Discussion

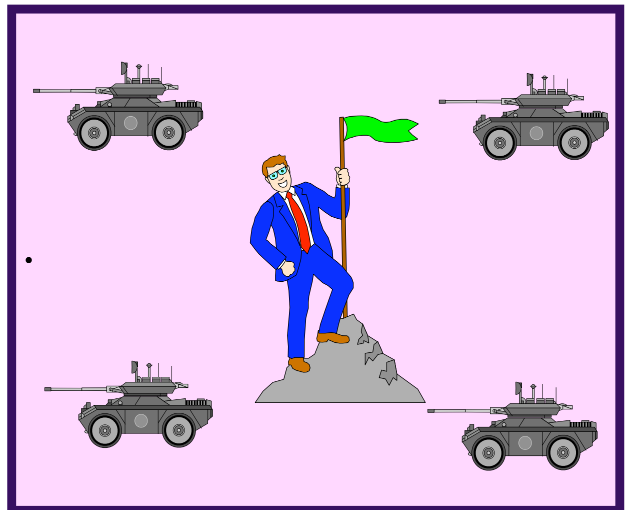
- We will now look at a low level approach to survivability
- There are some definite potential problems
  - During the presentation, think maliciously and identify the weaknesses.

Redundancy: A Curse or Blessing?

- Recall what we said about Redundancy:
- Recovery requirements imply Redundancy
- Three Types of Redundancy
  - Information Redundancy
    - » add information
      - e.g. error correction, authentication, codes
  - Time Redundancy
    - » repeat event in time
      - e.g. multiple sensor readings (of same sensor)
  - Spatial Redundancy
    - » physical redundancy, local or distributed
      - e.g. NMR, k-of-N

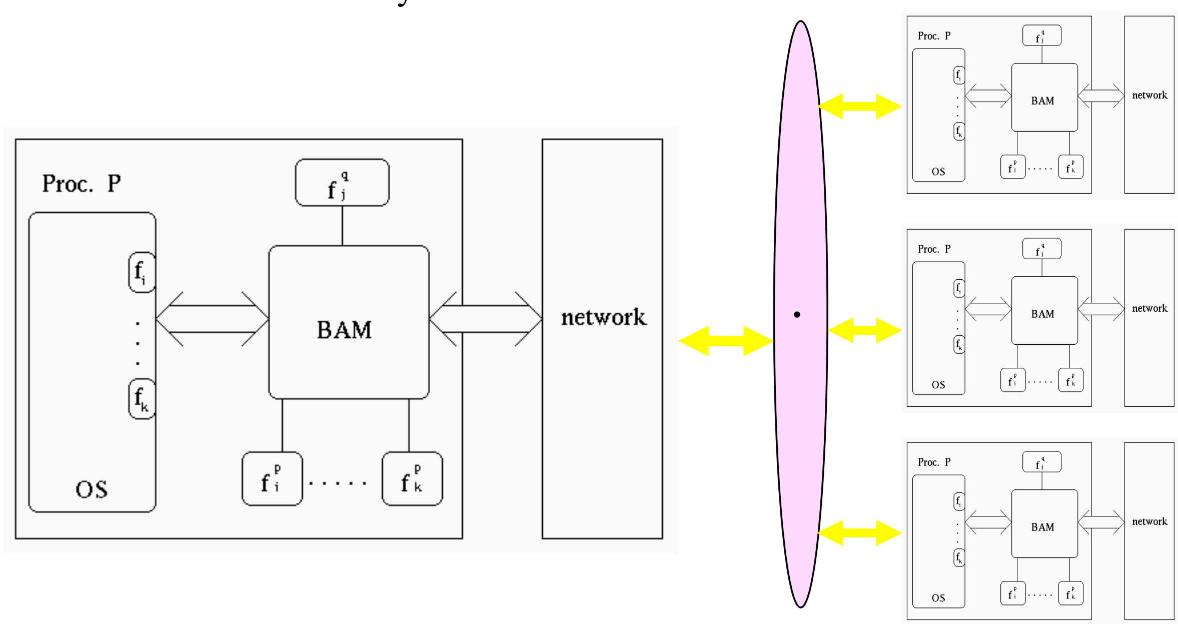
Putting it back together...

- How does one combine results from redundant operations?
- Fault-Tolerant Agreement
  - From Majority Voting to Byzantine Agreement (started with Lamport paper)
  - Many flavors
    - Network Topology
      - bus, ring
    - Network Protocols
      - ATM, TCP/IP, multicast
    - Communication Type
      - symmetric, asymmetric



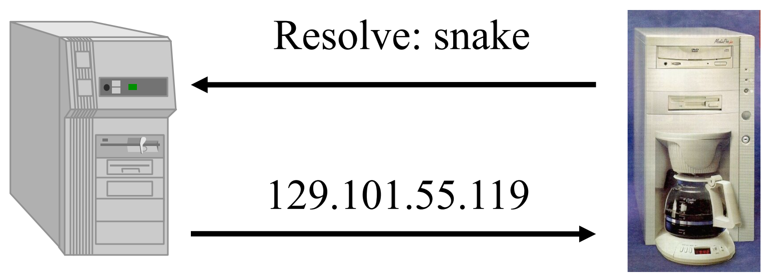
The BRANS Approach

- BAM = Byzantine Agreement Module
  - Survivability Cluster



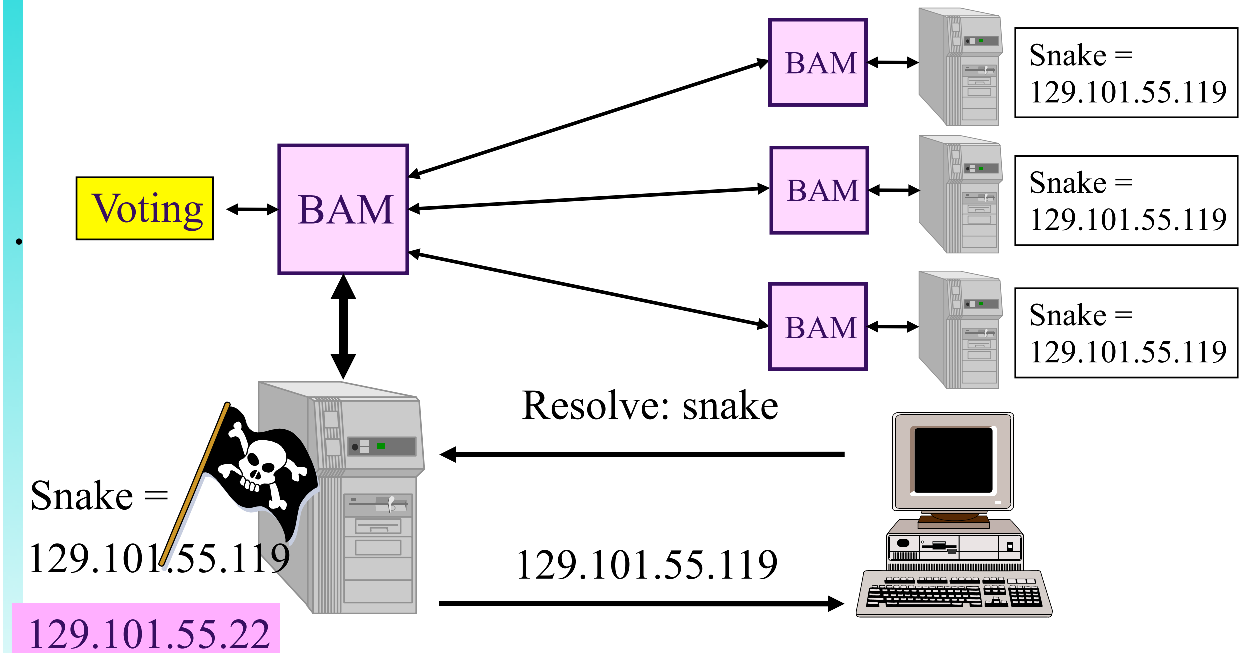
An Example: DNS

- DNS (Domain Name Service)
  - Resolves addresses
    - snake.cs.uidaho.edu = 129.101.55.119
    - DNS server maintains database of mappings



An Example: DNS

- Intruder changed DNS entry



Agreement Requirements

- Solutions with lowest overhead are applied, e.g.
  - simple majority voting,
  - Byzantine agreement with early stopping
  - full Byzantine agreement.
- Individual critical functionalities use those solutions that minimally satisfy their agreement requirements.

Note:

in the previous example a simple majority suffices, however, if the DNS table needs to be updated, stronger agreement solutions are needed that require the 4 computers shown.

Discussion

- Lets play "Devil's Advocate"

Systems under Attack

- How does one tell if a system is under attack?
  - IDSs?
  - How "real-time" should Real-Time be?
  - Decide on a "Level of Abstraction" to be considered.

Systems under Attack

- How can the Whittaker approach be modified to help attack recognition?
  - observing
    - dependencies
    - profiles
    - timing behavior
    - …

Systems under Attack

- We will look at two examples, one is bottom-up and the other top-down.
  - The next discussion is based on the paper
    - "A Two-Layer Approach to Survivability of Networked Computing Systems", by Krings A.W, et.al., *International Conference on Advances in Infrastructure for Electronic Business, Science, and Education on the Internet*, L'Aquila, Italy, Aug 06 - Aug 12, pp. 1-12, 2001.
- We will compare the basic approach with the concepts of the Whittaker paper.

Objective

- Achieve Survivability of Critical Functionalities
  - ultimate goal, holy grail (very general, very difficult)
- "*Some Attacks can be dealt with at Lowest Level*"
- Standard User Environment
- Implementing Survivability Mechanism
  - at the lowest level of abstraction
  - suitable for class of attacks with distinct signatures
  - survivability handlers & response agents

Assumptions

- Anything is possible!
  - » and it will happen!
- Intrusions will occur sooner or later
- Mechanisms that empower can be used against you

Standard User Environment

- Target System
  - Typical desktop computer
  - Mostly operated by single individual
  - Standard applications
    - » browser, email, sftp, ssh, multi-media, text processor, etc.
- System Characteristics
  - Low utilization!
    - » linux `top` command
  - "Idle Profile" of system is surprisingly clean

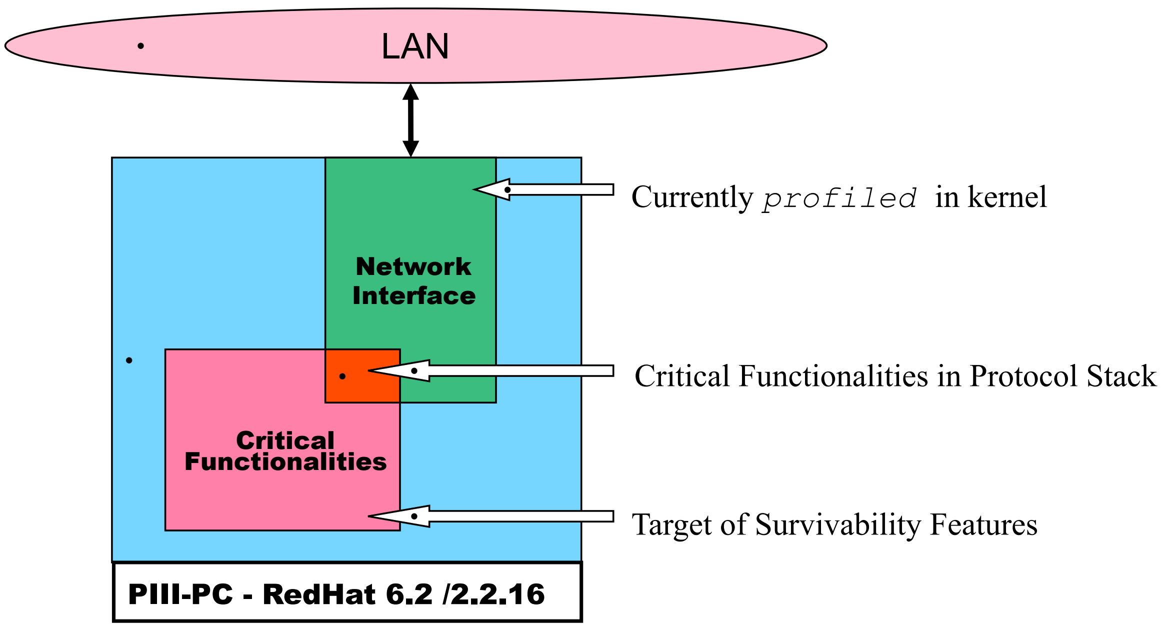
Off-line and On-line Survivability

- Off-line Design Process
  - clean system environment (off-line, no applications)
  - creation of attack signature database
  - attack signatures aid in identification of critical functions
  - implementation of reactionary mechanisms
    - » low level (kernel handlers)
    - » high level (migratory agents)
    - » a priori matching of critical functionalities with critical functions

Off-line and On-line Survivability

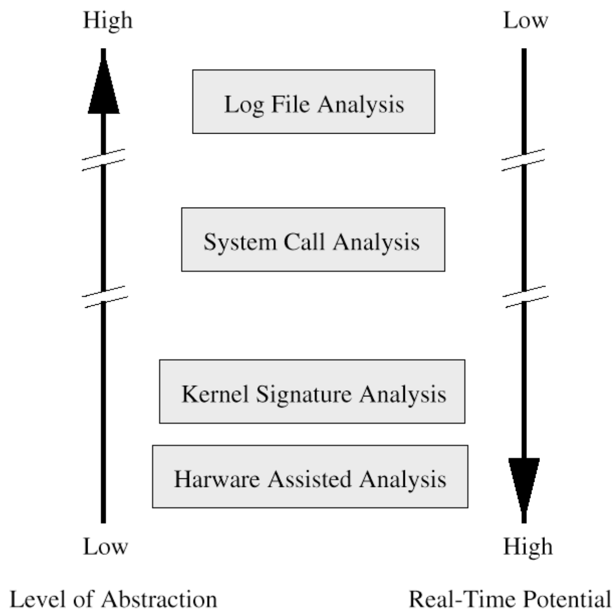
- On-line (real-time) Protective Capabilities
  - real-time attack recognition
  - at high level
    - » recognition triggers response agents
  - at kernel level
    - » survivability handlers get invoked (independent of attack recognition)

System Architecture



Levels of Abstraction

- Real-time Potential



Two Layers of the Architecture

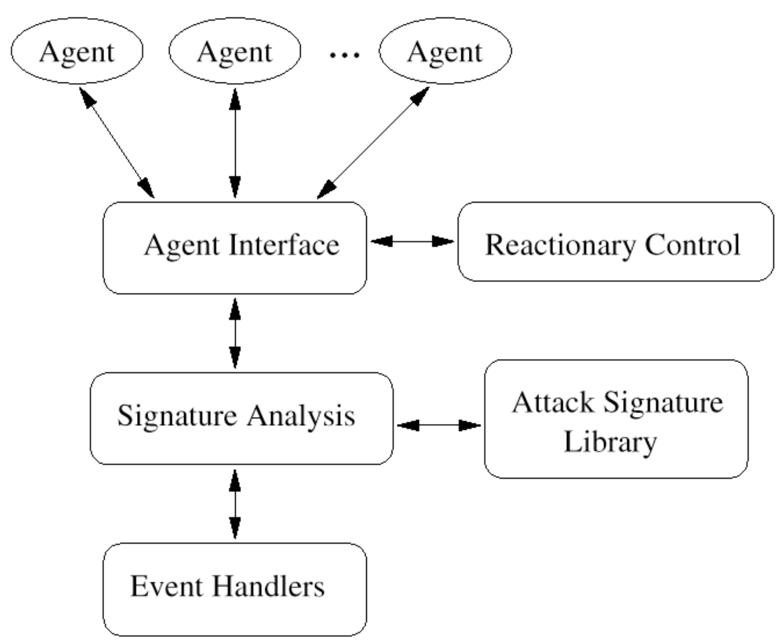
Real-time

- Low-level Event Handlers
  - Survivability handlers
  - Currently used for kernel instrumentation
  - Case study: Early Stopping Agreement

- High-level Reactionary Control
  - Implements high-level survivability features
    - e.g. filtering, patching, early warning
  - Migratory Autonomous Agent System
    - Small specialized program to perform specific task
    - Off the shelf technology, (Aglets)

Survivability Architecture Overview

- System Components



Profiles

- We view a system as a collection of profiles of its functionalities P_i

$$P_{sys}(\Delta t) = \sum_{i=1}^{k} P_i(\Delta t)$$

k is the number of functionalities active during Δt

- Functionality Profile

$$P_i(\Delta t) = (f_1(\Delta t), f_2(\Delta t), \ldots, f_n(\Delta t))$$

$f_j(\Delta t)$ is the number of times identity F_j has been invoked during Δt

Attack Signatures

- Atomic Attacks A_i
  - the smallest attack technology unit
  - e.g. a port sweep, sequence of unsuccessful login attempts
- Attack Signature S_i
  - the portion of a profile that is attributable to A_i

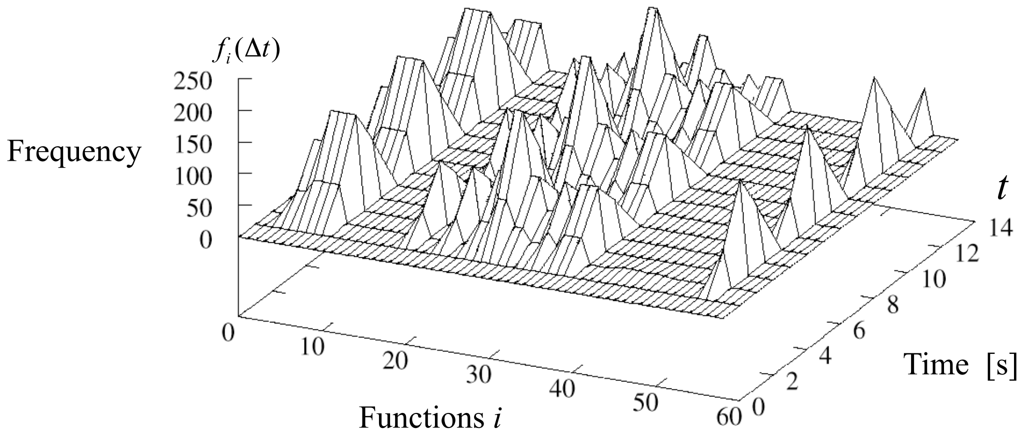
$$S_i(\Delta t) = (f_{\alpha(1)}(\Delta t), f_{\alpha(2)}(\Delta t), \ldots, f_{\alpha(s_i)}(\Delta t))$$

α is a one-to-one mapping from indices of S_i to indices of the identities F_j profiled

$f_j(\Delta t)$ is the number of times identity F_j has been called during Δt

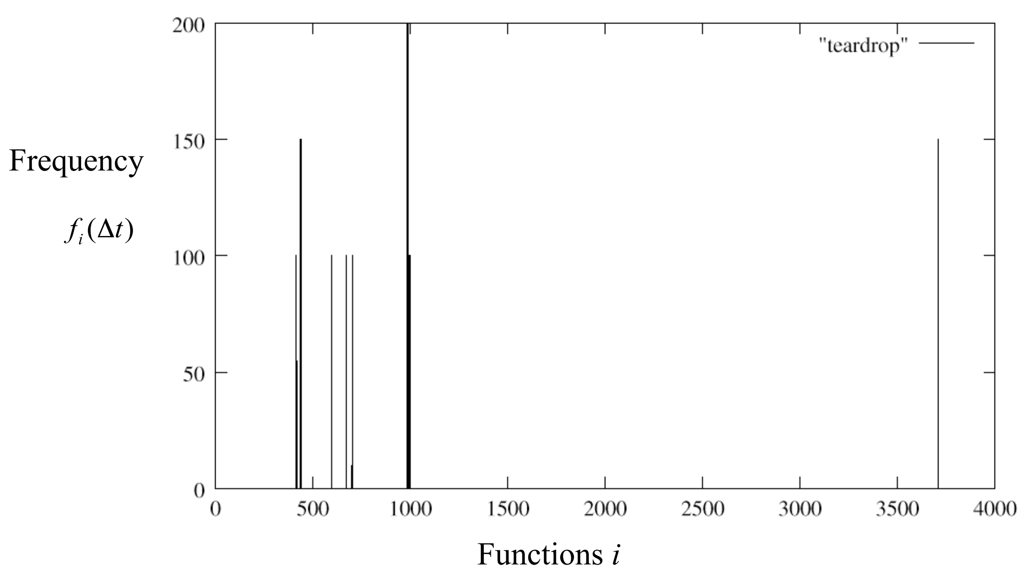
Attack Signature

- Attack Signature over Time
  - Example: “teardrop”
  (overlapping IP(TCP) fragments are formatted to cause reassembly crashes)



Attack Signature

- Example “teardrop”



Real-Time Attack Recognition

- Vector Analysis
  - Profile $P_i(\Delta t)$, Idle Signature $S_0(\Delta t)$, and Attack Signature $S_i(\Delta t)$ are vectors

- "Strictly Speaking"
  - there are three possible scenarios

$$P_{sys}(\Delta t) \geq S_i(\Delta t) \text{ possible attack}$$
$$P_{sys}(\Delta t) \neq S_i(\Delta t) \text{ attack not possible}$$
$$P_{sys}(\Delta t) < S_i(\Delta t) \text{ attack not possible}$$

Signature Analysis

- Relationship between Signatures

$$\mathbf{S}_i \subseteq \mathbf{S}_j$$

- Common functions

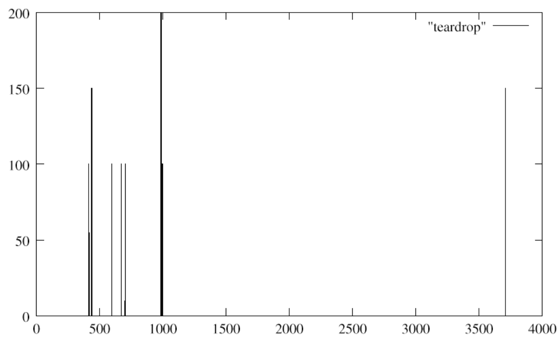
$$\mathbf{S}_i \cap \mathbf{S}_j$$

- Signature Correlation

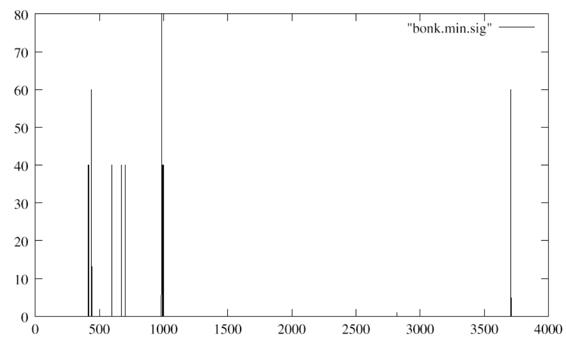
$$C(i,j) = \frac{|\mathbf{S}_i \cap \mathbf{S}_j|}{\min(|\mathbf{S}_i|, |\mathbf{S}_j|)}$$

Attack Signature

- Example “teardrop” vs. “bonk”
  - bonk: malformed IP header causes packet size violation upon reassembly
  - Note: scales differ
  - Correlation is 1.0



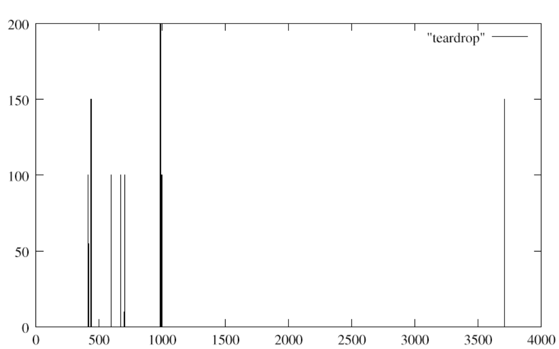
teardrop attack



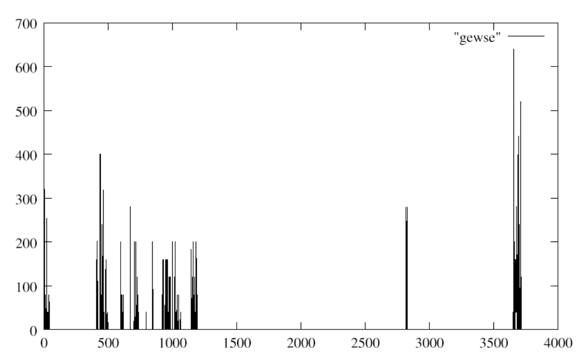
bonk attack

Attack Signature

- Example “teardrop” vs. “gewse”
  - Gewse: (DoS - attack) floods identd on port 139
  - Note: scales differ
  - Correlation is 0.54



teardrop attack



gewse attack

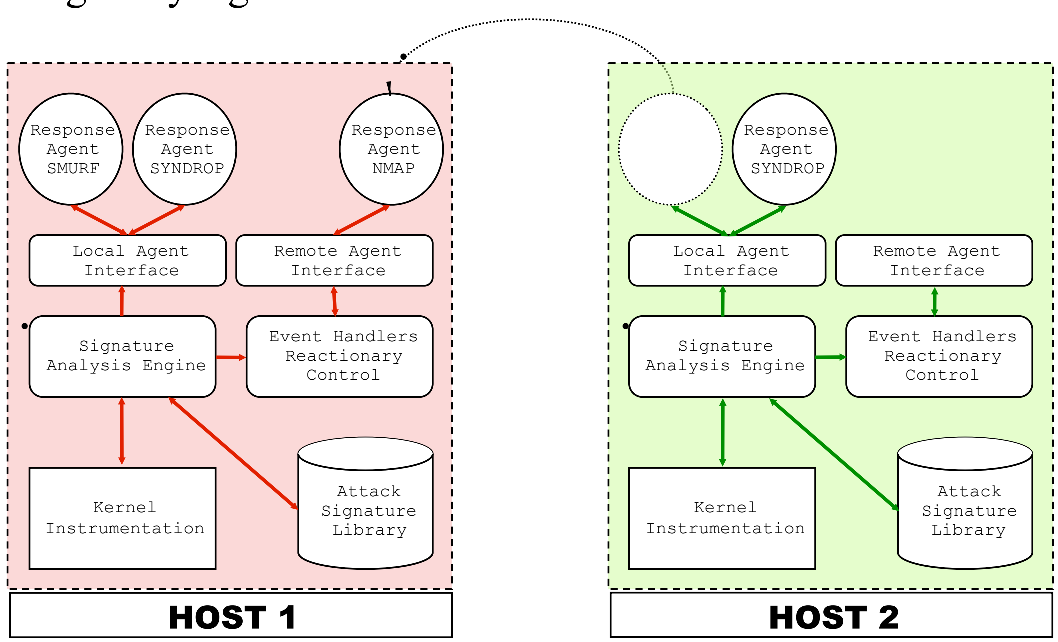
Correlation

- "Some things seem too good to be true"

		13	13	18	18	18	18	20	20	21	21	21	21	21	22	22	24	35	35	37	42	45	51	57	135	164	194
		conseal	misfrag	fawx	jolt	pimp2	ssping	flushot	trash	boink	bonk	newtear	syndrop	teardrop	nestea	smack	dcd3c	beer	spiffit	biffit	synhose	land	pepsi	trash2	gewse	gewse5	hiperbomb2
13	conseal	1.00	0.85	0.85	0.85	0.85	0.85	0.85	0.85	0.85	0.85	0.85	0.85	0.85	0.85	0.85	0.85	0.85	0.92	0.92	0.92	0.85	0.92	0.92	0.85	0.85	0.85
13	misfrag	0.85	1.00	0.85	0.85	0.85	0.85	0.85	0.85	0.85	0.85	0.85	0.85	0.85	0.85	0.92	0.92	1.00	0.85	0.85	1.00	0.77	0.85	0.85	1.00	1.00	1.00
18	fawx	0.85	0.85	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.61	0.61	0.61	0.61	0.61	0.61	0.67	0.61	1.00	0.61	0.61	0.61
18	jolt	0.85	0.85	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.61	0.61	0.61	0.61	0.61	0.61	0.67	0.61	1.00	0.61	0.61	0.61
18	pimp2	0.85	0.85	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.61	0.61	0.61	0.61	0.61	0.61	0.67	0.61	1.00	0.61	0.61	0.61
18	ssping	0.85	0.85	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.61	0.61	0.61	0.61	0.61	0.61	0.67	0.61	1.00	0.61	0.61	0.61
20	flushot	0.85	0.85	1.00	1.00	1.00	1.00	1.00	1.00	0.90	0.90	0.90	0.90	0.90	0.90	0.55	0.55	0.55	0.55	0.55	0.65	0.60	0.55	1.00	0.65	0.60	0.55
20	trash	0.85	0.85	1.00	1.00	1.00	1.00	1.00	1.00	0.90	0.90	0.90	0.90	0.90	0.90	0.55	0.55	0.55	0.55	0.55	0.65	0.60	0.55	1.00	0.65	0.60	0.55
21	boink	0.85	0.85	1.00	1.00	1.00	1.00	0.90	0.90	1.00	1.00	1.00	1.00	1.00	1.00	0.52	0.52	0.52	0.52	0.52	0.52	0.62	0.52	1.00	0.52	0.52	0.52
21	bonk	0.85	0.85	1.00	1.00	1.00	1.00	0.90	0.90	1.00	1.00	1.00	1.00	1.00	1.00	0.52	0.52	0.52	0.52	0.52	0.52	0.62	0.52	1.00	0.52	0.52	0.52
21	newtear	0.85	0.85	1.00	1.00	1.00	1.00	0.90	0.90	1.00	1.00	1.00	1.00	1.00	1.00	0.52	0.52	0.52	0.52	0.52	0.52	0.62	0.52	1.00	0.52	0.52	0.52
21	syndrop	0.85	0.85	1.00	1.00	1.00	1.00	0.90	0.90	1.00	1.00	1.00	1.00	1.00	1.00	0.52	0.52	0.52	0.52	0.52	0.52	0.62	0.52	1.00	0.52	0.52	0.52
21	teardrop	0.85	0.85	1.00	1.00	1.00	1.00	0.90	0.90	1.00	1.00	1.00	1.00	1.00	1.00	0.52	0.52	0.52	0.52	0.52	0.52	0.62	0.52	1.00	0.52	0.52	0.52
22	nestea	0.85	0.85	1.00	1.00	1.00	1.00	0.90	0.90	1.00	1.00	1.00	1.00	1.00	1.00	0.50	0.50	0.50	0.50	0.50	0.50	0.59	0.50	0.95	0.50	0.50	0.50
22	smack	0.85	0.92	0.61	0.61	0.61	0.61	0.55	0.55	0.52	0.52	0.52	0.52	0.52	0.50	1.00	1.00	0.55	0.59	0.59	0.73	0.64	0.77	0.68	0.55	0.55	0.55
24	dcd3c	0.85	0.92	0.61	0.61	0.61	0.61	0.55	0.55	0.52	0.52	0.52	0.52	0.52	0.50	1.00	1.00	0.50	0.54	0.54	0.75	0.63	0.75	0.67	0.50	0.50	0.50
35	beer	0.85	1.00	0.61	0.61	0.61	0.61	0.55	0.55	0.52	0.52	0.52	0.52	0.52	0.50	0.55	0.50	1.00	0.77	0.77	0.57	0.71	0.77	0.77	0.80	0.74	0.80
35	spiffit	0.92	0.85	0.61	0.61	0.61	0.61	0.55	0.55	0.52	0.52	0.52	0.52	0.52	0.50	0.59	0.54	0.77	1.00	1.00	0.43	0.86	1.00	0.91	0.66	0.60	0.66
37	biffit	0.92	0.85	0.61	0.61	0.61	0.61	0.55	0.55	0.52	0.52	0.52	0.52	0.52	0.50	0.59	0.54	0.77	1.00	1.00	0.41	0.86	1.00	0.92	0.62	0.57	0.62
42	synhose	0.92	1.00	0.61	0.61	0.61	0.61	0.65	0.65	0.52	0.52	0.52	0.52	0.52	0.50	0.73	0.75	0.57	0.43	0.41	1.00	0.50	0.50	0.57	0.67	0.64	0.62
45	land	0.85	0.77	0.67	0.67	0.67	0.67	0.60	0.60	0.62	0.62	0.62	0.62	0.62	0.59	0.64	0.63	0.71	0.86	0.86	0.50	1.00	0.87	0.96	0.47	0.42	0.47
51	pepsi	0.92	0.85	0.61	0.61	0.61	0.61	0.55	0.55	0.52	0.52	0.52	0.52	0.52	0.50	0.77	0.75	0.77	1.00	1.00	0.50	0.87	1.00	0.86	0.45	0.41	0.45
57	trash2	0.92	0.85	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.95	0.68	0.67	0.77	0.91	0.92	0.57	0.96	0.86	1.00	0.44	0.39	0.40
135	gewse	0.85	1.00	0.61	0.61	0.61	0.61	0.65	0.65	0.52	0.52	0.52	0.52	0.52	0.50	0.55	0.50	0.80	0.66	0.62	0.67	0.47	0.45	0.44	1.00	0.99	0.95
164	gewse5	0.85	1.00	0.61	0.61	0.61	0.61	0.60	0.60	0.52	0.52	0.52	0.52	0.52	0.50	0.55	0.50	0.74	0.60	0.57	0.64	0.42	0.41	0.39	0.99	1.00	0.96
194	hiperbomb2	0.85	1.00	0.61	0.61	0.61	0.61	0.55	0.55	0.52	0.52	0.52	0.52	0.52	0.50	0.55	0.50	0.80	0.66	0.62	0.62	0.47	0.45	0.40	0.95	0.96	1.00

Network Survivability Architecture

- Migratory Agent Framework



Case Study "Smurf"

- "Smurf" Attack
  - DDoS (limited protection against such attack)
  - attacker:
    - sends ICMP echo packets to generate multiple replies
    - attacker claims to be victim
      - forges source address
    - target of echo request is
      - all machines in broadcast subnet
      - *"Amplifier network"*
  - victim:
    - all systems in amplifier network respond
    - victim gets flooded with unwanted ICMP echo replies
- Response Agent
  - turns on filter in router

Conclusions

- Tow-layer approach to survivability
  - off-line and on-line component
- Low layer
  - Attack signatures aid in identification of critical functionalities
  - Survivability handlers applied at kernel level
  - Signature analysis triggers response mechanism at high level
    - attack recognition does not facilitate a general IDS!
- High layer
  - Migratory Agent system
  - Response agents act as reactionary mechanisms