Thinking Experience 2007* [cited March 2007]; Available from: www.sv.vt.edu/classes/ESM4714/Gen Prin/vizthink.html.
15. Gibbs, J.W., *A Method of Geometrical Representation of the Thermodynamic Properties of Substances by Means of Surfaces*. Trans. of the CT Academy, 1873. **2**: p. 382-404.

Tolerating Change in a Secure Environment: A Visual Perspective



Shawn Bohner
Virginia Tech

May 15, 2007



Problem

Common Criteria Evaluation Dilemma

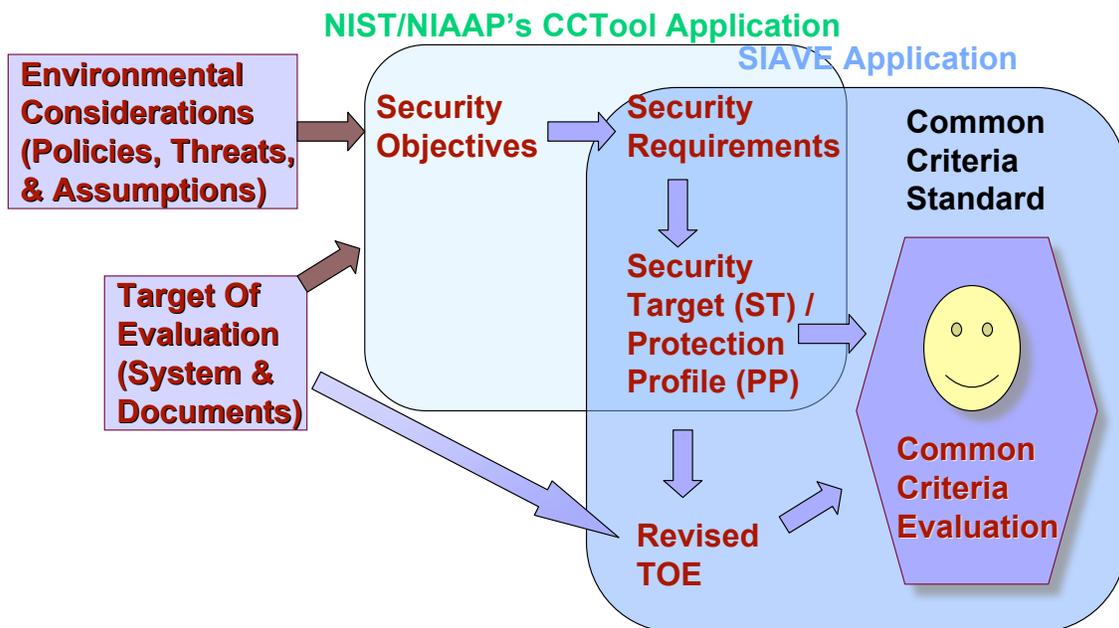
- **Common Criteria Security Evaluations (CCSE)**
Demand exceeding supply of Evaluators
 - Labor intensive CCSE process
 - Effort in Weeks and Calendar time in Months
 - National Information Assurance Acquisition Policy (NSTISSP #11) July 2002 mandate for security related software evaluation
 - Limited number of Testing Labs
 - And then there are all the software updates...
- **How can this situation be alleviated?**
 - Relax policy & allow lesser/non-evaluated systems
 - Increase supply of Evaluators
 - Increase the productivity of Evaluators 

Quicken and Clarify CCSE

- Improve Efficiency of CCSE Process through Better Navigation
 - Reduce time in navigating the documentation (shorten the conceptual distances)
 - Reduce effort and time by identifying failing evaluations early
 - Reduce time for key time consuming activities
- Improve Effectiveness of CCSE Process through Better Visibility
 - Increase confidence of evaluations
 - Better decisions



CCSE via Security Impact Analysis Virtual Environment



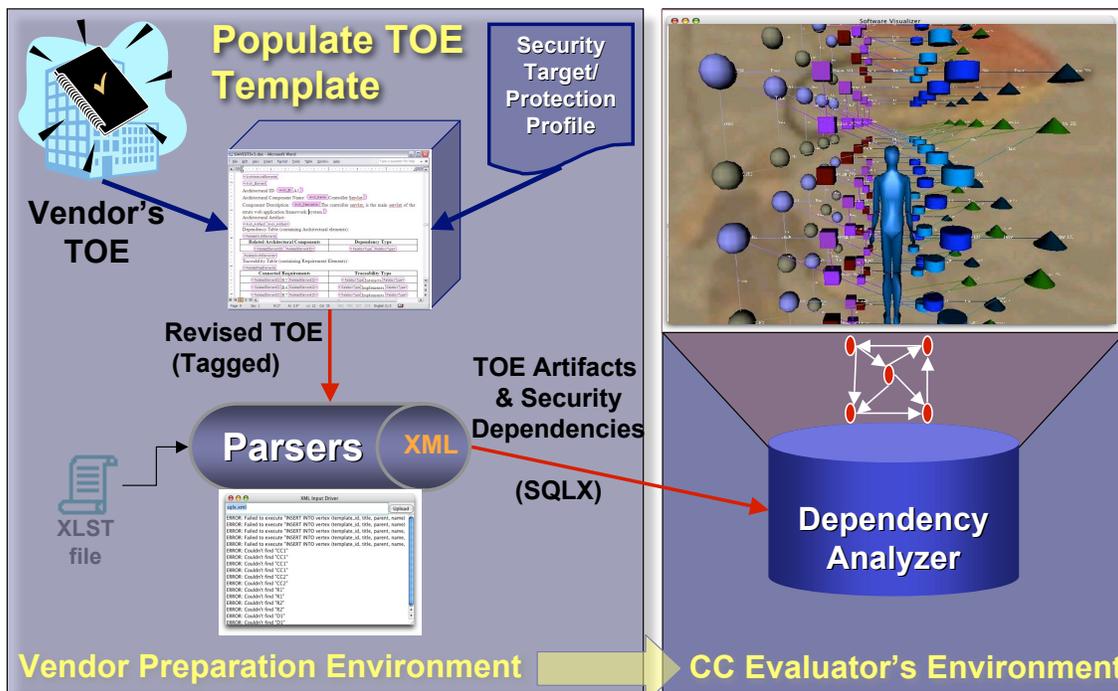
The SIAVE Research Vision



- Vendor Uses CCTool to Generate ST/PP
- ST/PP Used to Generate TOE Template in Vendor's Documentation Environment
- TOE Template Populated and Updated to form the Revised TOE
- Revised TOE Transformed into Software Life Cycle Objects that Populate the Database along with Dependency Relationships
- CC Evaluator Analyzes and Navigates Security Dependency Database in an Immersive Virtual Environment



SIAVE



Shawn Bohner and Denis Gracanic (Funded by Virginia Commonwealth Grant)

Technical Approach

- **Employ Complementary Technologies**
 - Software Impact Analysis (dependency based)
 - Software Visualization / Virtual Environments
- **Two Phase Approach– Evaluator then Vendor**
- **Phase 1: Automation for Evaluator’s Tasks**
 - Security Impacts Model to Analyze Relevant Dependencies
 - Visual Environment for Evaluators
- **Phase 2: Automate TOE capture for Vendors**
 - Build on CCTool to derive TOE templates
 - Start with common Vendor Documentation Tools
 - Templates & Parsers for TOE Capture
 - ST/PP derived TOE Template Generation
 - Capture & Revise TOE in Vendor friendly tools
 - MS Word to XML translation & DBMS population



TOE Template

SIAVEDTSv3.doc - Microsoft Word

File Edit View Insert Format Tools Table Window Help Type a question for help

ArchitecturalElements
Arch_Element

Architectural ID: Arch_ID (A1)

Architectural Component Name: Arch_Name (Controller Servlet)

Component Description: Arch_Description (The controller servlet is the main servlet of the struts web application framework. system)

Architectural Artifact:
Arch_Artifact (Arch_Artifact)

Dependency Table (containing Architectural elements):
RelatedArchElements

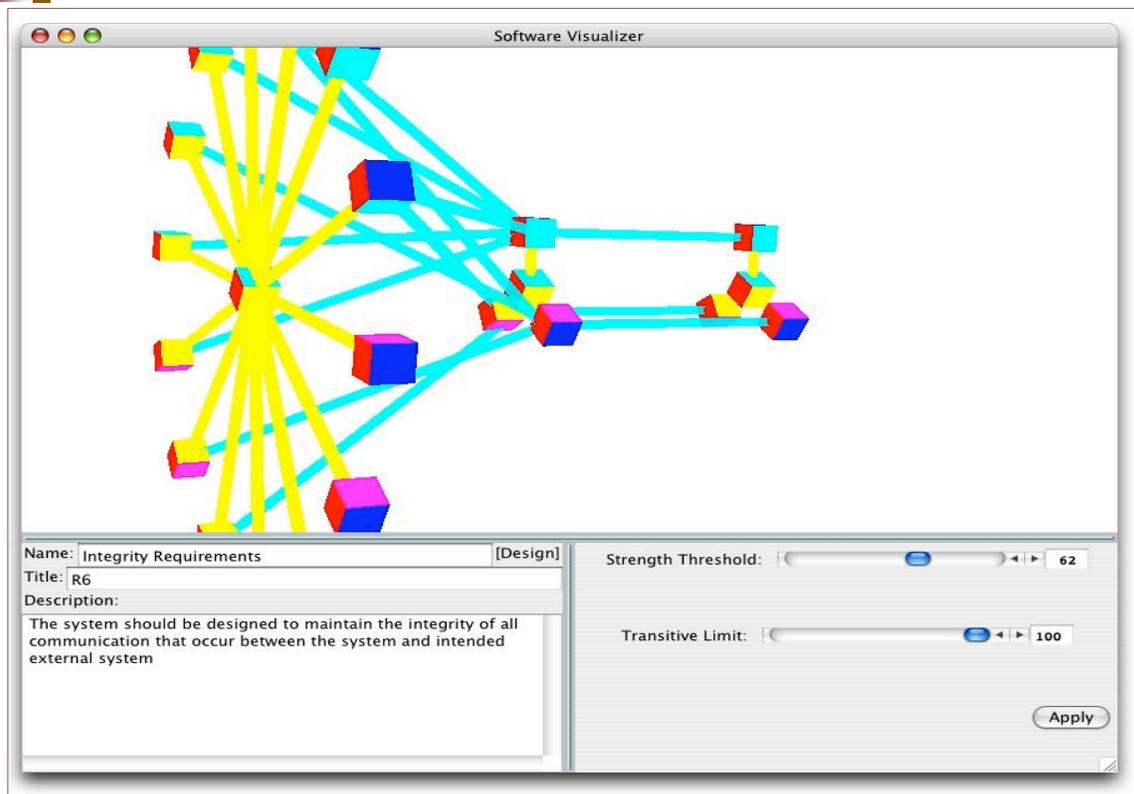
Related Architectural Components	Dependency Type
RelatedElementID () RelatedElementID ()	RelationType () RelationType ()

RelatedArchElements

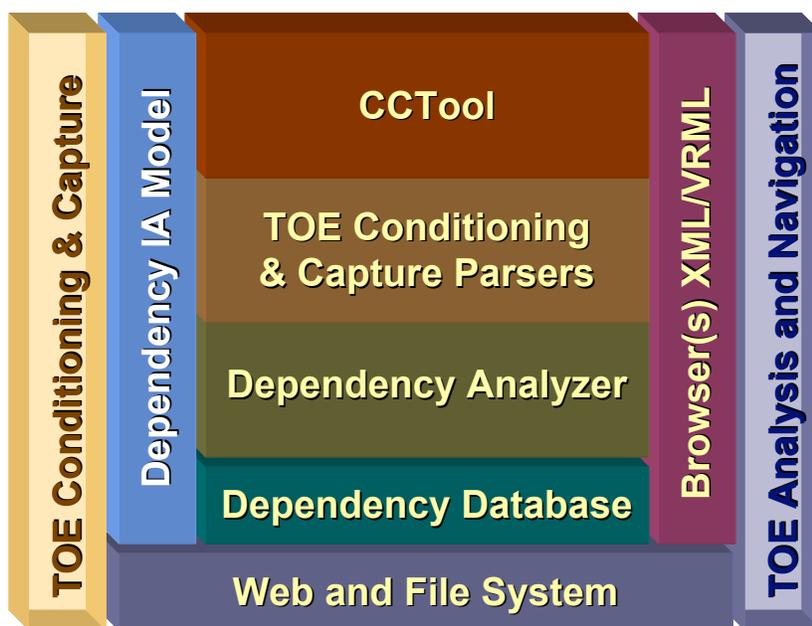
Traceability Table (containing Requirement Elements):
RelatedReqElements

Connected Requirements	Traceability Type
RelatedElementID (R2) RelatedElementID ()	RelationType (Interacts) RelationType ()
RelatedElementID (R4) RelatedElementID ()	RelationType (Implements) RelationType ()
RelatedElementID (R7) RelatedElementID ()	RelationType (Implements) RelationType ()

Page 9 Sec 1 9/17 At 3.9" Ln 12 Col 35 REC TRK EXT OVR English (U.S)



SIAVE Prototype



Evaluation Assurance Levels

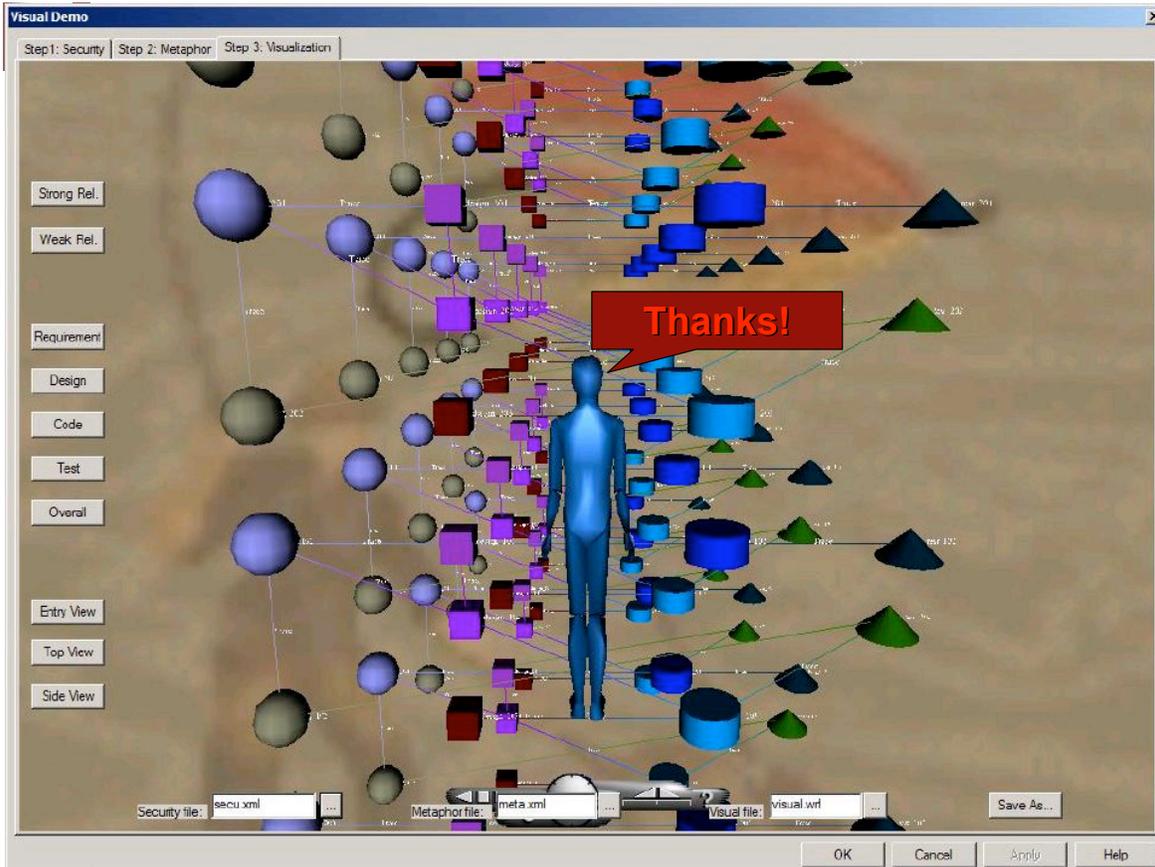
- **EAL1– Functionally Tested:** Basic assurance of security by analyzing functional specifications and guidance.
- **EAL2– Structurally Tested:** Moderate level of assurance by EAL1 plus high-level design and independent testing of the security functions for vulnerability assessment.
- **EAL3– Methodically Tested and Checked:** Provides moderate level of assurance by including EAL2 plus evidence of sound development practices.
- **EAL4– Methodically Designed, Tested and Reviewed:** Moderate/high level of assurance - highest level economically feasible to retrofit an existing product line.
- **EAL5– Semiformally Designed and Tested:** Provides security engineering based upon rigorous commercial development practices to ensure resistance to attackers.
- **EAL6– Semiformally Verified Design and Tested:** High assurance through security engineering techniques in a rigorous development environment to reduce risks.
- **EAL7– Formally Verified Design and Tested:** Highest assurance level - requires formal design verification.



Status and Next Steps

- Completed two Phases of prototype of Evaluator's Visual Environment
- Populated SIAVE with Initial Test TOE
- Refining VE used to Analyze and Navigate TOE artifacts during Evaluation
- **Next frontier** is to introduce Formalism
- Moving into EAL 5-7 with formal specifications
 - Build on Lamsweerde's constructive approach to the modeling, specification, and analysis of application-specific security requirements
 - Consider Specifying Systems in B or VDM++
- Engaging Testing Lab to use live TOE and explore SBIR possibilities





Backup Slides



Prototype Assumptions

- **Proof of Concept Prototype**
 - Navigation and Analysis
 - Assume XML and Java (for now)
 - **Evaluate Dependency Analysis Models**
 - Software Architecture + Security
 - **Experiment with Appropriate Metaphors**
 - Investigator/Explorer
 - Universe/Geographic Space
 - Immersion in Virtual Environment
 - **Establishes Foundation**
 - More Aggressive Analysis and Navigation
 - Formal Specification Analysis
 - Software Architecture Analysis
-



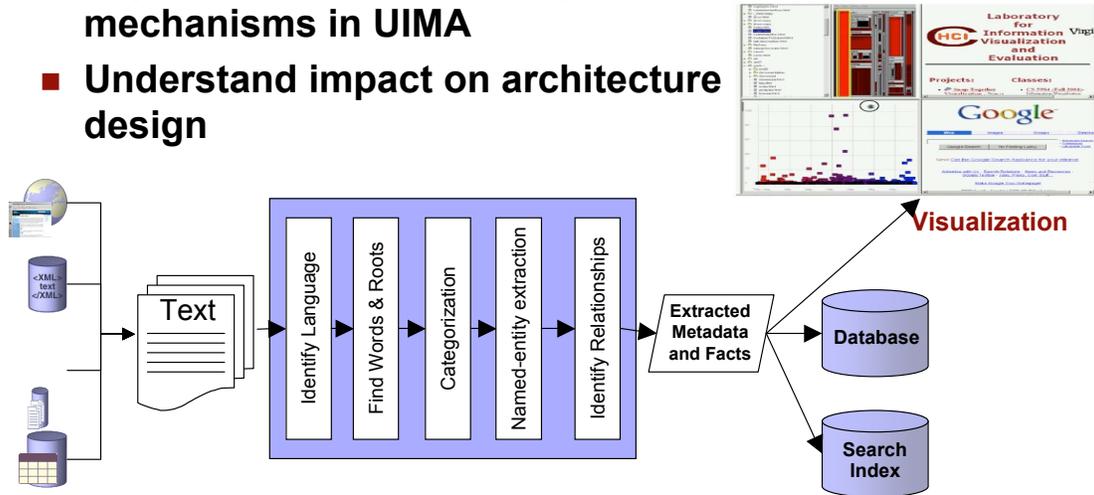
ISO Standards

- **Version 3 of CC released**
 - Substantial changes – more concise and clear
 - **The International community maintains CC as an ISO Standard**
 - ISO 15406 – Common Criteria
 - ISO 18045 – Common Evaluation Methodology
 - ISO 17025 Requirements for Common Criteria Evaluation Lab
 - ISO 15446 – PP/ST Authors Guide
 - ISO 19791 – System Evaluations based on CC product evaluations
 - ISO 15292 – Protection Profile Registration Procedures
-



Integrating Visualization into UIMA

- Integrate Visual Analytic tools in the UIMA Framework
- Explore feasibility of using standard integration mechanisms in UIMA
- Understand impact on architecture design

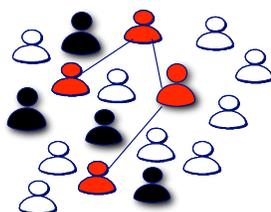


Shawn Bohner Funded by IBM Academic Innovations Grant

Research & Development Experimental Collaboration (RDEC) - Applied Research Center

How can we find relevant information within a mountain of data ?

“Finding the Dots”



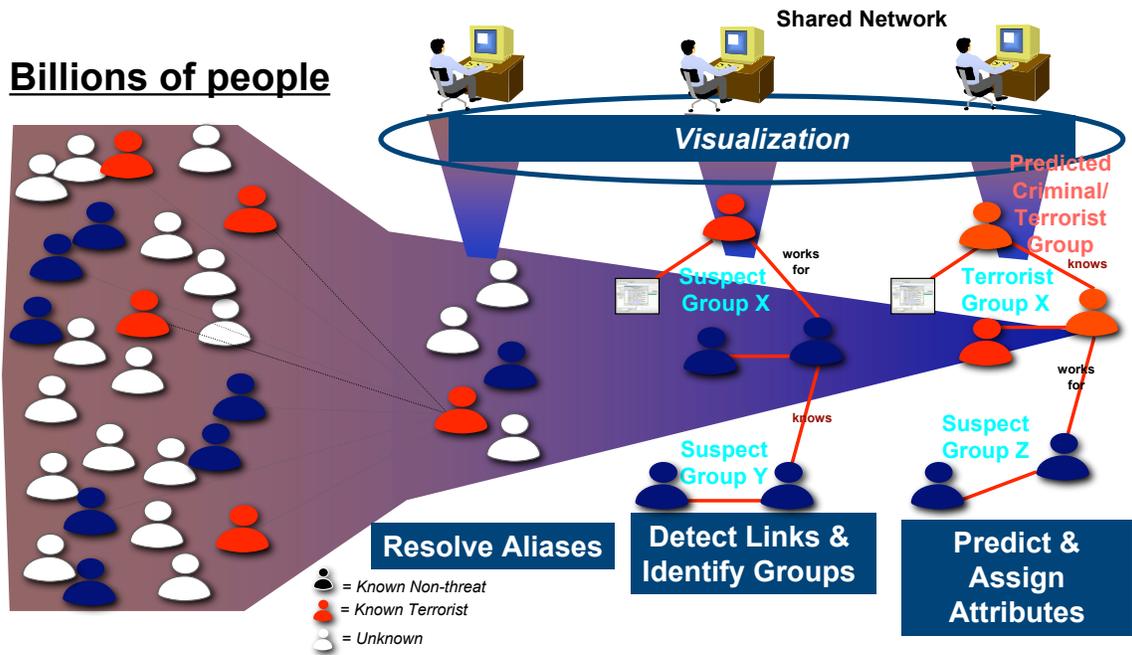
How can we identify those in the threat network?

“Connecting the Dots”



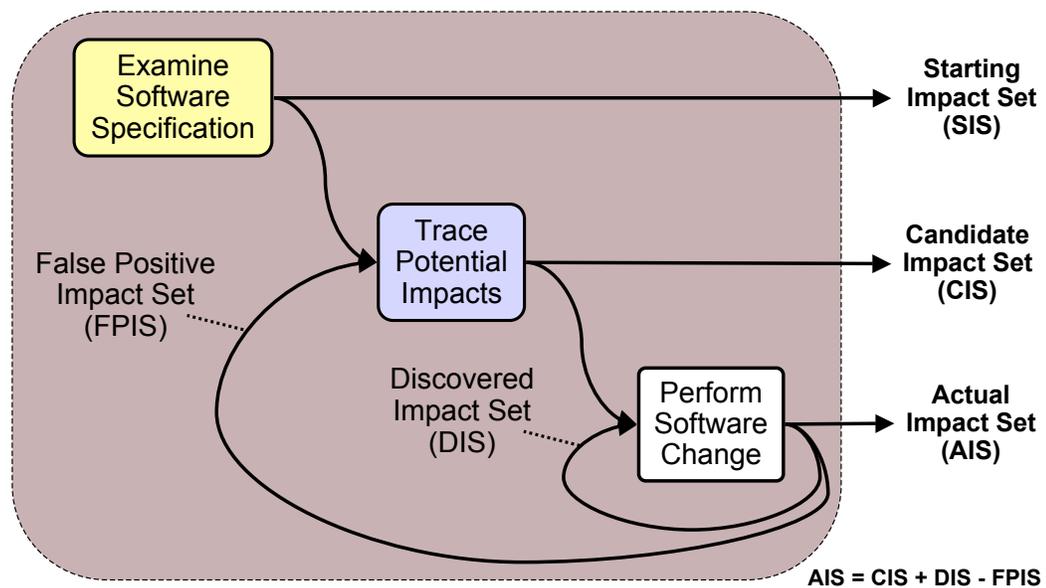
With Nick Stone – Funded by Intelligence Community Grant Through SAIC

RDEC: Visualization of Connected "Dots"

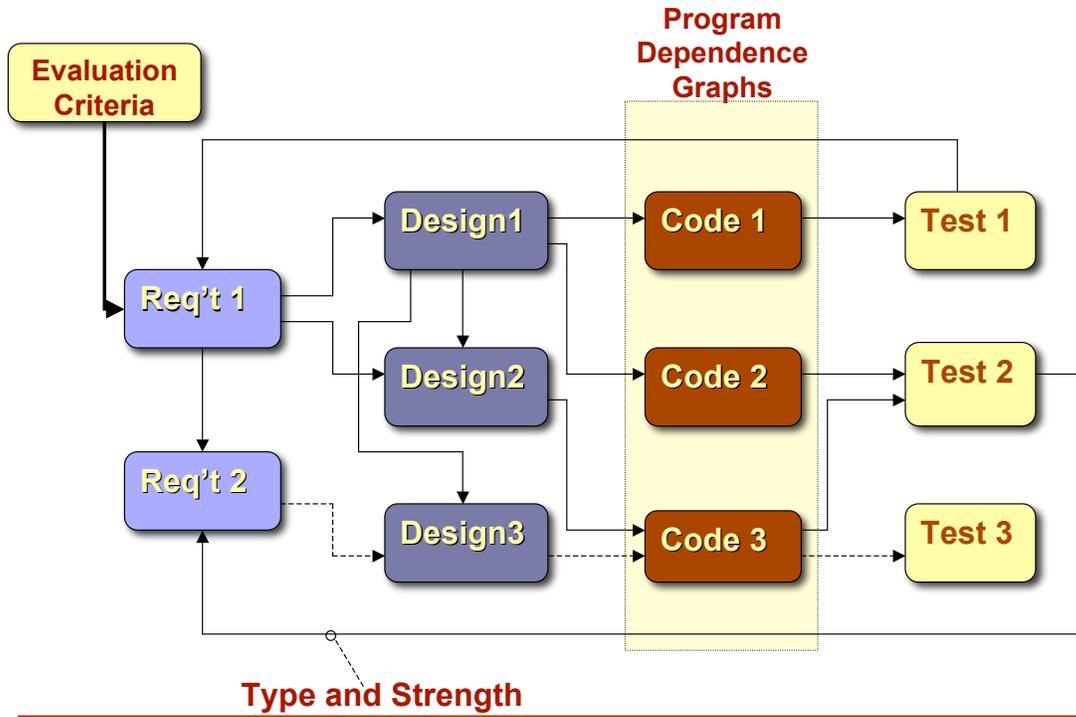


With Nick Stone – Funded by Intelligence Community Grant Through SAIC

Fundamental Software Impact Analysis



Traceability View of Life Cycle Objects



Two Complementary Views for Anomaly-based Intrusion Detection -- Macroscopic and Microscopic

Chin-Tser Huang
Department of Computer Science and Engineering
University of South Carolina
huangct@cse.sc.edu

1. Introduction

Network-based Intrusion Detection Systems (NIDS) are of increasing significance because of their role as the frontline of an entire network of computers. These systems primarily belong to one of two types: *signature-based* and *anomaly-based*. A signature-based NIDS, such as Bro [1] or Snort [2], examines network traffic in an effort to match the patterns of the traffic, or rules, to preestablished patterns of malicious activity. Such systems provide excellent detection capabilities against the known attacks, but require constant update to provide protection from new attack strategies. An anomaly-based NIDS works on the assumption that malicious network traffic is distinguishable from normal network traffic, as discussed in [3]. These systems attempt to quantify the protected network's "normal" network traffic and reports deviations from this norm.

Anomaly-based detection has attracted major research interest, since it has the ability to detect novel attack strategies that are often missed by signature-based methods. By understanding and defining what is "normal" in a network, deviations from this norm indicate activities that require further investigation. This method of detection maintains the same level of sensitivity in the presence of novel and classic attack strategies. However, current anomaly-based intrusion detection systems also face the following challenges:

- How to keep the advantages of anomaly-based detection while reducing the false alarms?
- How to lower the overhead and detect anomalies in a timely fashion?
- How to automatically differentiate and categorize the detected anomalies?
- How to hold attacking hosts accountable for their behavior?

Our research efforts aim to construct two separate but complementary views of the network's state in order to address the above challenges and provide a comprehensive and precise interpretation. The first is a *macroscopic* view, in which the overall network traffic is viewed as time-series signal. We apply wavelet-based technologies to expose the anomalies with the normal traffic regarded as the noise. We develop a framework called Waveman, which use an open source tool called LastWave [4] to provide a real time analysis of network traffic. The second is a *microscopic* view, in which network is viewed as a collection of individual hosts. We apply an adaptive algorithm to charge individual host for anomalous behavior. We develop an anomaly-based NIDS, Fates, which attempts to alleviate the challenges specified above while maintaining the advantage of detecting novel attacks. Fates has the ability to differentiate between characteristics of individual hosts and independently assess their threat to the network.

2. A Macroscopic View with Waveman

In any NIDS, it is desirable to automatically detect and categorize the anomaly in real time, such that less human interference is required and more response time is gained. To this end, we design and implement a Waveman framework, as shown in Figure 1, to carry out a real time wavelet analysis which is also used in [5]. Traffic is captured at an available interface using libpcap. Two counters corresponding to packet and byte counts are incremented on a per packet basis. To manage the capturing and sampling, two processes are used: one to capture the traffic on a per packet basis and update the appropriate byte and packet counters, and the other to access these counters via shared memory (`shmget()`), every 5 seconds.

Next, a time series signal of packets vs. time (sampled every 5 seconds) is built, prepared and sent to LastWave. Since LastWave can also be used as a scripting language, we develop our own scripts for the analysis, which are executed on a per analysis basis. The first three coefficients are of value to us (since any greater coefficients of the

analysis would contain very sparse information), and these are calculated as the output of LastWave (Coeff1, Coeff2, Coeff3 in Figure 1).

LastWave output is then processed, for purpose of normalization and ease of calculation of percentage deviations. The window we work with is five minutes long; i.e. five minutes worth of traffic, sampled every five seconds (these values are consistent with general network monitoring practices). Hence our window contains sixty samples. This size is consistent with the fact that a small window is good for localization. Several intermediate scripts are written in Perl to process and prepare the data for the next phase. The percentage deviations are calculated and recorded at each analysis. These values are normalized for ease of comparison.

In the last stage, Gnuplot is used to plot the graphs in the form of JPEG files, and an Apache web server is used to serve the current results of the analysis to remote viewers. The graphs were plotted every five seconds by default, providing an updated real time snapshot of the current analysis every five seconds. Most of the framework and analysis work was done on a Pentium 4 (Hyperthreaded), 1 GB RAM, Gigabit interface NIC, running Fedora Core 3, and initial development and testing was done on a Dual Xeon (Hyperthreaded), 1 GB RAM, Gigabit NIC, running RHEL 3.

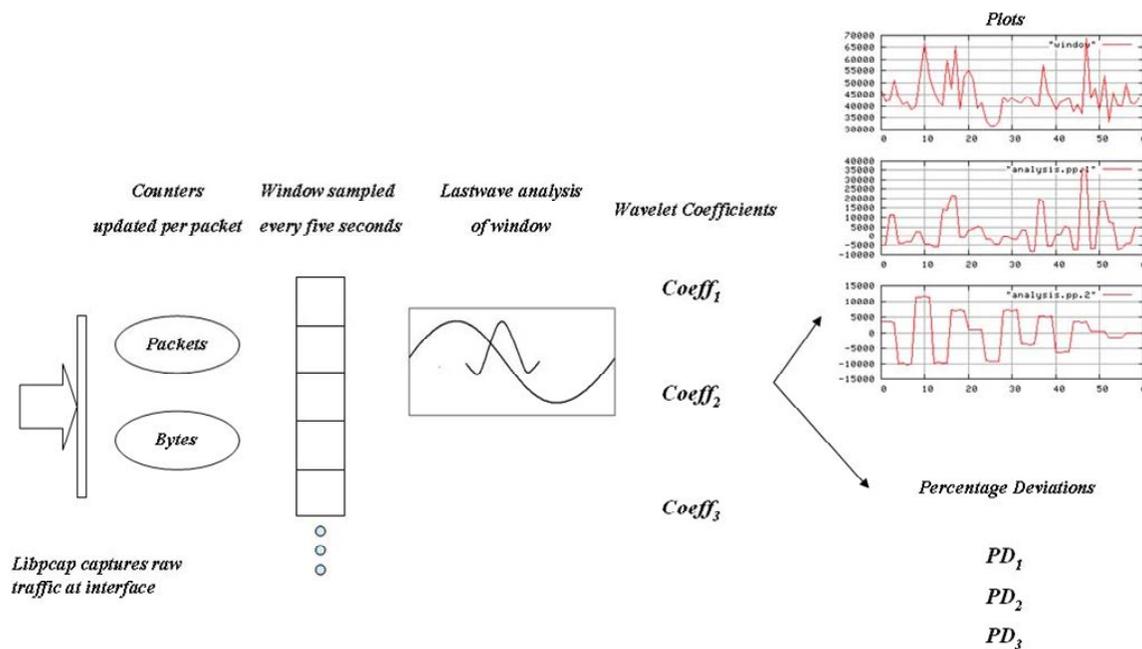


Figure 1. Framework of Waveman, a wavelet-based real time network traffic analysis.

One of the most important and distinct advantages of Waveman is it can use any wavelet function in its analysis. This feature allows us to apply different wavelet functions on analyzing the same anomalous traffic trace and evaluate their performance. Evaluation of the wavelets is based on twofold criteria: to have good localization in time characteristics, and to have a low mean deviation over the duration of the signal. Moreover, we can use Waveman to determine if there is a correlation that some wavelet function performs better in detecting some specific type of network attack or anomaly.

In order to evaluate the performance of various wavelet functions on detecting different types of anomalies, we employ two metrics, namely *percentage deviation* and *entropy*. The percentage deviation of the coefficient value measures the average deviation from the median for all the sample points in the coefficient. The rationale behind this is that those coefficients that display a lower PD are better, because the amount of deviation from the origin is indicative of an anomaly. To be more specific, a “better” wavelet should show a larger deviation at the locations of the start and end of an anomaly and show smaller deviations at all other locations in the signal, such that the contrast is larger and the anomaly is more identifiable. Entropy is a type of information measure of disorder in signals and systems. A spontaneous change in a system disperses energy and increases its entropy, which is indicative of an anomaly. From the experimental results we derive the conclusion that the percentage deviation-based method and

the entropy-based method generate consistent evaluation results in our tests and they can both be used to evaluate the effectiveness of the wavelets on detecting and analyzing network anomalies.

We use four widely used wavelet functions, namely Coiflet, Morlet, Daubechies, and Paul, to analyze traffic traces from MIT Lincoln Lab IDS evaluation dataset [6] and a domain name service company that contain five types of attacks/anomalies, namely Neptune, Smurf, Mailbomb, a simple portscan (ipsweep), and a stealth scan. The evaluation results show that Coiflet and Mexican Hat wavelets have better characteristics when faced with the anomalies considered in this work, based on a five-minute, sixty-sample window.

3. A Microscopic View with Fates

One insufficiency of the above approach is that it does not provide information about the source of attack or anomaly. This information is desirable because we can use it to apply corrective or containing schemes on the misbehaving host in order to minimize the impact on other hosts in the network. In an effort to provide both the granularity and the economy of operations required in modern networks, we develop a microscopic view of the network traffic. It is instructive to compare this approach with the related works by Jung et al. [7] and Weaver et al. [8], which use a Threshold Random Walk (TRW) scheme to assess the health of the network according to the likelihood ratio for packet delivery or the number of unacknowledged connection attempts a host makes. The health of a host is represented by a single value, and if this value exceeds a predefined threshold, the system disallows any new connection attempts. However, both [7] and [8] fail to distinguish between varying traffic needs. The thresholds they use are static and global, therefore these systems are unable to adequately represent a network of diverse traffic needs. In Fates, we incorporate dynamic, individual thresholds for each monitored host. As a result, Fates is able to independently assess individual host's health while keeping the processing load economical.

Fates examines the network as a collection of individual entities using three subsystems: a sniffer (Clotho), a measuring unit (Lachesis), and an alarm unit (Atropos). Clotho is a passive listener that records packets as they enter and leave the network. Lachesis, utilizes the granular view in internal-to-external monitoring. This is achieved with an internal hosts monitor component (IHM), which uses connection classification in order to assess the overall health of a specific monitored host. Atropos generates alarms according to the analysis result of Lachesis and the alarming policy defined by the administrator.

The IHM component utilizes both the a priori IP address information provided at initialization and current connection state information to produce an analysis of individual hosts in the network. Prior to active monitoring of the network, the measuring unit acquires a list of active IP addresses (or range of addresses) in the monitored subnet and the minimum thresholds of the host (or range of hosts). Fates regards each IP address or range of addresses as a separate unit with its own threshold and scoring so that it can differentiate between various traffic needs for a variety of hosts and support any number of protected hosts and any degree of granularity. The minimum threshold is the lowest sustainable threshold that Fates allows the host to have and uses the minimum threshold to adjust the current threshold of the host.

When IHM processes an IP packet, it first determines if the upper-layer protocol is connection-oriented, such as TCP, or connectionless, such as UDP. In the case of TCP, the state of the connection is of primary concern. The IHM component determines whether the packet is destined to or originated from a monitored host and the packet type. If the destination of the packet is a monitored host, the IHM component first finds from the IP_List the element corresponding to the destination address, uses the source IP address to index into the element's I/OCache, and then subtracts one from the I/OCache entry's current value (conversely, if the source of the packet is a monitored host, add one to the corresponding entry). The IHM component then assesses a charge for the packet using the entry's resulting value. The formula for calculating this charge is shown in Table 1. If the value of the entry is less than or equal to zero, the state is set equal to zero and the host is not assessed a charge. If the value of the entry is greater than zero, the state is set equal to the entry's value. The reason for the multiplication of the state information by two is to provide a quick jump in charges in the presence of persistent unacknowledged outgoing messaging. Note that in a standard three-way handshake and packet transmission (the destination transmits an ACK for each message received) the monitored host receives a net charge of zero. In the case of a UDP packet, the packet's payload is of importance because there is no connection information associated with protocol. When the IHM component processes a UDP packet, it uses the payload of the packet to index the IP_Packet_Table, increments the entry's count value by one, and sets the TTL of the entry to 255. If the source of packet is a monitored host, the IHM component then assesses

the host a charge. Note that an arbitrary non-duplicate packet would result in no charge. In the case of any other protocol, Fates skips the packet as the design of Fates is for standard practice. ICMP packets are also skipped because the same type of ICMP packets have identical payload and present an ambiguity to Fates.

Table 1. Formulas for packet charge.

Packet Type	Formula
TCP	Charge = $2 * (state - 1)$
UDP	Charge = $2 * (count - 1)$

At the expiration of each time step, the IHM component assesses the health of all monitored hosts by calculating the cumulative charge for all packets for each host seen during the current time step, resulting in a threat score for the host. The IHM component compares the threat score to the current threshold of the host. If the threat exceeds the current threshold, the IHM sets the threshold equal to the threat score and makes a note of the change in a log file. If the threat is less than the threshold, the IHM component compares the threshold with the minimum threshold. If the values are equal, the IHM component takes no action. In all other cases, the IHM component uses a threshold adjustment scheme. A threshold is easily increased but further analysis is required to determine if the threshold should be lowered. The principle idea is that the component attempts to ascertain an appropriate upper bound of a host's activity. A well-behaved host's threshold will plateau, but a scanning host's activity constantly causes the host's threshold to increase. In the IHM component's threshold adjustment, the threshold will remain the same until being exceeded by a host's score. Once a host's score exceeds the host's threshold, the value of the host's threshold will increase to the score that exceeded it. For every time step afterward, if the weighted average score of the host is lower than the minimum threshold, then the threshold value decreases by half of the difference between the minimum threshold and the weighted average score until it reaches the minimum threshold value. After the IHM component adjusts the thresholds of each host it then prepares for the next time step by resetting the threat score to zero, decreasing the TTL of each entry in the I/OCache by one, and decreasing the TTL of all elements in the IP_Packet_Table by one. If the TTL of an entry in the IP_Packet_Table is equal to zero, the IHM component sets the count of the entry to zero.

We test the Fates system on several different datasets in order to understand how the system functions under environments with different characteristics. The results show that Fates provides an accurate analysis of the current state of a network with regard to scanning behavior. Furthermore, Fates does not falter in the presence of lost acknowledgements. Instead, it tolerates occasional packet losses without instantaneous flagging of the host as malicious. At present Fates is intended to serve a small to medium sized network environment, but we will continue to investigate the scalability issue of Fates.

References

- [1] Bro Intrusion Detection System, available at <http://bro-ids.org/>
- [2] Snort, available at <http://www.snort.org/>
- [3] D. Denning. "An intrusion detection model". In Proceedings of the 1986 IEEE Symposium on Security and Privacy, pp 119–131, 1986.
- [4] E. Bacry, LastWave 2.0, available at <http://www.cmap.polytechnique.fr/~bacry/LastWave/index.html>
- [5] P. Barford, J. Kline, D. Plonka, and A. Ron, "A signal analysis of network traffic anomalies," Proceedings of ACM SIGCOMM Internet Measurement Workshop, Marseille, France, 2002.
- [6] J. W. Haines, R. P. Lippmann, D. J. Fried, E. Tran, S. Boswell, and M. A. Zissman, "1999 DARPA intrusion detection system evaluation: Design and procedures," In MIT Lincoln Laboratory Technical Report, 2000.
- [7] J. Jung, V. Paxson, A. Berger, H. Balakrishnan. "Fast Portscan Detection Using Sequential Hypothesis Testing". In Proceedings of 2004 IEEE Symposium on Security and Privacy, 2004.
- [8] N. Weaver, S. Staniford, V. Paxson. "Very Fast Containment of Scanning Worms". In Proceedings of the 13th USENIX Security Symposium, pages 29-44, 2004.



Two Complementary Views on Intrusion Detection

-- Macroscopic and Microscopic

Chin-Tser Huang

Department of Computer Science and Engineering
University of South Carolina

Computer Science and Engineering @ University of South Carolina



Network Activity

- **Benign Traffic** – Network traffic that should not result in a network compromise
 - Web Browsing, E-mailing, etc.
- **Malicious Traffic** – Any activity intended to result in a compromise of a network entity
 - Scanning, DoS, Session Hijacking, etc.

Computer Science and Engineering @ University of South Carolina



Network Intrusion Detection Systems

- Systems that look for malicious activities in a network environment
- Common classifications:
 - Signature/misuse-based
 - Anomaly-based
 - Hybrid

Computer Science and Engineering @ University of South Carolina



Signature/Misuse-Based Detection

- Attempts to fit malicious traffic characteristics to specific signatures
- Advantage
 - Very good at detecting known attacks
- Disadvantages
 - Can completely overlook novel attacks
 - Must constantly be updated

Computer Science and Engineering @ University of South Carolina



Denning's Assumption

- Malicious traffic is distinct from benign traffic
 - These differences are measurable
 - Example: Scanning has low probability of resulting in an established connection

Computer Science and Engineering @ University of South Carolina



Anomaly-Based Detection

- Treats benign traffic as norm
- Advantage
 - Can detect novel attacks
- Disadvantage
 - High false alarm rates
 - Costly computations

Computer Science and Engineering @ University of South Carolina



The Challenges

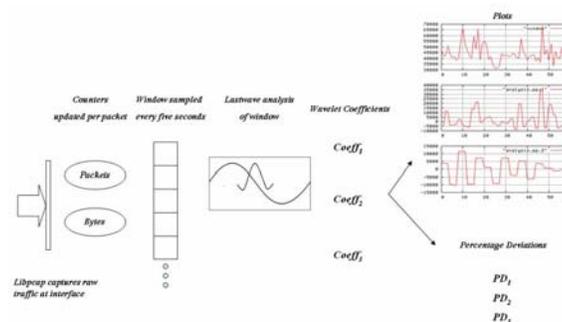
- How to keep the advantages of anomaly-based detection while reducing the false alarms?
- How to lower the overhead and detect anomalies in a timely fashion?
- How to automatically differentiate the detected anomalies?
- How to hold attacking hosts accountable?

Computer Science and Engineering @ University of South Carolina



Two Complementary Views

- A **macroscopic** view
 - view network traffic as time-series signal
 - use wavelets to capture different types of anomalies

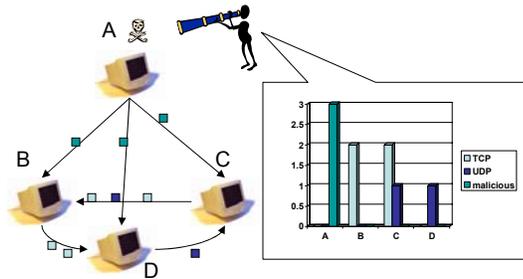


Computer Science and Engineering @ University of South Carolina



Two Complementary Views

- A **microscopic** view
 - view network as a collection of individual hosts
 - charge individual host for anomalous behavior



Computer Science and Engineering @ University of South Carolina



Macroscopic View

- Motivation
 - Perception at different detail levels, in close-to-real time
 - Applications include evaluation of security features, and for monitoring purposes
 - Build an Intrusion Detection System based on wavelet analysis

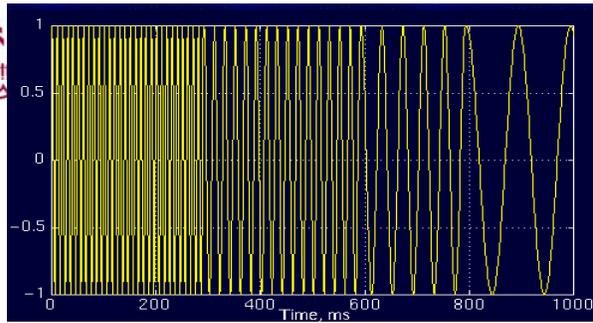
Computer Science and Engineering @ University of South Carolina



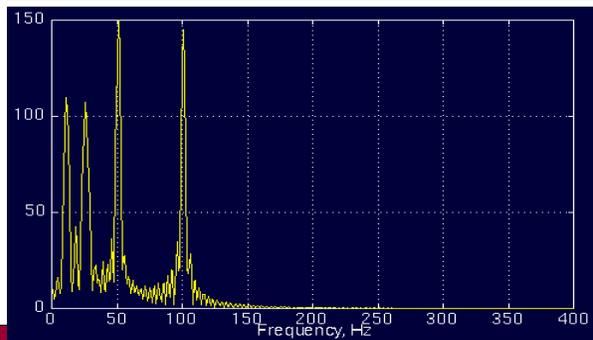
Macroscopic View

- Related works
 - “A Signal Analysis of Network Traffic Anomalies”, Paul Barford, Jeffery Kline, David Plonka and Amos Ron, ACM SIGCOMM Internet Measurement Workshop 2002
 - “A Wavelet-Based Approach to Detect Shared Congestion”, Min Sik Kim, Taekhyun Kim, Yong-June Shin, Simon S. Lam, and Edward J. Powers, ACM SIGCOMM 2004

Computer Science and Engineering @ University of South Carolina

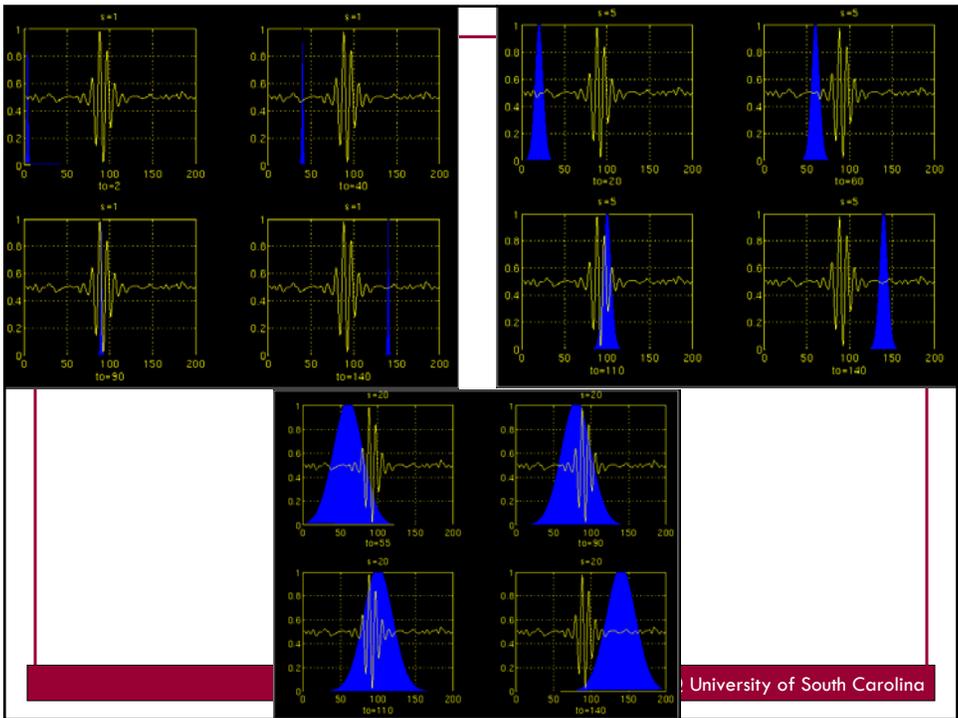
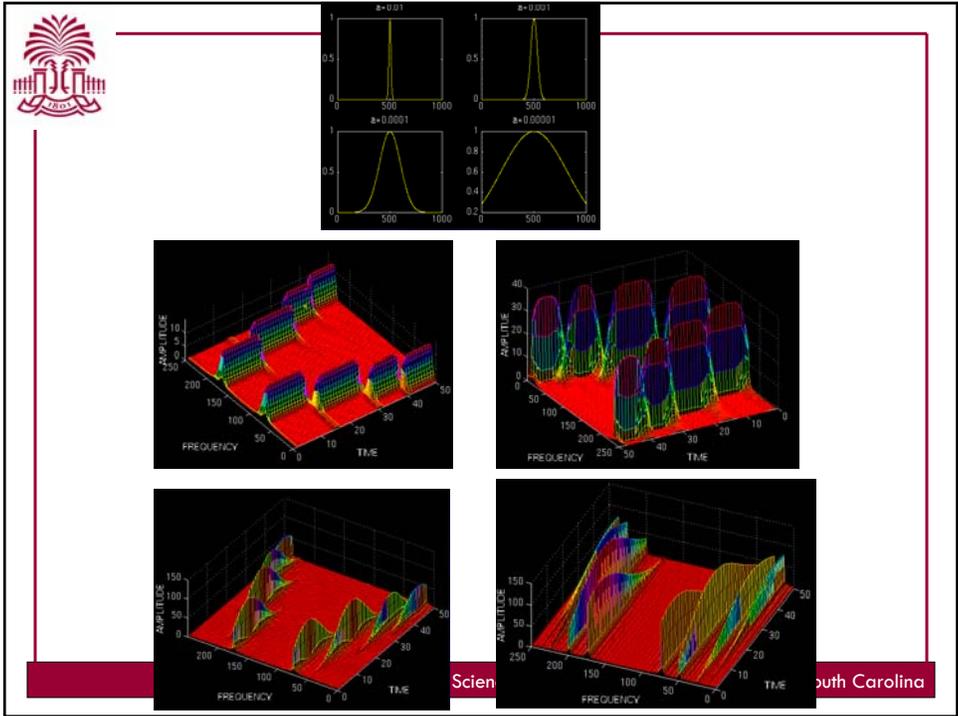


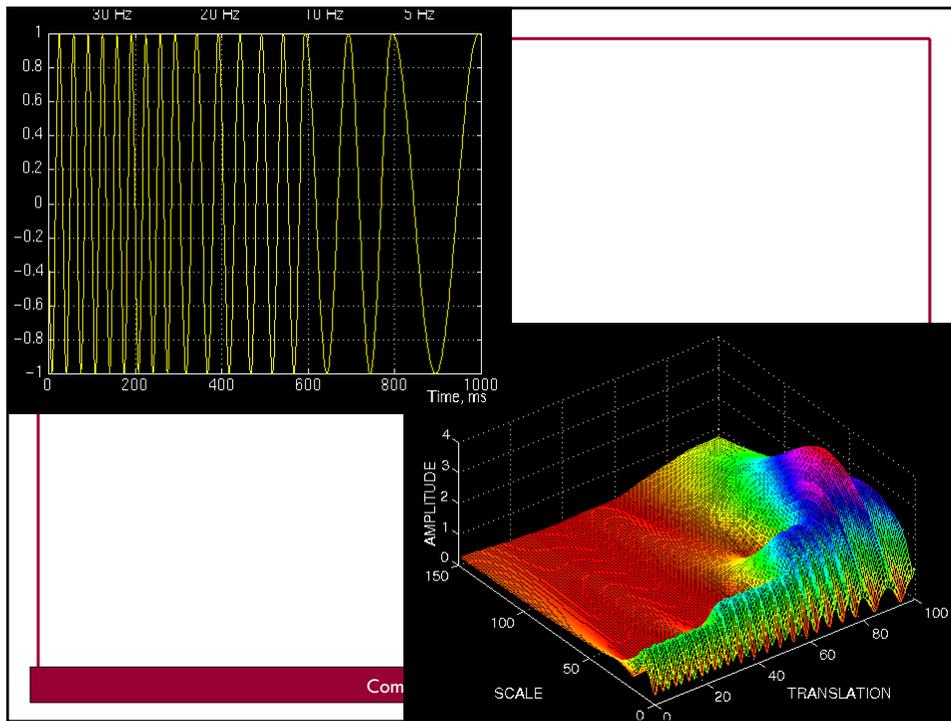
Frequencies?



Frequencies:
100, 50, 25, 10 Hz

Computer Science and Engineering @ University of South Carolina







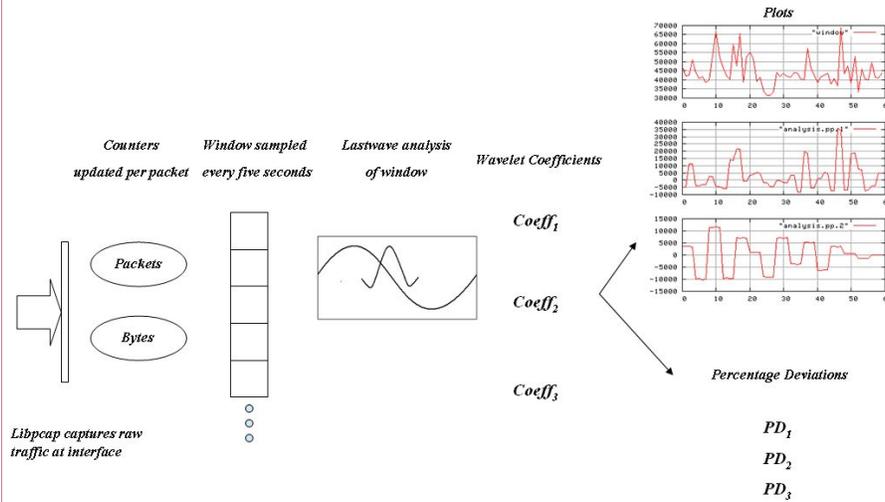
Analysis

- Iterative process (**Subband coding** or **Multi Resolution Analysis**):
 - Input for each iteration: a signal x of length N
 - Output: a collection of two, more derived signals, each of length $N/2$
 - Each output obtained by
 - convolving x with a specially designed filter F
 - decimating every other coefficient
 - $F(x)$ is the output signal
 - Special Filter L has a smoothing/averaging effect
 - corresponding output *low-frequency* output
 - Other filters, $H_1 \dots H_r$: discrete differentiation
 - output $H_i(x)$ should capture only the “fine-grained details”
 - Iterations proceed with the further decomposition of $L(x)$, creating the (shorter) signals $L^2(x); H_1 L(x) \dots H_r L(x)$
- We obtain a family of output signals of the form $H_i L^{-1}(x)$

Computer Science and Engineering @ University of South Carolina



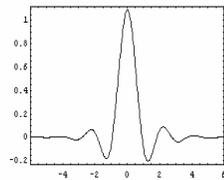
Framework



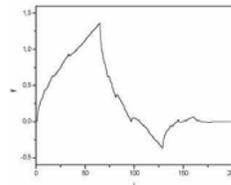
Computer Science and Engineering @ University of South Carolina



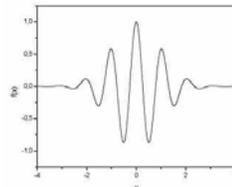
Wavelets used



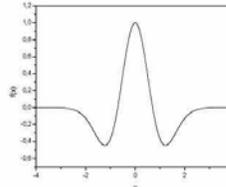
(a) Coiflet
Lengths: 11, 21, 41, 61



(b) Daubechies
Lengths: 6, 11, 21



(c) Morlet
Lengths: 15, 30, 40



(d) Mexican hat
Lengths: 15, 30, 40

Computer Science and Engineering @ University of South Carolina



Datasets

- MIT Lincoln Laboratory Intrusion Detection System Evaluation (1999)
 - Neptune
 - Smurf
 - Mailbomb
- EnetRegistry Inc. (2004-2005)
 - Portscan
 - Stealthscan

Computer Science and Engineering @ University of South Carolina



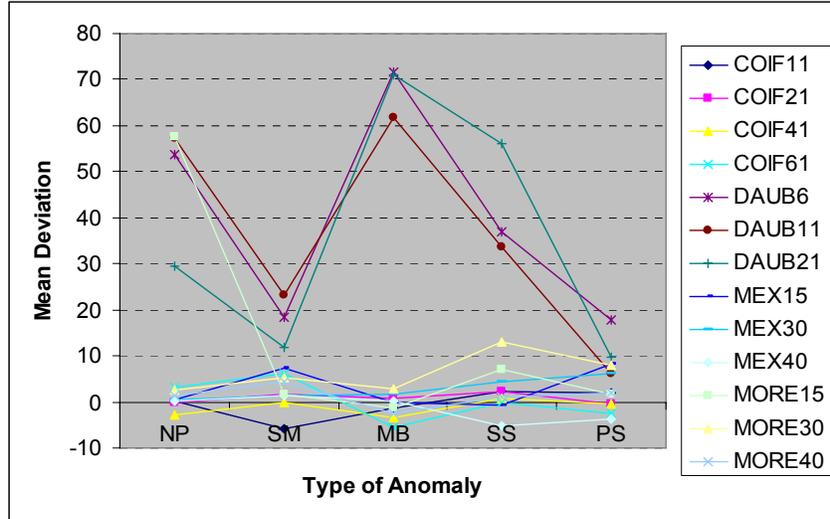
Evaluation

- Established anomalies
- Percentage Deviation: low value for the length of the anomaly is better
- Localization in time characteristics

Computer Science and Engineering @ University of South Carolina



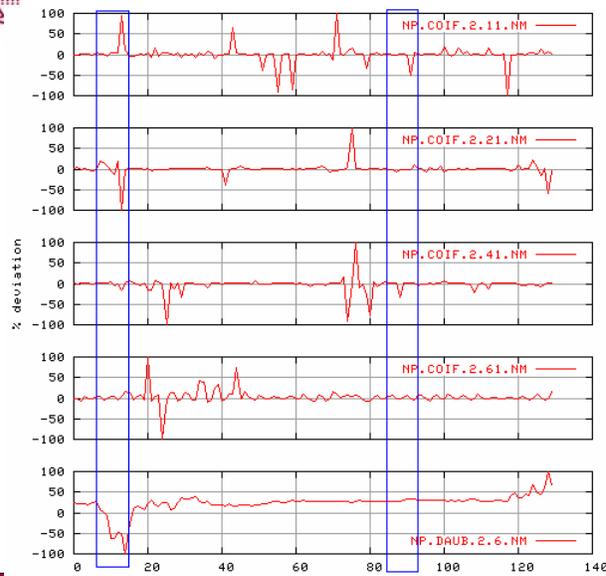
Results: Deviation Characteristics



Computer Science and Engineering @ University of South Carolina

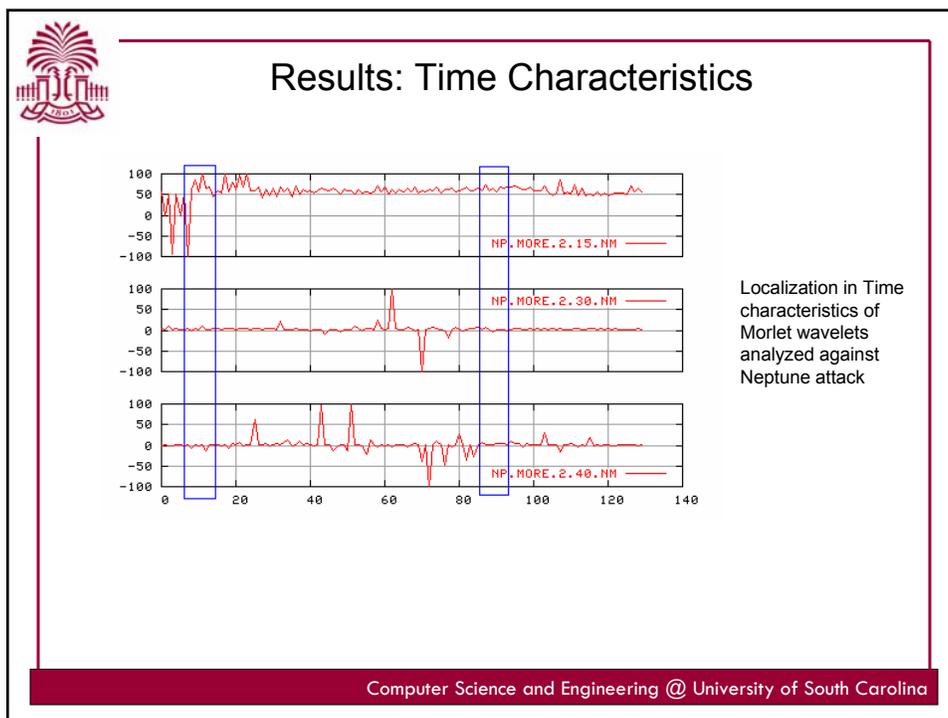
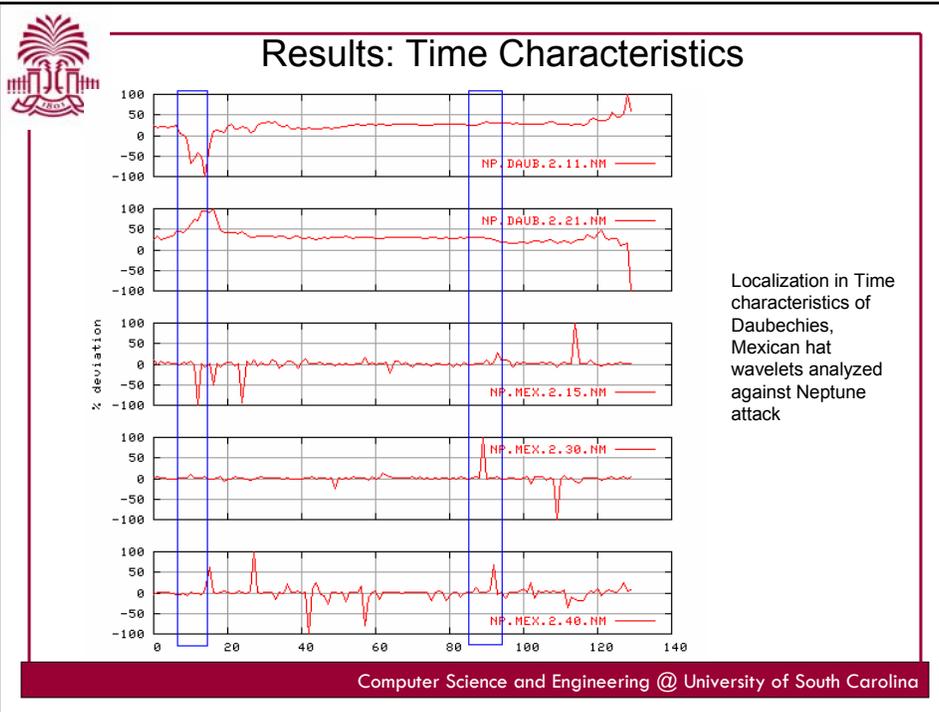


Results: Time Characteristics



Localization in Time characteristics of Coiflet, Daubechies wavelets analyzed against Neptune attack

Computer Science and Engineering @ University of South Carolina





Results Summary

- Based on
 - Window length of five minutes
 - Lengths of filters,Coiflet wavelet and Mexican Hat wavelets show good characteristics for anomalies analyzed
- Daubechies shows weakest characteristics for both localization in time and mean deviation

Computer Science and Engineering @ University of South Carolina



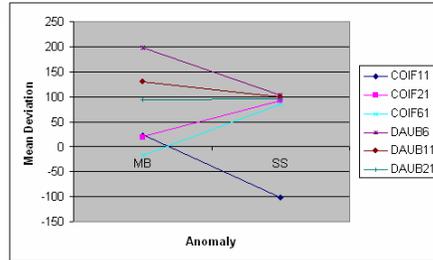
Next Step

- Varying window sizes
 - Anomalies are of varying sizes, need to be analyzed using different window sizes
- Other methods of evaluation
 - Entropy based
- Some preliminary results

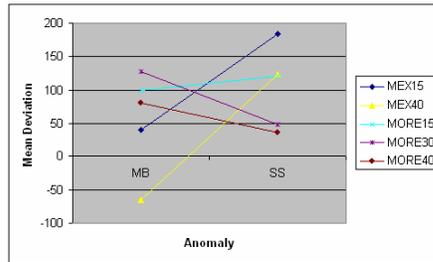
Computer Science and Engineering @ University of South Carolina



Varying Window Sizes



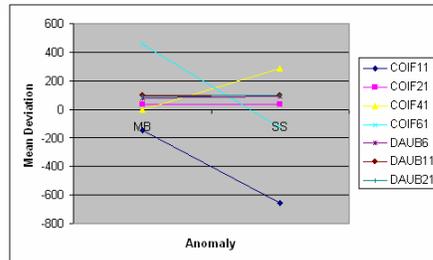
Mailbomb and Stealth scan anomalies analyzed using a window length of two minutes



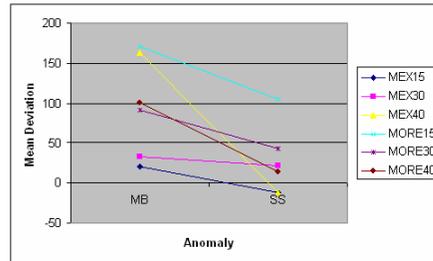
Computer Science and Engineering @ University of South Carolina



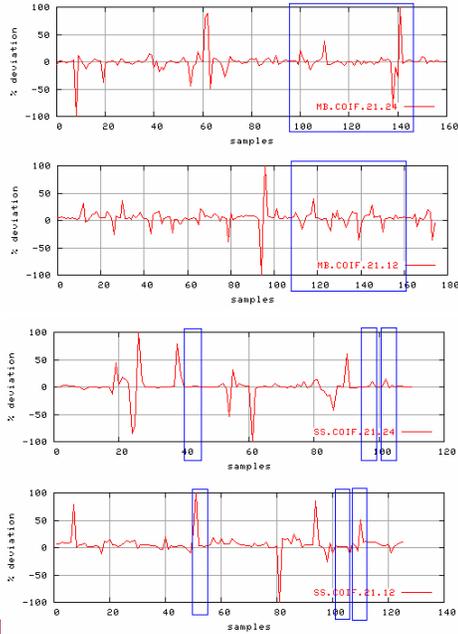
Varying Window Sizes



Mailbomb and Stealth scan anomalies analyzed using a window length of one minute



Computer Science and Engineering @ University of South Carolina



a) Mailbomb,
Coiflet,
window
lengths 24, 12

b) Stealth scan,
Coiflet,
window
lengths 24, 12

@ University of South Carolina

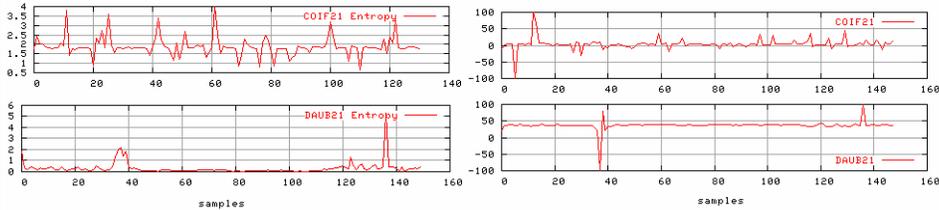


Entropy Based Evaluation

Entropy:
$$H_r(x) = \frac{1}{1-r} \log \left(\int f^r(x) dx \right), \quad 0 < r < \infty, r \neq 1$$

Rényi Entropy:
$$H_r(x) = -\frac{1}{2} \ln \left(\frac{1}{n} \sum_{i=1}^n f^r(n) \right)$$

Neptune Attack, Coiflet and Daubechies Wavelets, window length one minute



Entropy Based

Percentage Deviation Based

Computer Science and Engineering @ University of South Carolina



Summary

- Real Time analysis
 - Generate signal from network traffic
 - Windowed analysis by subband coding/MRA
 - Evaluation of five anomalies from two datasets: low mean deviation, good localization in time
 - Coiflet and Mexican Hat wavelets show overall good characteristics, Daubechies shows poorest
- Implications:
 - Perception at different detail levels, in real time
 - Applications include evaluation of security features, and for monitoring purposes
 - Intrusion Detection System

Computer Science and Engineering @ University of South Carolina



Microscopic View

- Motivation
 - Provide pinpointed analysis of anomalous activity at individual host
 - Keep computation overhead and memory consumption low
- Related works
 - Threshold Random Walk
 - Very Fast Containment of Scanning Worms

Computer Science and Engineering @ University of South Carolina

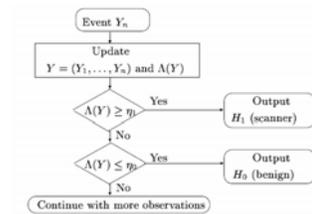


Threshold Random Walk

- Sequential hypothesis testing

- $Y=0 \rightarrow$ success
- $Y=1 \rightarrow$ failure
- $H_0=$ benign
- $H_1=$ malicious

$$\Lambda(Y) \equiv \frac{\Pr[Y|H_1]}{\Pr[Y|H_0]} = \prod_{i=1}^n \frac{\Pr[Y_i|H_1]}{\Pr[Y_i|H_0]} \quad (3)$$



Computer Science and Engineering @ University of South Carolina



Very Fast Containment of Scanning Worms

- A simplified version of TRW
- View the network as a collection of autonomous regions
- Uses approximation caches to limit memory consumption
- Counts the number of un-established connections

Computer Science and Engineering @ University of South Carolina



Fates

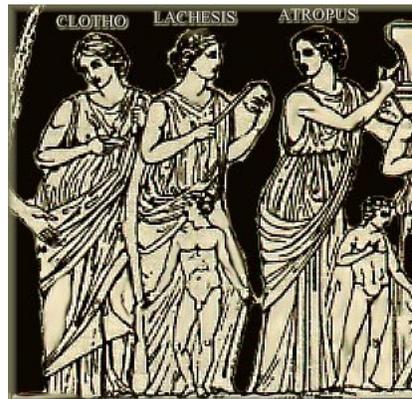
- Common features between Fates and both of these approaches
 - Granular view of the network
 - Examines state of connections
- Differences
 - Thresholds are dynamic
 - Charges are additive
 - Monitored hosts are always suspect

Computer Science and Engineering @ University of South Carolina



Fates Overview

- Three components
 - Clotho the Weaver
 - Lachesis the Apportioner
 - Atropos the Cutter of Threads



Computer Science and Engineering @ University of South Carolina



Fates Overview

- Three components
 - Clotho the Weaver – Packet sniffer
 - Captures packets
 - Lachesis the Apportioner – Packet analyzer
 - Assesses charges to each host
 - Atropos the Cutter – Alarming mechanism
 - Produces human readable analysis

Computer Science and Engineering @ University of South Carolina



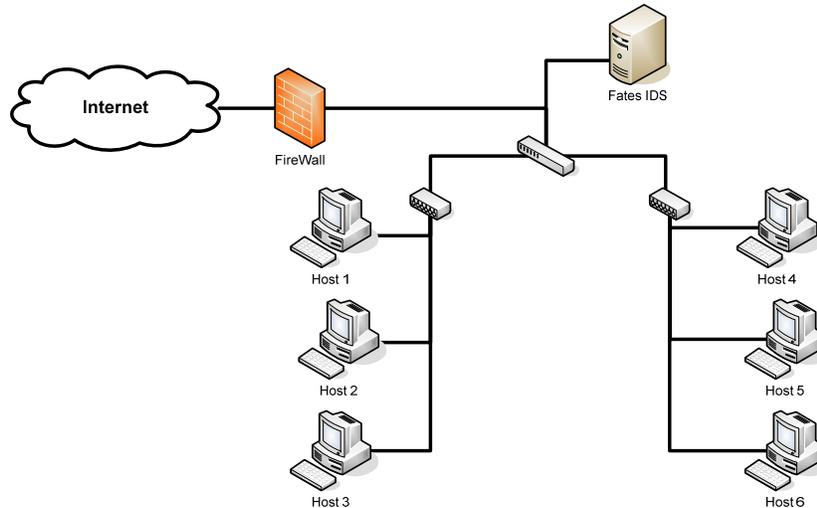
Sniffer

- Offline detection
 - Parsing TCPdump files of previously recorded traffic
- Real-time detection
 - Promiscuous capturing of packets as they come into/out of the network

Computer Science and Engineering @ University of South Carolina



Sniffer



Computer Science and Engineering @ University of South Carolina



Packet Processing

- The time of operation is divided into time steps (predefined by the user)
- Static windows are used to cut down on processing time
- All data used in analysis has a time-to-live measured in windows
 - Alleviates skewing of results

Computer Science and Engineering @ University of South Carolina



Packet Processing

- Maintains a list of internal IP addresses
- Two processing components
 - External Scan Detection Component
 - Detects scans from the outside world
 - Internal Host Monitor Component
 - Examines the state of monitored hosts' activities

Computer Science and Engineering @ University of South Carolina



Packet Processing

- External Scan Detection Component
 - Approximation cache of miss behavior
 - Provides a best approximation of potential scans with finite space requirements
 - If neither the source or destination is a monitored host, the packet could be part of a scan

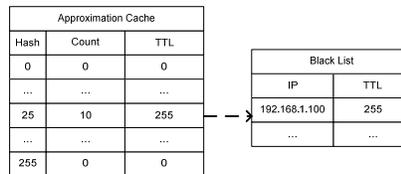
Computer Science and Engineering @ University of South Carolina



Packet Processing

- Hash of the Source address is the index into an approximation cache
- TTL is set at each time step and whenever entry is accessed
- If count exceeds a threshold, the source is listed as a potential scanner

MAX_COUNT_TTL: 255
MAX_MISS_COUNT: 10
MAX_BLACKLIST_TTL: 255



Computer Science and Engineering @ University of South Carolina



Packet Processing

- Internal Host Monitor Component
 - Monitors subnet by IP or range of IP (stored in binary search tree)
 - A hash table of hosts
 - Current threshold
 - Current charge
 - Produces cumulative charges to be compared to individual thresholds

Computer Science and Engineering @ University of South Carolina



Packet Processing

- Each host is charged for each packet it sends
- Charge is a result of packet type
 - Connectionless
 - Connection-oriented

Packet Type	Formula
TCP	Charge = $2 * (state - 1)$
UDP	Charge = $2 * (count - 1)$

Computer Science and Engineering @ University of South Carolina



Packet Processing

- TCP state
 - Incoming packets decrease state by one
 - Outgoing packets increase state by one

	Type	Modifier
Incoming	SYN ACK FIN SYNACK FINACK	+1
Outgoing	SYN ACK FIN SYNACK FINACK RST	-1

Computer Science and Engineering @ University of South Carolina



Packet Processing

- UDP count
 - Number of packets with duplicate payload
 - Count of packet is stored in an approximation cache
 - Payload is hashed to index
 - Entries associated with a TTL



Packet Processing

- At end of time step
 - States used in TCP/IP are adjusted
 - If greater than zero, decremented by one
 - If less than zero, increased by one
 - TTL of elements in UDP's approximation cache is decremented by one
 - If TTL is zero, count is set to zero



Packet Processing

- At end of time step (continued)
 - All charges to hosts are added up
 - The total is compared to the host's initial threshold
 - Initial threshold is user defined for each host
 - If threshold is exceeded, threshold is set equal to the total

Computer Science and Engineering @ University of South Carolina



Packet Processing

- Threshold decay
 - If in subsequent time steps the average is less than the initial threshold, it is decayed
 - Average of time step charges
 - $avg = avg_{prev} * (1-\alpha) + TotalCharge * (\alpha)$

Computer Science and Engineering @ University of South Carolina



Packet Processing

- Threshold decay rate
 - $T_{current} = T_{current} - 1/2(T_{initial} - avg)$
 - Quality:
 - Slowly redemptive
 - Decay rate is directly correlated to the history of a monitored host

Computer Science and Engineering @ University of South Carolina



Alarming

- In a well-behaved network the thresholds reach equilibrium
- In presence of scanning the threshold continually grows (only plateaus at saturation)
- This behavior is obvious upon observation (dependent on human interpretation)

Computer Science and Engineering @ University of South Carolina



Testing

- Experimental Data
 - Slammer (simulation)
 - Very effective worm
 - Blatantly obvious scanning behavior
 - Nmap (observed network traffic)
 - Standard issue scanning tool
 - Used to test TCP/IP detection capabilities

Computer Science and Engineering @ University of South Carolina



Testing

- Experimental Data (continued)
 - World of Warcraft (observed network traffic)
 - Sporadic packet transmission
 - Taxed servers with need for retransmission
 - Peer-to-Peer (observed network traffic)
 - Uses scanning to establish overlay network
 - Allows for file transfer

Computer Science and Engineering @ University of South Carolina



Slammer

- High-speed worm
- Propagates through UDP packets
- Provides a good lower-bound

Computer Science and Engineering @ University of South Carolina



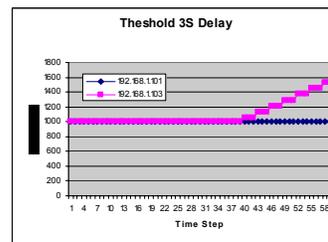
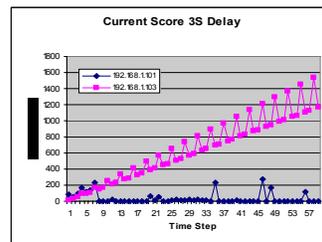
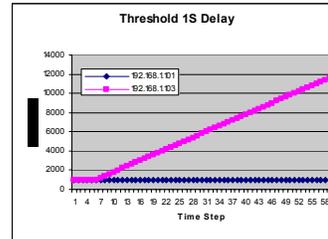
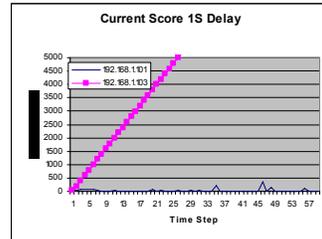
Slammer

- Simulation
 - Advantages:
 - No legal issues
 - Specifics of the traffic are already known
 - Adjustable
 - Optional parameters:
 - Rate of Infection
 - Time of propagation
 - Size of network
 - Delay before inception of infection

Computer Science and Engineering @ University of South Carolina



Slammer



Computer Science and Engineering @ University of South Carolina



Nmap

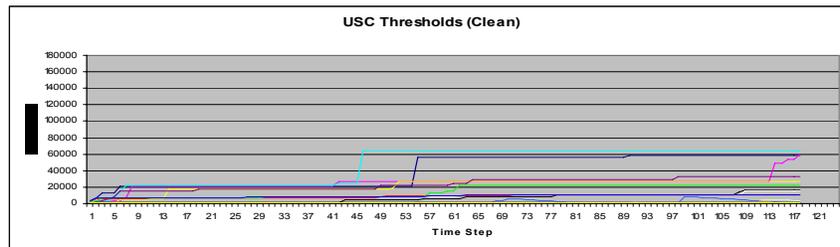
- The network
 - Subset of the University of South Carolina's network
 - Monitoring 8 /24 subnets
 - Running Snort for comparison
- The scans
 - Half-Open scan
 - Also known as SYN scan
 - ACK scan
 - Distinct scan type
 - FIN scan
 - Stealth
 - RST scan
 - Stealth

Computer Science and Engineering @ University of South Carolina



Nmap

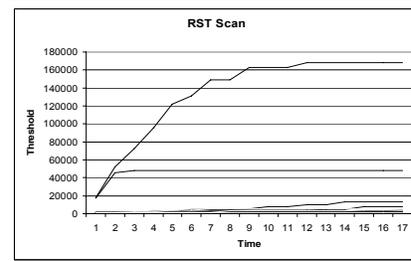
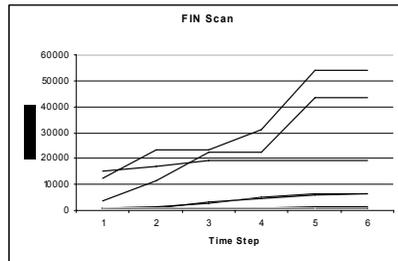
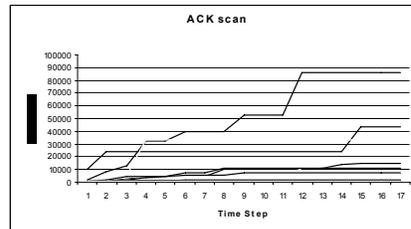
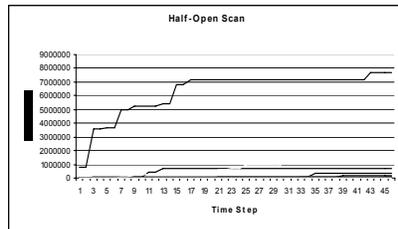
- Clean USC traffic
 - Thresholds tend to “jump” and “plateau”
 - The network reaches equilibrium



Computer Science and Engineering @ University of South Carolina



Nmap Thresholds



Computer Science and Engineering @ University of South Carolina



World of Warcraft

- Massively Multiplayer Online Role-Playing Game (MMORPG)
 - 1.5 million users
 - Several servers
 - Divided into regions
 - Possibility of lag due to congestion at servers

Computer Science and Engineering @ University of South Carolina



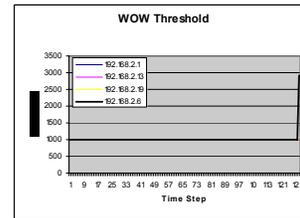
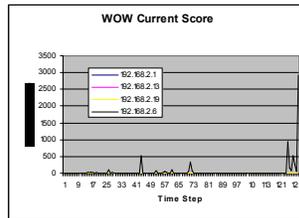
World of Warcraft

- TCPdump of 4 hosts on a home network
 - All ran HTTP traffic
 - One ran a WOW client
- Recorded 20 minutes of network traffic
 - Including: video streaming, HTTP, and WOW traffic

Computer Science and Engineering @ University of South Carolina



World of Warcraft



- The spikes are from transfer between servers
- Even in the presence of large lag, no extreme jumps in charges

Computer Science and Engineering @ University of South Carolina



Peer-to-Peer Networks

- Clients use scanning to find other peers, or contact a central servers
- Clients maintain a list of servers, but the server list changes
- Resembles scanning in finite space

Computer Science and Engineering @ University of South Carolina



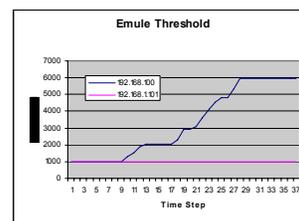
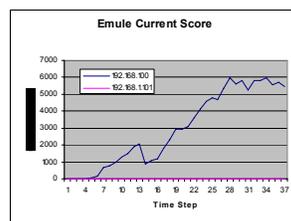
Peer-to-Peer Networks

- Test data
 - TCPdump of Emule traffic from a home network
 - 1 host (no network activity)
 - 1 host running Emule client
 - Contacting servers
 - Transfer files

Computer Science and Engineering @ University of South Carolina



Peer-to-Peer Networks



- Though benign the attempt to connect to the servers resembles scanning
- As a result, the threshold looks similar to scans seen in the USC dataset

Computer Science and Engineering @ University of South Carolina



Evaluation

- Advantages
 - The simple calculations are still effective in detecting scans
 - Individual assessment of hosts aids in correcting the anomaly
 - Dynamic thresholds provide better understanding of diverse network entities

Computer Science and Engineering @ University of South Carolina



Evaluation

- Disadvantages
 - Does not distinguish between benign and malicious scanning
 - Intent is not our focus
 - Scalability
 - The less the granularity, the less the precision
 - Assumes source addresses are not spoofed
 - Many other such systems are also victim to this

Computer Science and Engineering @ University of South Carolina



Areas of Improvement

- Integrate a GUI interface
 - Alternately, integrate into other systems
- Integrate a rate of change analytical tool set
 - Providing automated alarming

Computer Science and Engineering @ University of South Carolina



Summary

- Fates provides a granular approach that allows for useful notification of anomalous activities
- Alarming is as specific as the user wishes
- Detection is feasible in a real-time network deployment without complex mathematical models

Computer Science and Engineering @ University of South Carolina



Conclusion

- Present two complementary views on intrusion detection
- Develop and implement two intrusion detection approaches based on the two views
- Experimental results show the effectiveness of the two approaches
- Investigate the feasibility of integration

Detection of Undesirable Insider Behavior

Joseph A. Calandrino^{1*}, Steven J. McKinney^{2*}, and Frederick T. Sheldon³

¹ Princeton University, Princeton, NJ 08544, USA
jcalandr@princeton.edu

² North Carolina State University, Raleigh, NC 27695, USA
sjmckinn@ncsu.edu

³ Oak Ridge National Laboratory, Oak Ridge, TN 37830, USA
sheldonft@ornl.gov

Insiders, those within or closely related to an organization, pose the greatest risk to an organization's information systems. Organizations grant insiders both authorized access to and knowledge of their information systems, primarily computer systems and the organization's network. In the past, insiders have abused this trust by stealing or corrupting data, committing fraud, and modifying performance reports [10]. Because these insiders may act within the bounds of their privileges, mitigation of the insider threat differs from that of external threats.

Undesirable insider behavior involves any willful or negligent misuse of resources in an organization's information systems. Numerous existing systems, such as firewalls and intrusion detection systems (IDSs), seek to mitigate the threat that parties external to an organization pose to its information systems. Unfortunately, these mechanisms often do not restrict or monitor insiders [1]. Little research beyond access control strives to mitigate the threat that malicious or apathetic insiders may introduce. These insiders present a particularly insidious problem as they may behave adversely without violating access control policies. A 2004 survey of security and law enforcement executives found that, among respondents that experienced e-crime or intrusions and could classify attacks as internal or external, an average of 29% of attacks against their organizations came from insiders [11]. This fact, combined with the established difficulty of the insider problem [7] and the ability of a single malicious insider to cause significant financial impact (\$500 million in one case [1]), demonstrates the need for further research on insider threat detection systems.

This document describes the Intelligent Insider Threat Detection (I²TD) system for monitoring and evaluating insider behavior to detect potentially malicious or otherwise undesirable activity. The system observes an insider's local system and network-based activities and is extensible to other aspects of the information system. A rule-based method immediately notifies administrators of activity known to be suspicious, and data mining tools regularly inspect compiled user behavior for anomalies that could indicate undesirable activity. Although the monitoring, rule-based, and data mining components of the system are all of critical importance, this document focuses on the data mining component.

Numerous systems have attempted to apply machine learning and statistical learning techniques to detect intrusions and other forms of fraud. Anderson et

* This research was performed during an internship at Oak Ridge National Laboratory.

al. used a statistical approach to evaluate the deviation of present user behavior from past behavior [2], and Chan et al. applied data mining techniques to the similar problem of detecting credit card fraud [4]. [8, 9] describe work towards real-time intrusion detection using data mining. The underlying systems use learning agents to regularly mine data and extract patterns that serve as classifiers for real-time detection of intrusions. [8, 9] are complementary to this paper: additional mining algorithms could assist in refining the I²TD system's set of real-time rules. This work attempts to derive additional utility beyond [8, 9] by considering characteristics of aggregated data.

The Minnesota Intrusion Detection System (MINDS) detects anomalous network connections based on characteristics of other connections over the same time period [5, 6]. MINDS does not detect anomalies in real time, but similar to [8, 9], it also mines for association rules to assist in real-time detection. Because MINDS is not real time, it may derive and analyze characterizes of aggregate data that allow more accurate evaluation. The data mining component of this work utilizes an approach similar to MINDS to assess user behavior.

The data mining component of the I²TD system regularly retrieves recent user activity from a database and analyzes it for anomalies. Rather than mining the raw data, the system computes aggregate characteristics of user activity during the period with the goal of smoothing inconsequential deviations and producing more accurate results. Following aggregation, the system compares the characteristics to those of the user during past periods to compute an anomaly score for the most recent period. The system also offers additional information regarding the impact of various behavioral characteristics on the anomaly score. An analyst may use the anomaly score and additional derived data to determine whether a user's behavior over a given period warrants further exploration.

At this time, the data mining component considers seven derived characteristics: number of logins, number of host machines used, number of file opens caused, number of file deletes caused, number of unique files accessed (opened or deleted), number of unique files opened, and number of unique files deleted. The system presently operates under the assumption that deviations from typical login and file access patterns are an effective indicator of undesirable insider behavior. Various other behavior attributes may also be useful, such as network or database accesses. Because the system presently aggregates data on a daily basis and considers only a user's personal historical data, however, the sample space may be small. Given a small sample space, high dimensionality may scatter the data points too greatly to draw meaningful inferences. Thus, naive consideration of additional characteristics may have an adverse impact on analysis.

Like [5, 6], the data mining component of this system isolates anomalies using the local outlier factor (LOF) metric, developed by Breunig et al. [3]. Given an item in a data set, algorithms for LOF map the set to vector space and establish a group of n nearest neighbors for the item. The algorithms then determine how much more or less isolated the item is than its nearest neighbors and use the ratio to produce the item's LOF score. If the item is part of an evenly dispersed cluster, the LOF for the item will be close to one. An LOF far greater than

one may occur if the item's nearest neighbors are relatively far away yet those neighbors are members of tight clusters.

To determine whether a user's behavior on a given day is anomalous, the system examines the day's activity record in the context of previous days' records and computes an LOF score for the day. Prior to this step, however, the system computes the standard deviation of each characteristic in previous days' records and uses the results to normalize all characteristics of the records. Because the LOF algorithms look at the mapping of records to vector space, this normalization step prevents characteristics with greater deviation from having a disproportionate impact on the Euclidean distance between records.

The rationale behind the use of LOF as opposed to a single cluster or multiple clusters is that user tasks may change from day to day and some tasks may result in greater behavior variation than others. If certain task combinations arise with reasonable regularity, a stable set of clusters will emerge. During a user's first few days in the system, the data may be too sparse and widely dispersed for the data mining component to draw meaningful conclusions. Thus, management must closely monitor users' initial activity until simple patterns begin to emerge. As patterns emerge, automated analysis increases in effectiveness.

In addition to the local outlier factor metric, the data mining component determines each characteristic's relative contribution to an item's LOF and offers the results to administrators. This additional impact information not only makes analysis far quicker and easier for administrators but also raises the possibility that supervisors may be able to assist in threat detection. In general, supervisors are most familiar with a user's day-to-day tasks, and therefore, most qualified to assess the actions necessary to complete those tasks. Unfortunately, not all supervisors are equally technically literate. Additional information beyond a cryptic local outlier score, if presented properly, is a critical first step towards making results accessible to non-technical users.

Preliminary results indicate that the data mining component effectively detects anomalous behavior in synthetic data sets. Also, the additional characteristic impact information is meaningful and helpful in those tests. System administrators have the ability to use LOF scores in whatever manner they choose: an administrator may look at all users with LOF scores above a given threshold or may consider only users with the top k anomaly scores. Should an event occur that an administrator knows will influence the anomaly scores of certain users in a given manner, the administrator can filter those scores to remove false positives. Administrators may even create more complex rules, such as ignoring certain LOF scores heavily influenced by attributes that the administrator considers insignificant. The strength and adaptability of this data mining approach indicate that it may be a useful tool for uncovering novel insider threats.

The prototype implementation of the system is presently only partially complete. Extensions are necessary for the monitoring tools, and the analysis components are not optimized for wide-scale deployment. Continued testing of the analysis components of the system as new insider activity data becomes available would be useful. Comprehensive analysis of system- and network-level charac-

teristics of user behavior also may yield insight regarding the utility of these characteristics in assessing insider threat. For example, certain aspects of user behavior be too noisy to serve as reliable indicators while others may be extremely consistent over time. Finally, the system would benefit from the addition of components to mine continuously for heuristic rules to supplement the system's real-time rule-based analysis component.

Acknowledgments. Calandrino and McKinney performed this research while under appointment to the Department of Homeland Security (DHS) Scholarship and Fellowship Program, administered by the Oak Ridge Institute for Science and Education (ORISE) through an interagency agreement between the U.S. Department of Energy (DOE) and DHS. ORISE is managed by Oak Ridge Associated Universities (ORAU) under DOE contract number DE-AC05-06OR23100. All opinions expressed in this paper are the authors' and do not necessarily reflect the policies and views of DHS, DOE, or ORAU/ORISE.

References

1. D. Anderson, D. M. Cappelli, J. J. Gonzalez, M. Mojtahedzadeh, A. P. Moore, E. Rich, J. M. Sarriegui, T. J. Shimeall, J. M. Stanton, E. A. Weaver, A. Zagonel. Preliminary system dynamics map of the insider cyber-threat problem. Proceedings of the 22nd International Conference of the System Dynamics Society. 2004.
2. D. Anderson, T. F. Lunt, H. Javitz, A. Tamaru, A. Valdes. Detecting unusual program behavior using the statistical component of the next-generation intrusion detection expert system (NIDES). Technical Report SRI-CSL-95-06. SRI International, May 1995.
3. M. M. Breunig, H.-P. Kriegel, R. T. Ng, J. Sander. LOF: Identifying density-based local outliers. Proceedings of the ACM SIGMOD 2000 International Conference on Management of Data. ACM, 2000.
4. P. K. Chan, W. Fan, A. L. Prodromidis, S. J. Stolfo. Distributed data mining in credit card fraud detection. IEEE Intelligent Systems. IEEE, 1999.
5. V. Chandola, E. Eilertson, L. Ertöz, G. Simon, V. Kumar. Data mining for cyber security. Data Warehousing and Data Mining Techniques for Computer Security. Springer, 2006.
6. L. Ertöz, E. Eilertson, A. Lazarevic, P.-N. Tan, V. Kumar, J. Srivastava, P. Dokas. MINDS - Minnesota intrusion detection system. Next Generation Data Mining. MIT Press, 2004.
7. INFOSEC Research Council. Hard problem list. November 2005.
8. W. Lee, S. J. Stolfo. Data mining approaches for intrusion detection. Proceedings of the Seventh USENIX Security Symposium. January 1998.
9. W. Lee, S. J. Stolfo, P. K. Chan, E. Eskin, W. Fan, M. Miller, S. Hershkop, J. Zhang. Real time data mining-based intrusion detection. Proceedings of the 2001 DARPA Information Survivability Conference and Exposition II. June 2001.
10. M. R. Randazzo, M. Keeney, E. Kowalski, D. Cappelli, A. Moore. Insider threat study: Illicit cyber activity in the banking and finance sector. Technical Report CMU/SEI-2004-TR-021. Carnegie Mellon Software Engineering Institute, June 2005.
11. United States Secret Service, CSO, CERT. 2004 e-crime watch survey. 2004.

Detecting Undesirable Insider Behavior

Joseph A. Calandrino*
Princeton University

Steven J. McKinney*
North Carolina State University

Frederick T. Sheldon
Oak Ridge National Laboratory

May 15, 2007

*This research was performed during an internship at ORNL

OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY

An Example

"[A bank] employee gained control over data after performing several preparatory actions, such as eliminating some monitoring ... or convincing ... personnel to take deliberately corrupted data from his ... computer instead [of] from the official Reuters terminal."

-From Anderson et al., 2004

- Enters false data
- Receives promotions, bonuses, control
- Financial impact: **\$500 million**

Undesirable Insider Behavior

- **Results in ~29% of attacks against organizations**
(US Secret Service et al., 2004)
- **Can devastate an organization**
- **Fundamentally differs previously addressed threats**
- **Comes from trusted individuals**
- **Is a fuzzy threat**
- **Is tedious to identify**

OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



Roadmap



- **Existing work**
- **Our contributions**
- **Mining approach**
- **Evaluation**
- **Discussion**
- **Conclusion and Future Work**

OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



Existing Work

- Insider threat characterization
- Intrusion detection
- Machine learning / data mining
 - Statistical deviation (Anderson et al., 1995)
 - Real-time IDS (Lee et al., 1998; Lee et al., 2001)
 - MINDS (Ertöz et al., 2004; Chandola et al., 2006)

OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



Our Contributions

- Designed a system to:
 - Monitor insider system and network activity
 - Perform rule-based (or other static) analysis
 - **Mine compiled behavior for anomalies**
- Implemented the system

OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



Data Mining Component Role

- **Periodically extracts aggregate data**
- **Analyzes data to isolate points of interest**
- **Identifies novel threats**
- **Generates new rules (future work)**

OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



Characteristic Derivation

- **Daily data analysis**
- **Per-user data**
- **System-level events alone... for now**
- **Seven characteristics (file accesses, hosts, logins)**
- **Normalization using historical SD**

OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



Data Analysis

- **Similar to MINDS (implementation differs):**
 - Map data to vector space
 - Compute local outlier factor (Breunig et al., 2000)
 - “Neighborhood” outlier metric
 - Euclidean distance – not mandatory
 - Derive additional hints for administrators

OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



Local Outlier Factor

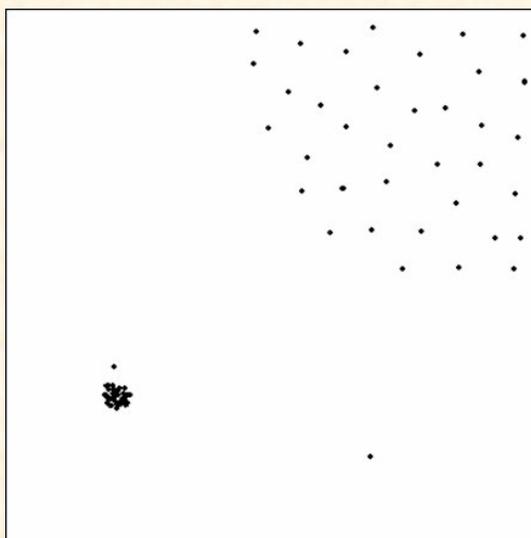


Image Reproduced from Breunig et al. (2000)

OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



Local Outlier Factor

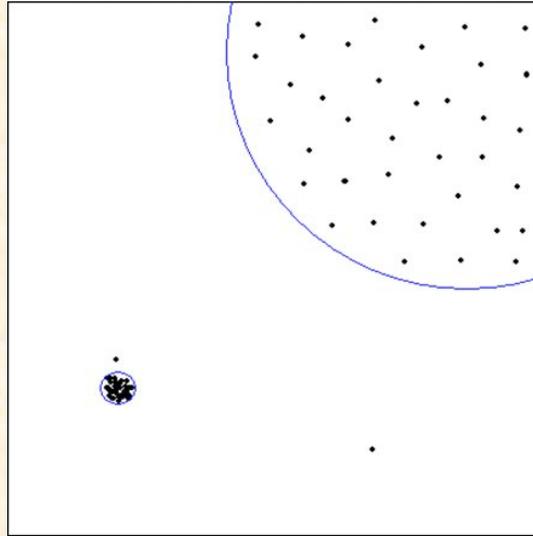


Image Reproduced from Breunig et al. (2000)

OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



LOF Scores

- **Could report to administrator**
- **Supervisors may be preferable**
 - Supervisors are most familiar with day-to-day tasks
 - Supervisors may be less technically literate
- **Consider impact of characteristics on outlier factor**



OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



Evaluation

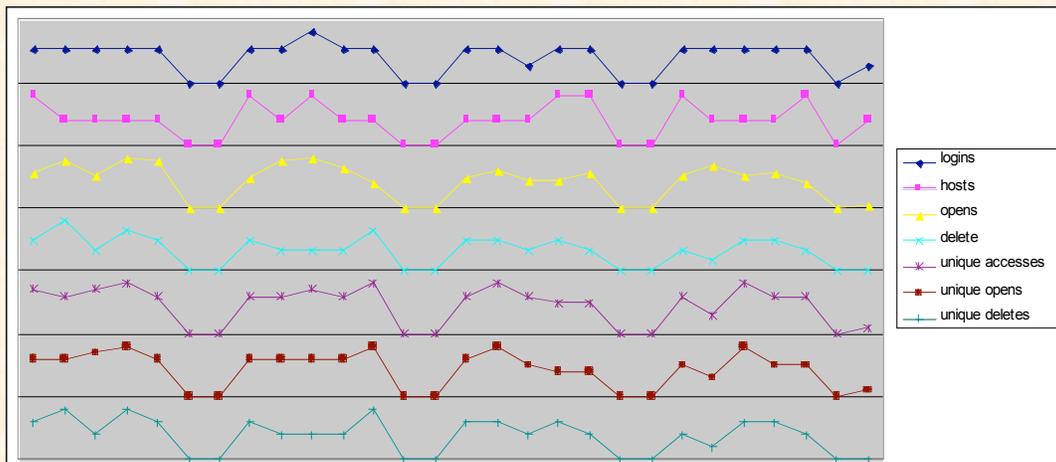
- **Generated 28 Days of artificial data**
 - Presume patterns
 - No activity on weekends until final weekend day
 - Activity comes from distribution on weekdays
 - Is real behavior like this?

OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



Test Data

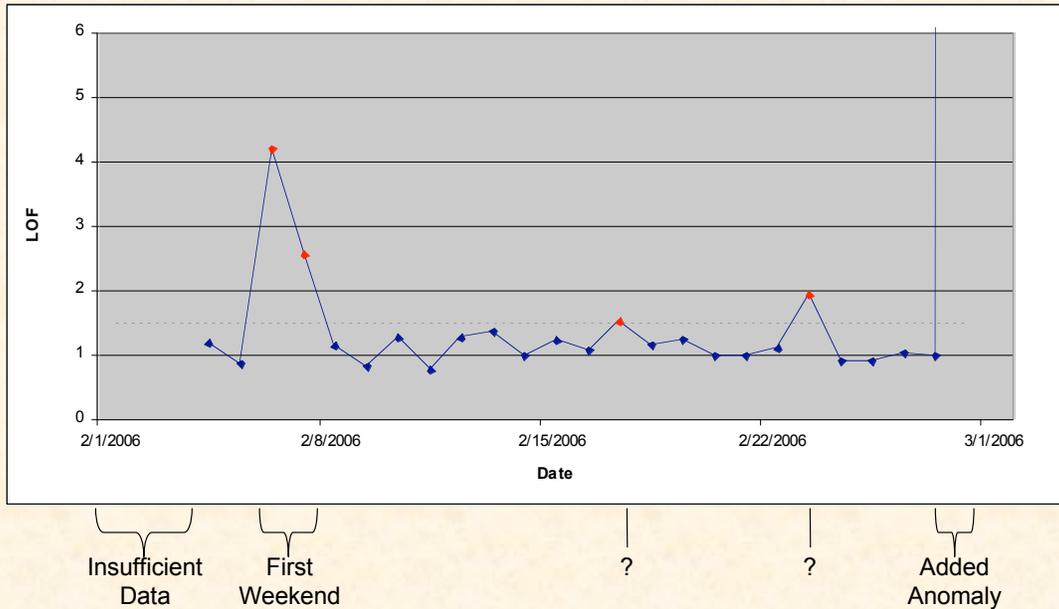
- **Graphically (scaled):**



OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



Results

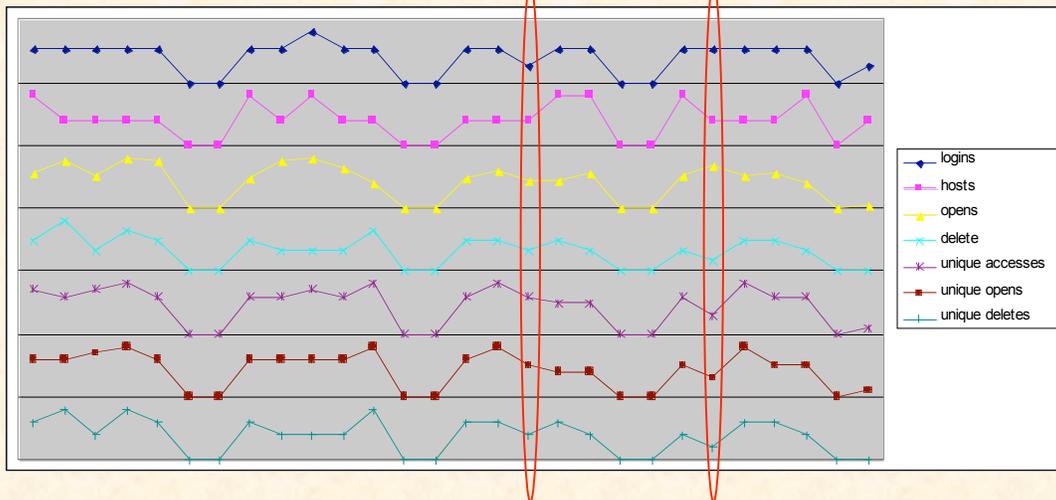


OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



Results

• What's wrong?

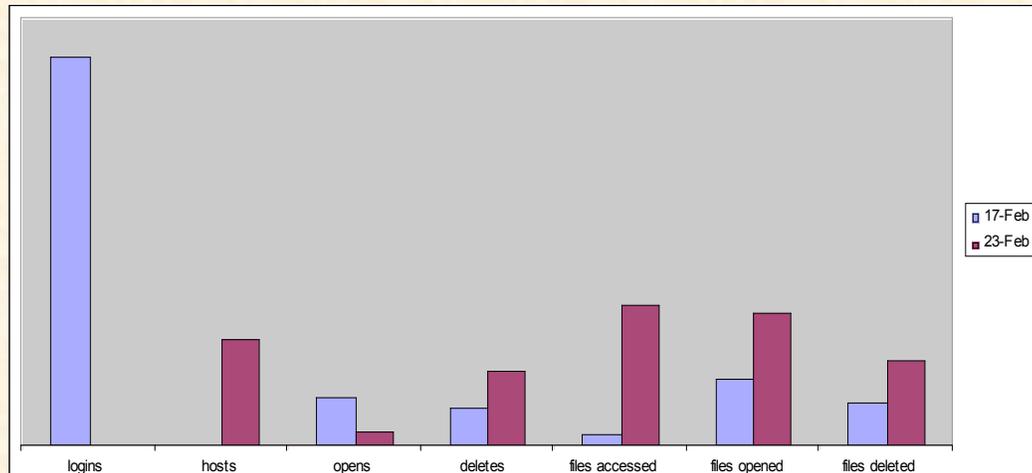


OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



Results

- **Why guess? Look at hints for 2/17, 2/23...**



OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



Discussion

- **Caught added anomaly**
- **What about the four others?**
 - All were “anomalous”
 - Hints allowed rapid analysis
 - Depend on parameters, administrator focus

OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



Conclusions

- Insider threat – important problem
- Data mining – helpful technique
- New tool – promising results
- TODO list – long

OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



Future Work

- Additional aspects and characteristics
- Issues: drift, deja vu
- Better test data
- Rule extraction

OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



Thank You

- **Questions?**

Calandrino and McKinney performed this research while under appointment to the Department of Homeland Security (DHS) Scholarship and Fellowship Program, administered by the Oak Ridge Institute for Science and Education (ORISE) through an interagency agreement between the U.S. Department of Energy (DOE) and DHS. ORISE is managed by Oak Ridge Associated Universities (ORAU) under DOE contract number DE-AC05-06OR23100. All opinions expressed in this paper are the authors' and do not necessarily reflect the policies and views of DHS, DOE, or ORAU/ORISE.

OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



Extended Abstract

Quantifying the vulnerability of tactical data networks

Andy Loebel, James Nutaro, and Teja Kuruganti
Oak Ridge National Laboratory
{loebblas,nutarojj,kurugantipv}@ornl.gov
Rajanikanth Jammalamadaka
University of Arizona
rajani@ece.arizona.edu
September 22, 2006

Overview

Experience with the first five generations of modern military data networks has shown a need to address physical and operational security of computer based functions. The sixth generation of large scale, complex systems software represents a major change in how computing and communications will support military operations. This new paradigm requires that network security and information assurance be addressed in the earliest phases of the system life cycle. An affordable and deployable system must begin with an analysis of security requirements in the context of anticipated operational use. How the network is used and operated is likely to have a significant impact on its intrinsic security. This paper takes a first step towards establishing requirements oriented models to assess security needs in the context of overall system performance objectives. This paper presents an abstract and, in this first issuance, a simplified view of network security and information assurance as it pertains to the evolving vision of network centric warfare. When system requirements can be quantified, it is possible to make informed technical decisions concerning development options and expected functionality. If important functional requirements can be broadly understood and quantified, then accurate and precise answers concerning a design's adequacy can be given early in the design lifetime. This, in turn, will significantly reduce the overall cost of the system.

Introduction

Cost reduction and expected reliable performance are two goals of model based system design. This is especially true for the design of weapon systems and command and control systems, which are particularly difficult to test in situ. The coming generation of complex systems, and the Future Combat System in particular, will be tested

piecemeal prior to their combat debuts. This makes model based studies the only way to predict the overall performance of these systems prior to their critical use. Informative model based studies must be built on quantifiable definitions of performance metrics. The ability to share data is a critical feature of the Future Combat System. In the absence of heavy armor, superior situational awareness and understanding are essential for an engaged unit to survive and complete its mission. Because the Future Combat System is a highly mobile force, data sharing will be enabled by an ad hoc, wireless data network. This network will need to be robust, reliable, and carry a great quantity of information. It will, therefore, be expensive.

Stringent performance requirements and the anticipated high cost of the Future Combat System data network strongly motivates careful identification of its functional requirements. The most essential requirements must be quantifiable to allow for early, model based studies of proposed designs. Network security is one of these critical requirements, and one that should be subjected to early, detailed, and methodical scrutiny. This paper recommends a research effort to develop a methodology for analyzing network security and information assurance requirements for network centric combat systems. A security focused requirements analysis methodology is essential if affordable and deployable network centric systems are going to be produced.

The remainder of this paper presents a preliminary model that serves to illustrate how security related requirements analysis might be conducted. However, only a few of the many security and performance attributes¹ of modern systems (or system of systems) are empirically understood. A comprehensive analysis methodology is beyond the scope of this paper; a complete methodology will require an intensive and focused research effort.

The benefit of pursuing such a research agenda will be a quantifiable understanding of security and assurance requirements as they relate to modern tactical data networks. If successful, there will be two long term benefits. First, a substantial reduction in the lifetime cost of these systems will be possible due to a reduction in experimental development, improved system of systems testing and validation, and reduced maintenance and configuration management costs. Second, the operational performance of network centric systems (or system of systems) will be improved. These benefits will be the result of a precise understanding and articulation of system performance requirements.

A Systems Development and Implementation Study for 21st Century Software and Security

Third Cyber Security and Information Infrastructure Research Workshop May 2007

Andrew Loebl, James Nutaro, and Teja Kuruganti
Oak Ridge National Laboratory
{loebblas,nutarobj,kurugantipv}@ornl.gov

Rajanikanth Jammalamadaka
University of Arizona
rajani@ece.arizona.edu

OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



This work was inspired by the lead author's experience with projects implementing the *Revolution in Military Affairs*. This concept development was not funded by any combination of work related to any of these projects.

OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



Requirements analysis for net-centric systems

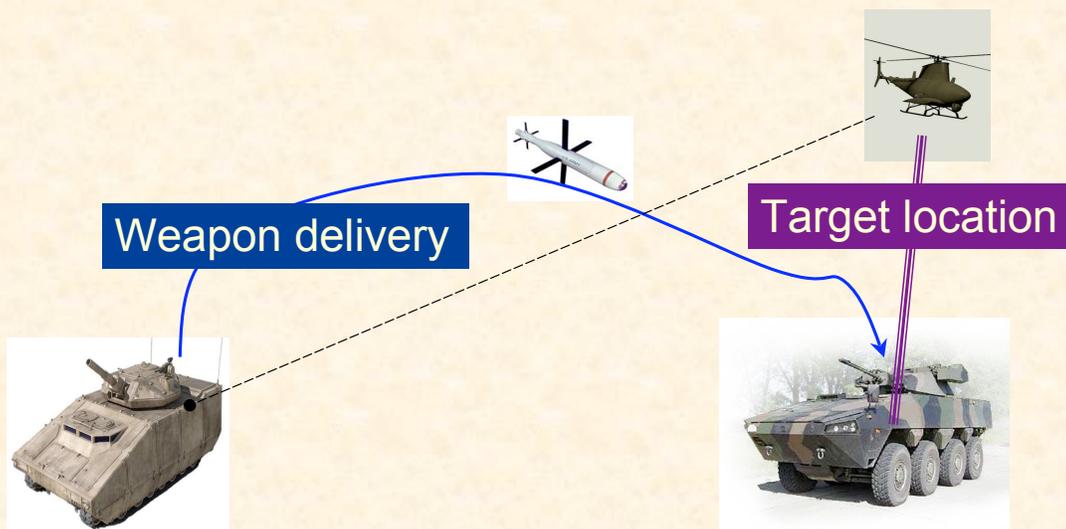
- What do I need?
- Can we build it?
- How much will it cost?
- What are my alternatives?



OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



Precision mortar munitions



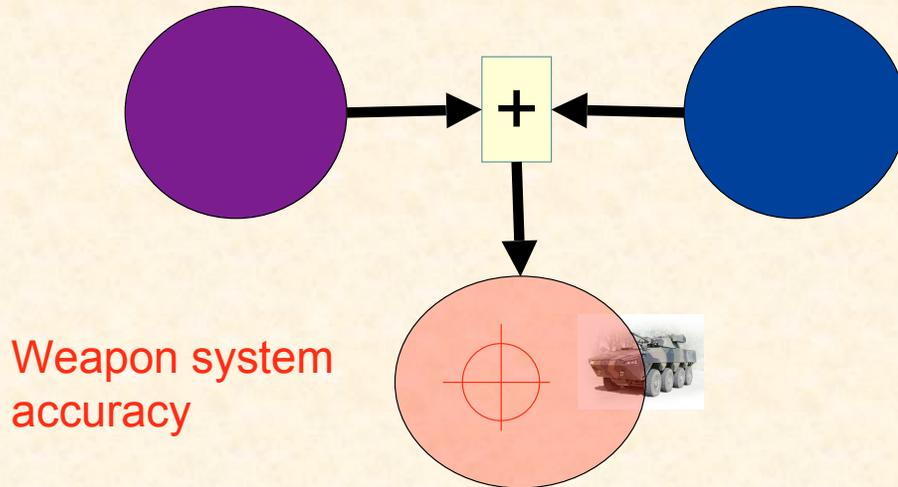
OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



Precision mortar munitions: Metrics

Target location error

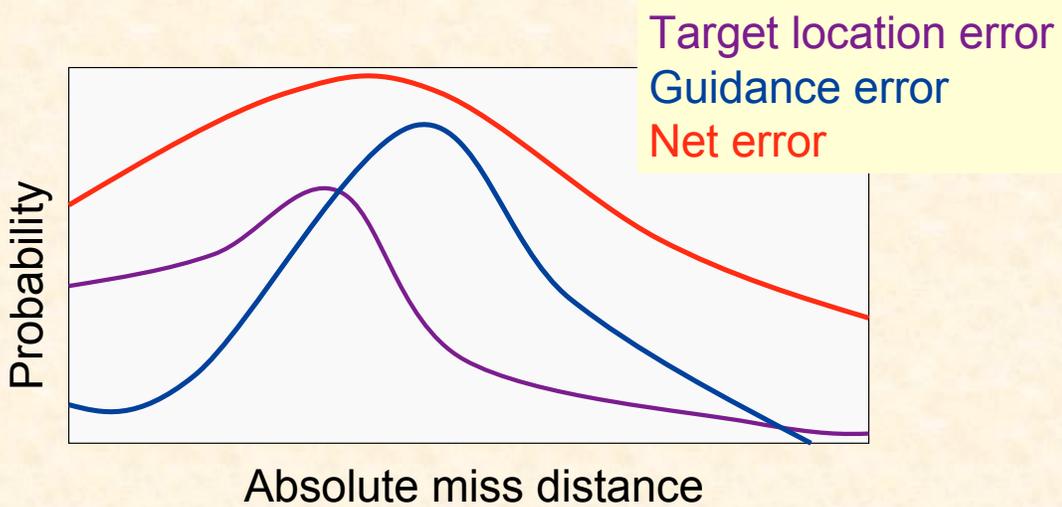
Guidance error



OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



Quantifying performance



OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



Main features of this model

- Quantified key performance characteristics
- Anticipates overall system performance
- Allows us to ask
 - Is it good enough?
 - Where should I spend my money?
 - What can I expect from the system when built?
- **These question must be asked early in the system lifecycle!!!!!!**
- Ask later and you only find out what you got for your money - big gamble!

OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



Design of networked systems

- How much security do I need?
- What will it cost?
- How will the system perform?
- What are my alternatives?
- **If you can't answer these questions early, then you can not design network based systems**
 - Expensive to build
 - Even more expensive to maintain and operate
 - Need constant supervision to ensure consistent performance

OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



Modern Systems Attributes include: Networked, Embedded

- Embedded systems are rapidly replacing desktop systems for critical applications
- Embedded systems are becoming more powerful and more flexible
- Embedded systems are often invisible systems
 - No owner
 - No administrator
 - No upgrades or patches
 - Ubiquitous
 - Are they secure enough?
 - How secure should they be?



OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



Networked systems are becoming pervasive for the armed forces

- Global Information Grid
- Future Combat System
- Unmanned vehicles and sensors



OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



Mix of embedded and “standard” applications

- **World-spanning military network tied to regional tactical networks**
 - Embedded, networked systems in sensors, vehicles, weapons
 - Connect to critical command and control applications through the network
- **Secure operations = kitchen sink?**
- **How much security is enough?**
- **How much will it cost to build, operate, and maintain?**

OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



Implications of networked, embedded systems

- Vulnerable systems will be difficult to locate, and impossible to “fix”
- Attackers can use relatively insecure embedded systems to silently access and move through the network
- Our current approach to network security does not handle this new paradigm well

OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



Characteristics for a Preliminary Model for Information Assurance and Systems Security

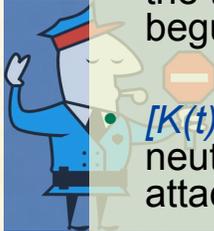
- Security related requirements analysis
- Security and performance requirements understood early in the system lifecycle
- Marginal additional assurance of measures considered better understood
- Cost consequences in two dimensions better understood

Notional Vulnerability Model Offered for Illustration

- **Illustrate specification needed to inform requirements determination and design decisions**
- **Expected operational life time of a threat**
- **Appraise performance metrics for essential security processes**
- **Describes expected operational life-time of a threat in terms of**
 - **Identification**
 - **Elimination**

A Probabilistic Model Formulation

- $D(t_d)$ - probability of identifying the attack within the time interval $[0, t_d]$ after the attack has begun
- $[K(t)|D(t_d)]$ is the conditional probability of neutralizing the attack at time t given that the attack was identified at time t_d .
- $K(t)$ is probability of neutralizing the attack after a time t
- $K(t) = [K(t)|D(t_d)]D(t_d)$ is the expected lifetime of an attack.



Notional (cont'd)

- $k(t)$ is the probability density function of $K(t)$

$$E[K] = \int_0^{\infty} tk(t) dt$$

This is the quantified vulnerability

Example 1; GPS Jamming

- $D(t_d)$ denotes the time, in minutes, to detect jammer
 - triangular distribution with $[0, 60]$ minutes as end points and a mode of 30
- $[K(t)|D(t_d)]$ is a random variable denoting the time in minutes to kill jammer following detection
 - $[0, 10]$ minutes as end points and a mode of 5 minutes

$$D(t_d) = \begin{cases} 0 & \text{if } t < 0 \\ t^2/1800 & \text{if } 0 \leq t \leq 30 \\ \frac{t^2 - 120t + 1800}{-1800} & \text{if } 30 < t \leq 60 \\ 1 & \text{if } 60 < t \end{cases}$$

$$[K(t)|D(t_d)] = \begin{cases} 0 & \text{if } t < 0 \\ t^2/50 & \text{if } 0 \leq t \leq 5 \\ \frac{t^2 - 20t + 50}{-50} & \text{if } 5 < t \leq 10 \\ 1 & \text{if } 10 < t \end{cases}$$

OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



Example 1; GPS Jamming (con't)

- $K(t)$ time needed to kill the jammer is

$$K(t) = [K(t)|D(t_d)]D(t_d) = \begin{cases} \frac{t^4}{90000} & \text{if } 0 \leq t \leq 5 \\ \frac{t^2}{1800} \frac{t^2 - 20t + 50}{-50} & \text{if } 5 < t \leq 10 \\ \frac{t^2}{1800} & \text{if } 10 < t \leq 30 \\ \frac{t^2 - 120t + 1800}{-1800} & \text{if } 30 < t \leq 60 \\ 1 & \text{if } 60 < t \end{cases}$$
- expected attack lifetime is $\int_0^{\infty} t \frac{d}{dt} K(t) dt = 30$ minutes

OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



Can now ask...

- Can my system continue to operate with a 30 minute loss of the GPS sub-systems?
- If not, should I improve
 - Detection?
 - Elimination?
- Which is more cost effective?
- How much improvement is needed?

Early Understanding Is Valuable

- Fundamental relationships between system performance parameters can be studied early
- Quantified requirements can be validated by simulation
 - Force on force simulations using assumed performance parameters
 - Adequacy of a requirement to detect within a critical time period can be evaluated in (simulated) operations
- Modeling allows a host of assumptions and scenarios can be evaluated **before the system is built!**

Conclusions

- A research program is needed to develop a quantitative, model based, requirements analysis methodology for network security and information assurance
- This is needed to
 - Produce testable requirements that contribute to security and assurance
 - Produce requirements that can be validated before the system is built
 - Clearly communicate design criteria to stakeholders and developers
 - Reduce the total lifetime cost of network-centric systems

OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY



Open problems pertaining to RFID anti-cloning and some observations

Benjamin Arazi
Dept. of Computer Engineering and Computer Science
University of Louisville
Louisville, KY 40292

March 24, 2007

1 Motivation

RFID (Radio Frequency IDentification) facilitates the identification and tracing of an object by wireless communication. Its main purpose is to provide automated and efficient product traceability along the entire supply chain.

RFID security issues have been identified and analyzed. In this presentation we treat the threats posed by cloned tags that transmit wrong information, or transmit pre-recorded genuine information at the wrong time or place. This presents a vital need for technologies that facilitate authenticity validation of the tag itself (known as 'object integrity') and its stored and transmitted information. A GAO report on the use of RFID technology dedicates considerable attention to security issues [1], explicitly treating cloning and re-play. Identifying a product at a cashier's terminal, for payment purposes, based on RFID communication with the product, is also associated with cloning and replay threats. Supply chain management reliability depends on information authenticity. Recording valid communication and transmitting it later at the wrong time or place can cause a havoc. It should be noted that a hostile party that wishes to disturb an RFID-based system does not even need to have RFID tags. It can use any kind of transmitter in order to transmit misleading information.

The FDA treats the use of RFID in preventing drug counterfeiting [2]. The following is taken from [3]. "*The inherent problem with EPC technology, from a pharmaceutical perspective, is the lack of anti-cloning features in the EPC chip itself....* With current EPC specifications, it is possible to programme one chip with the exact data of another, effectively cloning the first chip. Without guaranteed authentication, the usefulness of RFID is significantly reduced. META's analysts believe *RFID use within the pharmaceutical industry will be limited to a "track and trace" role until EPC specifications are revised to make cloning more difficult.*" (EPC is Electronic Product Code, the value stored in an RFID tag.)

RFID anti-cloning techniques should basically enable the tag to prove that it stores a secret key unique to the tag. The key should not leak out from the tag, either by physical means or by eavesdropping. Furthermore, the communications should include a random element, to prevent invalid replay.

2 Physical-level considerations

A secured system is as strong as its weakest link. There is no point in protecting a communication channel, if the security at the edges is compromised. As RFID tags can be physically accessed by any party, either a friend or a foe, the secret key stored at the tag should be protected based on techniques whose security strength is not inferior to that of the techniques used in protecting the transmitted data. However, while channel security techniques reached sophistication to an extent where the level of security can be *proved* based on mathematical theorems, current approaches to physical security are based on technical considerations that are more 'art' than science. (e.g., special glue, special coating techniques, clock irregularities.) No fundamental proofs are employed here, beside 'good feeling' of the designers.

Security strength is based on the extent of the inability of an adversary to perform an operation that is easily performed by valid users. This poses two challenges: devising means for executing this principle and devising means for accurately evaluating the adversary's inability. The following are some fundamental considerations that can be applied when implementing provably secured memories.

In nonvolatile molecular memory, a molecule is used to store a charge. Setting and resetting a memory cell is implemented by applying appropriate electric bias. Multilevel molecular memory involves nanowires whose electrical conductance is adjusted by molecules that accept or give electrons. Such principles provide ample possibilities for preventing external probing, whereby the interaction between the probe and the molecule disrupts the stored charge or the electrons that control the conductance of nanowires. The energy needed for reading a stored value by an external tampering probe, and the meaning of the interaction between the probe and the charge or electrons that support the memory, can be exactly evaluated using rules from classical physics. A provably secure memory should be based here on an approach whereby the mere attempt to read a stored value by resorting to illegal means would provably destroy the value attempted to be read.

Secured memories based on established theories from mechanics should also be considered. For example, possible implementation of electromechanical supports that would collapse when an outside probe approaches the structure.

Physical security considerations can also be employed in providing functionality obfuscation. This security application concerns the inability of an adversary to 'reverse-engineer' a transformation, meaning that the entire transformation acts as a secret key. Energy-preserving considerations can also be applied here using thermodynamics laws.

3 Implementation-level considerations

HMAC - keyed Hash Message Authentication Code - is a hash implementation parameterized with a secret key. The output value strongly depends on a variable input and the fixed parameterizing key, whose leakage should be protected when choosing any number of the variable inputs. RFID anti-cloning based on HMAC are an obvious choice, implementing what is known as challenge-response interrogation. However, implementation difficulties have also been raised.

The following representing statement [4] summarizes the issue: "*Tag MAC functionality would allow tags to authenticate themselves, but is beyond current low-cost tag resources*". An accurate analysis [5] states: "*the implementation of the hash standard SHA-1 requires 12,000 logic gates, while the cost constraints of an RFID tag permit the use of no more than 2,500 gates*".

The main problem with current HMAC implementations concerns the need to execute a software code using a general processor. The software code runs a well analyzed mathematical algorithm. However, should resource-demanding highly complex cryptographic schemes be enforced just because they can be mathematically well analyzed? Shouldn't it be the other way around? That is, *an implementation that suits the conditions under which it is intended to operate should first be designed, and then analyzed, with subsequent possible modifications that enhance security while keeping the fundamental infrastructure.*

The very low processing power of an RFID-tag inherently cannot support software-specified HMAC. *There is a need to devise direct hardware implementations of HMAC, in which the security is enforced and analyzed based on hardware design considerations.*

We consider the possibility of using a variation of a stream cipher in implementing an HMAC. The cipher is a symmetric encryptor (i.e., the transmitter and receiver share the same secret key). The shared key forms a seed which generates a pseudo random stream, XORed with the cleartext stream, yielding a ciphertext stream. The receiver, having the same seed key, generates synchronously the same pseudo random stream. XORing this stream with the received ciphertext yields the cleartext back. Stream ciphers operate at a higher speed than block ciphers and have significantly lower hardware complexity.

A fundamental security requirement that should be satisfied by a stream cipher concerns noise characteristics of the generated pseudo random stream and inability to recover the secret seed key, knowing the generated stream. The security of the proposed HMAC is associated with the latter characteristics of the cipher.

As low-cost anti-cloning is based on symmetric cryptography, the issue of key management deserves here special considerations. For example, in pharmaceutical implementations there is a need to provide all intended legal validators with means to check the authenticity of drugs they received. The drug producer that initiates the RFID tags accompanying his shipments, and the validating parties, can be at different locations on the globe. Furthermore, the identity of intended validators is even not known to the drug producer when he seals his shipments. The key-management challenges here are clear.

4 Presentation content

The issues raised above will be treated in detail at the presentation. The possibility of harnessing nanotechnology tools in providing proved physical security will be exhibited. Functionality obfuscation will also be discussed.

A complete compact hardware-based HMAC implementation based on stream ciphers will be presented, accompanied with security analysis. System-level key-management issues will also be treated.

References

- [1] GAO, “Radio frequency identification technology in the federal government,” United States Government Accountability Office, May 2005.
- [2] FDA, “Fda counterfeit drug task force report: 2006 update,” June 2006.
- [3] PharmaTechnologist, “Rfid take-up by pharma will be rapid - eventually,” in-Pharmatechnologist.com, August 2004.
- [4] S. Sarma, S. Weis, and D. Engels, “Rfid systems and security and privacy implications. cryptographic hardware and embedded systems,” *CHES 2002*, vol. 2523, pp. 454–469, 2002.
- [5] M. Ohkubo, K. Suzuki, and S. Kinoshita, “Cryptographic approach to “privacy-friendly” tags,” in *RFID Privacy Workshop*, MA, USA, November 2003.
- [6] D. Henrici and P. Müller, “Hash-based enhancement of location privacy for radio-frequency identification devices using varying identifiers,” in *First International Workshop on Pervasive Computing and Communication Security, PerSec’04*. Orlando, FL, USA: IEEE Press, March 2004.
- [7] eSTREAMPHASE2, “Ecrypt network of excellence,” Update on eSTREAM and SASC 2007, November 2006.
- [8] J. Lano, “List of stream ciphers,” homes.esat.kuleuven.be/~jlano/stream/designs.htm.

Physical protection in mobile constrained devices

Benjamin Arazi

A fundamental problem

Mobile devices like wireless sensors or RFID tags can be physically accessed by any party, raising a particular need for physically protecting the secret key they use in secure communication.

This vulnerability should be treated while also considering the constrained conditions (cost, energy) under which such devices operate.

State of the art

Physical-security state of the art: Current physical security techniques resort to using special glue, coating methods, and the like. **No fundamental proofs are employed here, beside the 'good feeling' of the designers.**

Channel-security state of the art: **The security level of current channel security techniques is proved by mathematical theories.** Solutions must be accompanied with security modeling and proofs.

Motivation for physical security R&D

The security of a system is as strong as its weakest link.

The security of an entire RFID or mobile sensor system is currently based on `art' rather than science.

There is a need to devise physical security mechanisms whose strength can be evaluated and proved based on established theories from physics.

Suggested physical security R&D

Security strength is based on the extent of the inability of an adversary to perform an operation that is easily performed by valid users.

Two challenges:

- **devising means for executing this principle** and
- **devising means for accurately evaluating the adversary's inability.**

The possibility of **harnessing nanotechnology tools in satisfying secured memory execution and evaluation** should be considered.

Molecular memory

In nonvolatile molecular memory, a molecule is used to store a charge. Setting and resetting a memory cell is implemented by applying appropriate electric bias. Multilevel molecular memory involves nanowires whose electrical conductance is adjusted by molecules that accept or give electrons.

Such principles provide ample possibilities for preventing external probing, whereby the interaction between the probe and the molecule disrupts the stored charge or the electrons that control the conductance of nanowires.

More on probe interaction

The energy needed for reading a stored value by an external tampering probe, and the meaning of the interaction between the probe and the charge or electrons that support the memory, can be exactly evaluated using rules from classical physics.

A provably secure memory should be based on an approach whereby the mere attempt to read a stored value by resorting to illegal means would **provably** destroy the value attempted to be read.

Magnetic QCA

Special attention will be dedicated to the possibility of using here magnetic QCA. (Quantum Cellular Automata.) These nano-scale devices use **magnetic particles for both storage and processing**.

The integration of storage within the processor, in a nano-scale frame, opens new horizons in physical memory security. The extremely low power consumption of a magnetic processor is another advantage of this application when considering constrained components like RFID or wireless sensors.

Mechanical considerations

Secured memories based on established theories from mechanics should also be considered.

For example, possible implementation of electromechanical supports that would collapse when an outside probe approaches the structure.

Another aspect of physical security: Functionality obfuscation

The problem:

The possibility of hiding the functionality of a program or a logic circuit has major security implications. The hidden functionality can be regarded as a secret key, known only to the designer.

Trying to obfuscate the functioning of a CMOS logic circuit, it is obviously realized that an adversary can “shave” a circuit layer-by-layer and copy the fabrication masks. The circuit’s functionality can then be analyzed.

Static vs. dynamic ‘reverse engineering’

QCA circuitry lacks fabrication masks that can be shaved off-line, under static conditions, and optically copied. The circuit functionality is rather forced to be analyzed under dynamic conditions. That is, logic values actually have to be measured in order to realize the circuit functionality.

This observation is further enhanced when considering that in order to differentiate here between an AND and an OR gate, there is a need to measure the value of a fixed bit. This makes the problem similar to reading a stored value, making obfuscation similar to storage security.

Mixture of logic gates and conduction lines

In standard microelectronics, **the transistors and the conduction lines are made of different substances**, introduced into the process by different masks. As the masks can always be recovered by reversed engineering, the logic gates and the functionality of the circuit are recovered.

On the other hand, **QCA cells are the building blocks of logic gates as well as the conduction lines that join the gates when forming a logic circuit**. Having the gates and the conduction lines being made of the same cells obfuscates the discrete logic structure.

Hardware encryption based on controlled reversible computation

Traditionally, cryptographic transformations are based on a mathematical algorithm. The algorithm is first devised and analyzed, and then executed in a general CPU or by dedicated hardware. **This approach dictates hardware resources that cannot always be met by constrained components like RFID.**

The possible implementation, directly in hardware, of cryptographic transformations, where the cryptographic strength of the implementation is analyzed based on hardware/physical means should be investigated. This will facilitate the efficient implementation of cryptographic means in highly constrained environments.

Summary

Mobile devices raise a particular need for physically protecting a stored secret key

Current physical security solutions are more art than science

Nanotechnology can be possibly harnessed in satisfying secured memory execution and evaluation

The meaning of the interaction between a probe and a storage cell should be modeled under a variety of implementations

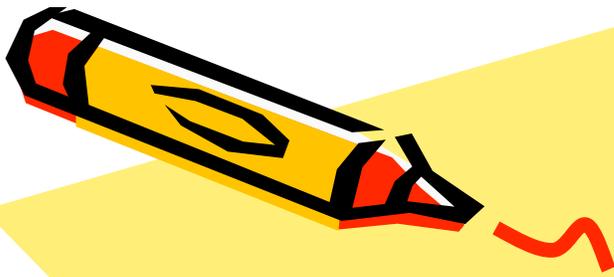
Storage security can be utilized in functionality obfuscation

Physical security mechanisms can be possibly utilized in devising hardware-based secured reversible transformations

**Standards and Interoperability have exposed
Energy Management System Commands and Data
to Cyber Attack
by Dennis Holstein and Jay Wack**

ABSTRACT

The good news is that standards have greatly improved the interoperability of Energy Management System (EMS) components and access to and use of EMS data. The bad news is these improvements have provided means to execute a wide variety of cyber attacks that can disable the EMS system operation or to steal the information that resides in the EMS repositories. The Department of Homeland Security has funded an initiative to address these issues. This work is nearing completion and is ready to be deployed by EMS asset owners to meet the emerging government standards for cyber security. This paper describes the features and capabilities offered in the solution set and describe the tests conducted at the Idaho National Laboratories to evaluate its effectiveness. Specifically, a cryptographic-based schema is used to protect commands throughout the EMS system to the external interface of the EMS servers. The same schema is used to protect data that resides in any EMS repository by controlling access to that data and controlling the use of the data to those who have legitimate access privileges. This solution requires minimal changes to the EMS software and data repositories. Access and use privileges are controlled using an ANSI-based standard that requires two factor authentication supported by a hierarchical security management that can be tailored to any organizational or responsibility need.



EMS Cyber Security

Dennis Holstein, OPUS Publishing
Jay Wack, TecSec

2006-08-19

1

Good news - Bad News

- Standards have greatly improved interoperability and use of EMS data
- Insider cyber attack is getting easier
 - Disable EMS system operation
 - Steal EMS information
- DHS is aggressively sponsoring research to find solutions



2006-08-19

2

Clear statement of need

- Asset owners want a comprehensive solution - not stove pipe or band aids
- Business case needs to address
 - How to recover cost
 - Liability exposure
 - Technical wizardry doesn't sell
- Foundational requirements are addressed



2006-08-19

3

7 foundational requirements

1. AC: Access Control - "Control **access to** selected devices, information or both to protect against unauthorized interrogation of the device or information."
2. UC: Use Control - "Control **use of** selected devices, information or both to protect against unauthorized operation of the device or use of information."
3. DI: Data Integrity- "Ensure the integrity of data on selected communication channels to protect against unauthorized changes."
4. DC: Data Confidentiality - "Ensure the confidentiality of data on selected communication channels to protect against eavesdropping."
5. RDF: Restrict Data Flow - "Restrict the flow of data on communication channels to protect against the publication of information to unauthorized sources."
6. TRE: Timely Response to Event - "Respond to security violations by notifying the proper authority, reporting needed forensic evidence of the violation, and automatically taking timely corrective action in mission critical or safety critical situations."
7. NRA: Network Resource Availability - "Ensure the availability of all network resources to protect against denial of service attacks."



2006-08-19

4

The devil is in the details

- Solutions require cooperation between IT and Operations
 - Security policies must be extensible to accommodate operational constraints
 - Central control (IT) with distributed execution (OPS) is the preferred approach
- Timely response to Event involves everyone
- **Access and Use control is extremely important**
 - The subject of this paper
 - HSARPA initiative: TecSec, GE, OPUS & INL



2006-08-19

5

ANSI X9.69 defines the core technology for RBAC

- X9.69 originally designed for the financial industry
 - ANSI X9.73, X9.93 and X9.96 included
 - Currently being adopted as an ISO standard (ISO 22895)
- Applied successfully to selected critical infrastructure sectors



2006-08-19

6

Cryptographic-based schema

- Protect EMS/SCADA commands
- Protect data residing in any EMS repository
- Control requires legitimate privileges
 - Access to data
 - Use of data
- Minimal changes to EMS software and data repositories



2006-08-19

7

Cool! How does this work?

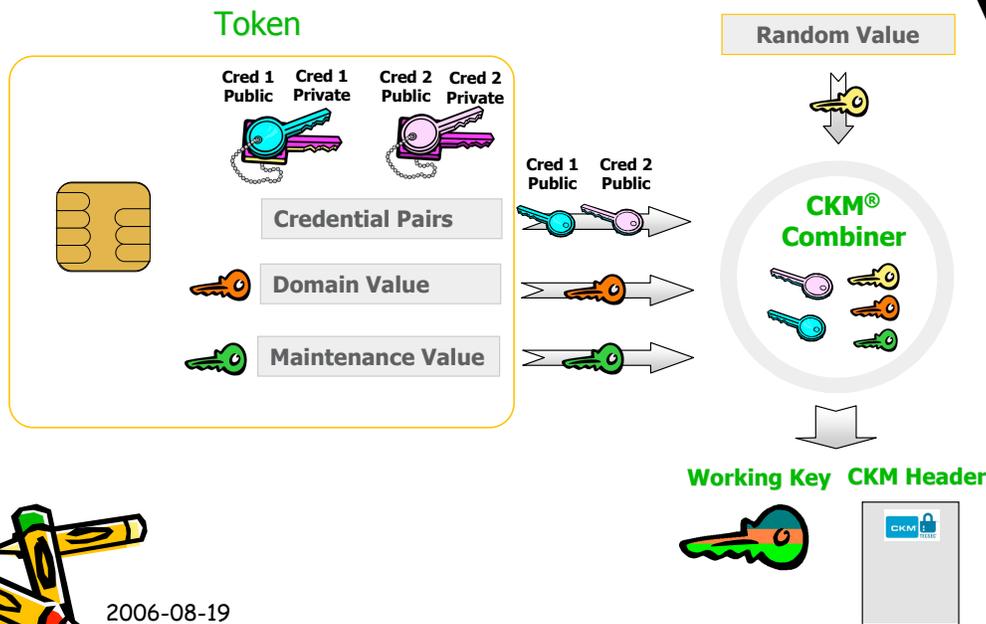
- Control who has access to what using Role Based Access Control (RBAC) & Granular Encryption
- Provide physical & logical access control through Smart Tokens™ and Cryptography
- Integrate the solution into existing business systems and processes



2006-08-19

8

Encryption - logical view



2006-08-19

RBAC roles & credentials

- Roles are established by function/responsibility in Communities of Interest (COI)
- A Role is defined by a set of credentials
 - Each credential represents an attribute
 - Credentials may be further refined by access mode:
 - Read
 - Write
- Individuals who are assigned to more than one Role may be issued multiple credentials reflecting those information access needs
- Individuals assigned the same role, and thus having the same credentials, share the ability to access the same information

2006-08-19

10

Example of who needs what

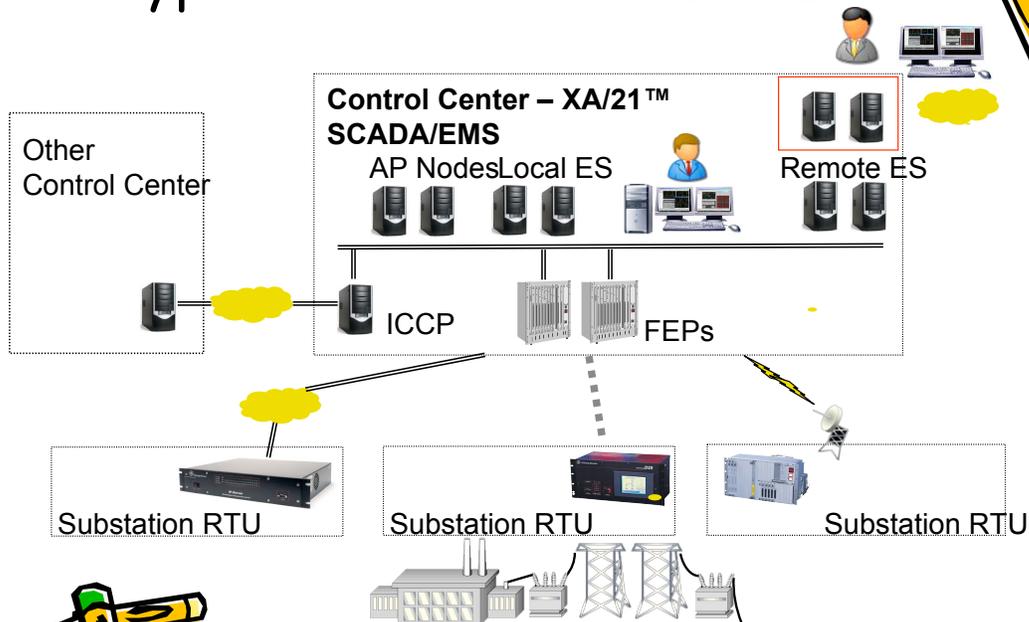
Data types	External Users				Internal to the host utility		
	Power pool member	ISO	Merchant generator	Energy traders	System planning	Crew Dispatch	Revenue Accounting (billing)
Status	R	R			R	R	
Outages					R	R	
Billing data	R @		R @	R @			R
Sched. outages	R	R			R	R/W	
Energy contracts			R @	R @			
Energy bids		R/W @	R/W @	R/W @			

@ = access to only that business entity's own data

2006-08-19

11

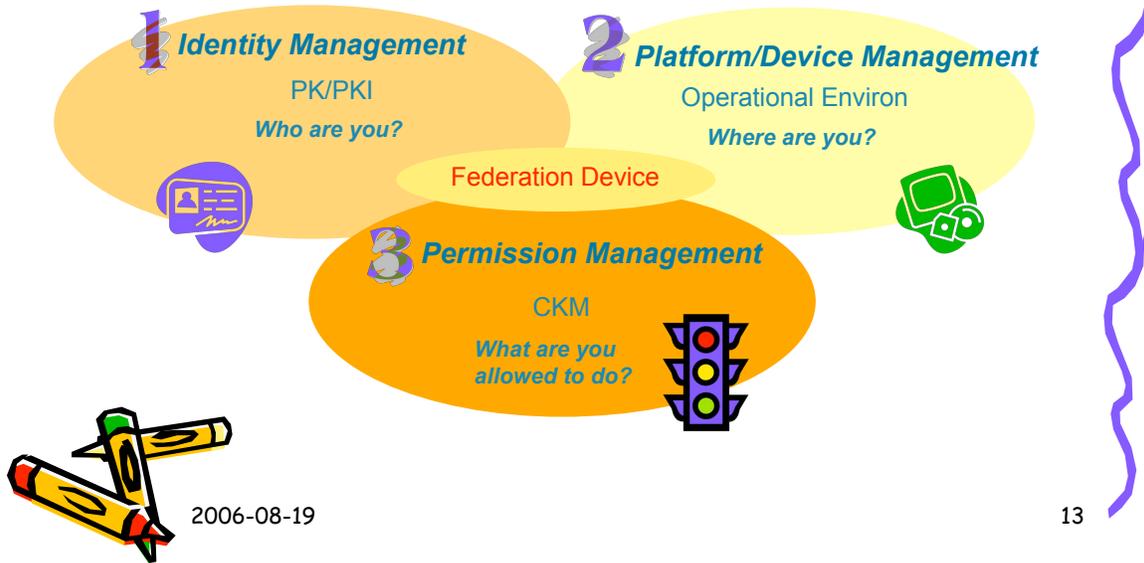
A typical XA/21™ SCADA/EMS



2006-08-19

12

SCADA/EMS Security Implementation



GE has verified security

- All XA/21 programs are digitally signed before being installed on the operational system
- XA/21 validates the digital signature prior to execution and will abort application if it has not been digitally signed
- Every application that directly issues a supervisory control request requires a CKM[®] token with **write** access to a Supervisory Control role
- Every system operator that will be performing supervisory control requires a personal CKM[®] token with **write** access to a Supervisory Control role
- Special logic present in SCS messages to transparently 'pass' (proxy) access control information from originating source
- SVC logic in the Front End Processors have a CKM[®] token that grants it **read** access to Supervisory Control ACL
- SVC checks all supervisory control requests - if they were not issued by authorized actor in the Supervisory Control ACL, it will log and reject the request.



SVC: Supervisory Control
ACL: Access Control Logic

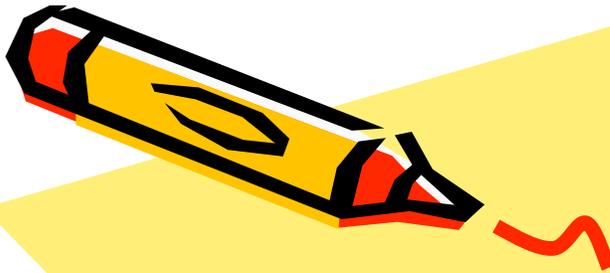
The next steps

- Test security implementation in XA/21 at Idaho National Labs
- Commercialize as an option for future XA/21 release
- Implement CKM-based security in other SCADA/EMS systems
 - Current efforts are underway with Siemens
 - Additional efforts to include this approach in the PJM Power Grid Architecture w/ NERC
- Continue field testing CKM-based security in utility operational environments



2006-08-19

15



Thank you for your attention

Dennis Holstein
holsteindk@adelphia.net
562-716-4174
Jay Wack
jayw@tecsec.com
703-744-8447

2006-08-19

16