Case Study: SITAR

- We consider SITAR
  - Source of the discussion is the paper:
    - *SITAR: A Scalable Intrusion-Tolerant Architecture for Distributed Services,*
    - by Feiyi Wang, Fengmin Gong, Chandramouli Sargor, Katerina Goseva-Popstojanova, Kishor Trivedi, Frank Jou,
    - Proc 2001 IEEE Workshop on Information Assurance and Security, United States Military Academy, West Point, NY, 5–6 June 2001

SITAR

- Main issues
  - focus on “continuation of operation”
  - utilize redundancy and diversity
  - architecture components:
    - » proxy servers
    - » acceptance monitor
    - » ballot monitors
    - » adaptive reconfiguration
    - » audit control

Focus on attacks or their effect?

- Should we focus on attacks?
  - if yes, then what attacks?
  - all attacks are not even known.

Focus on attacks or their effect?

- Shift from attacks or attacker to target of protection.
  - "the effect of attack is more important than the cause of the attack"
  - Focus on essential functionalities instead
    - » recall that we specified a system as the union of all functionalities
    - » these functionalities may have different fault assumptions

SITAR objectives

- Scalable Intrusion-tolerant Architecture for Distributed Services (SITAR)
  1. uses network-distributed services based on COTS components (paper discusses web services)
  2. utilize fault-tolerance (FT) mechanisms
     - FT specific: redundancy and diversity
     - malicious act: external attacks & compromised components
  3. dynamic reconfiguration

SITAR Architecture

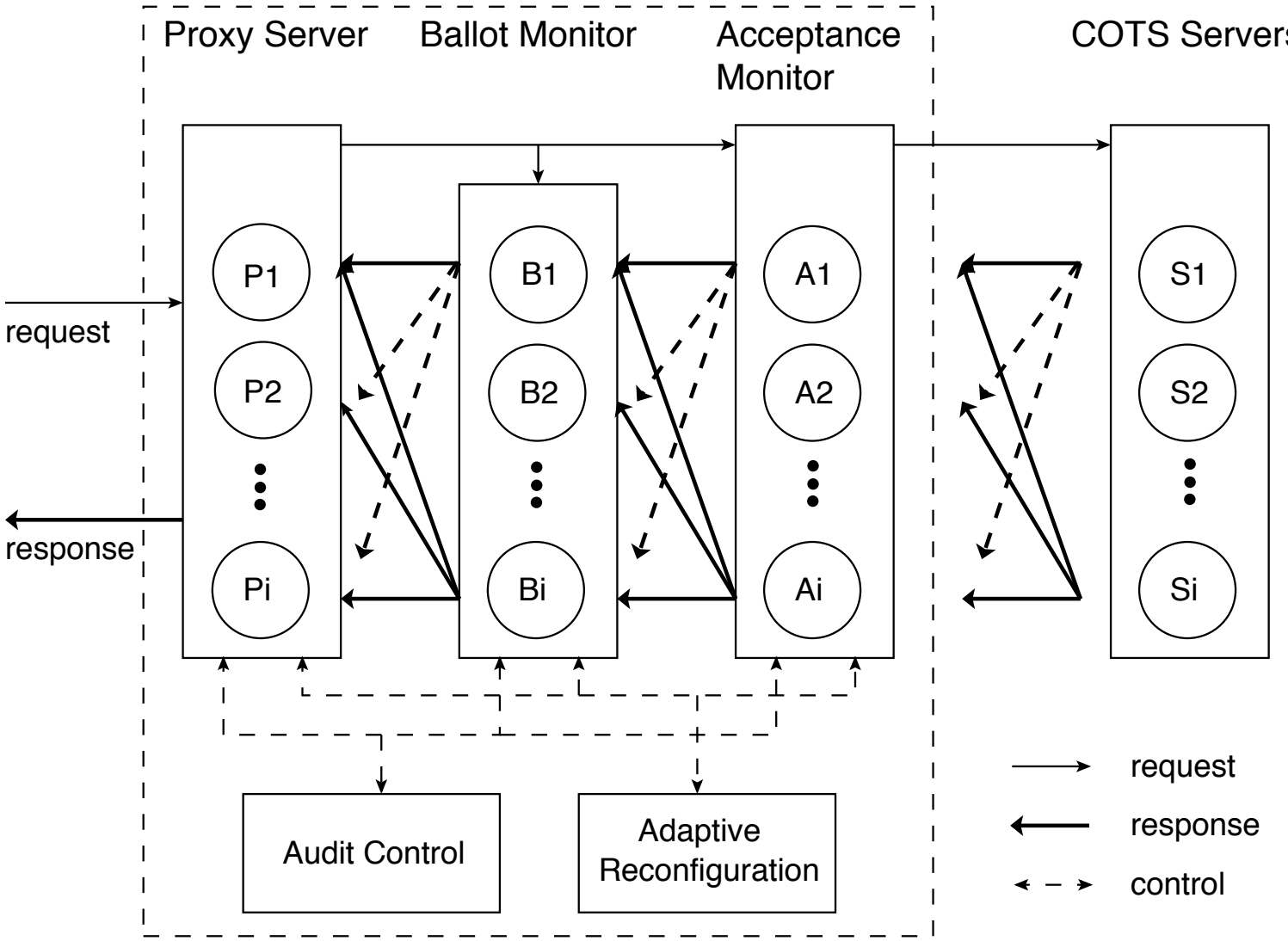
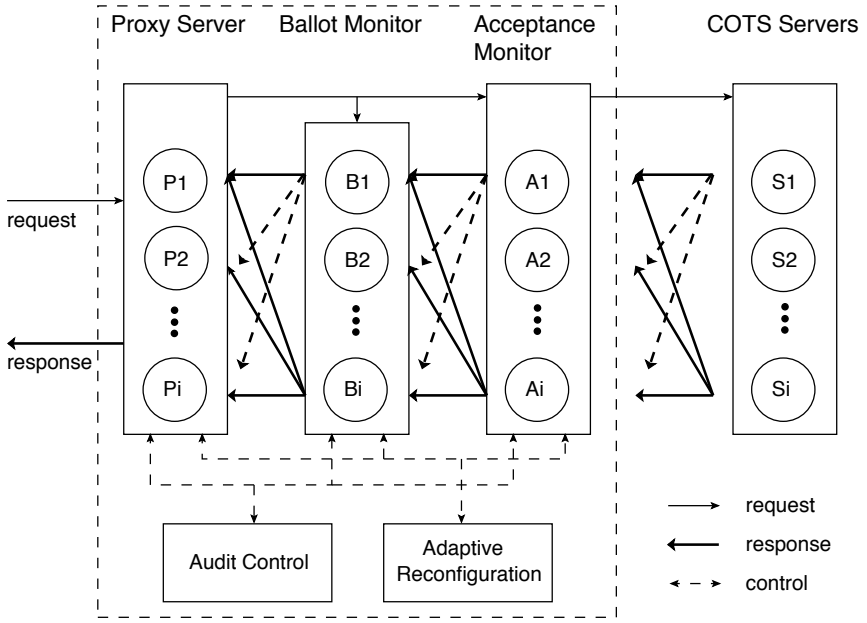


Fig. 1 generic intrusion-tolerant service architecture

Proxy Servers

- What is a Proxy Server?
- Proxy Servers
  - requests are
    - » not made to servers
      - ■ may not be known
    - » made to proxy
  - have physical IP addr.
  - share pool of virtual IP addresses



Proxy Servers

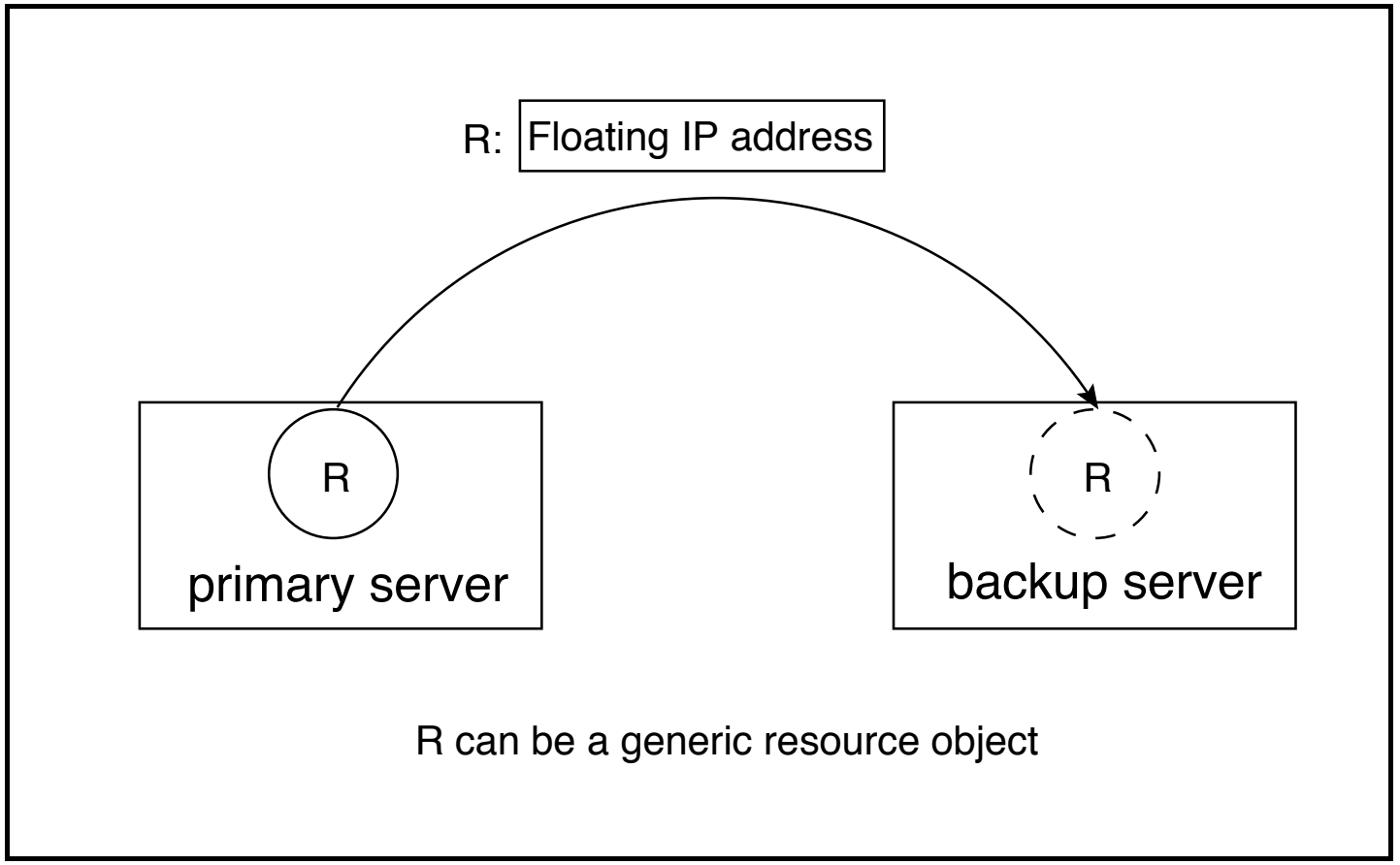


Fig. 2. Primary/backup operation mode

Proxy Servers

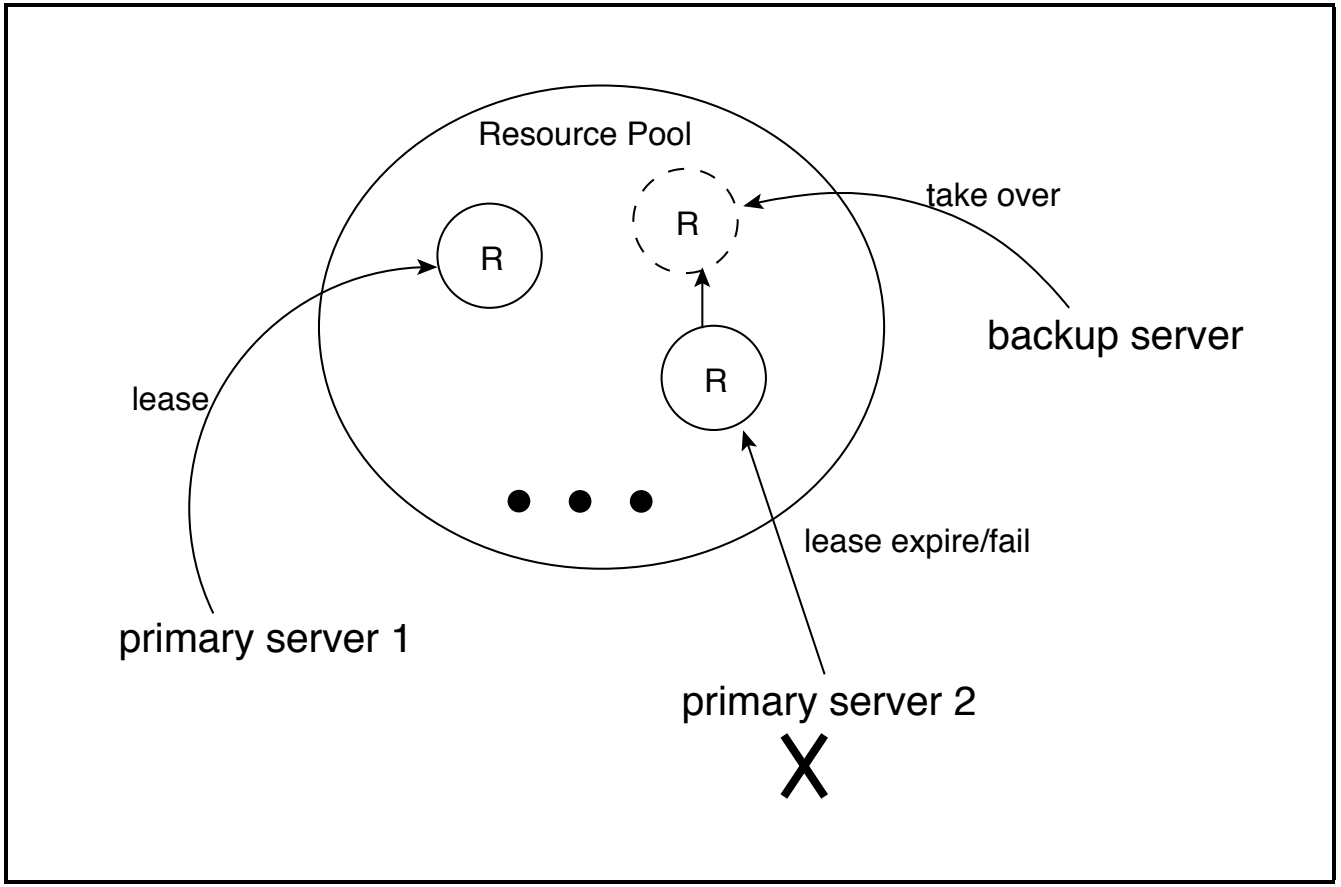


Fig. 3. Shared resource pool operation mode

Proxy Servers

- Proxy Server needs to maintain state tracking
  - requests issued by clients
  - set of servers
  - set of ballot monitors
  - set of acceptance monitors
  - virtual IP address, etc.

Proxy Servers

- What issues are involved when a set of proxy servers needs to have agreement about the "global" state of the global shared state?
  - need *reliable broadcast*
    - what is that again?
  - they use Javaspaces as it has a mechanism that acts as reliable broadcast
    - allows to *take* state object of the space and, after updating, to *write* it back
    - *take* is only committed if subsequent *write* operation is completed as well

Proxy Servers

- Proxy Servers responsibilities:
  - receive client requests
  - forward them to the actual COTS servers
  - decide (per request) which servers to forward request to
  - correlate request with response from Ballot Monitor
  - forward result back to client

Proxy Servers

- Proxy Servers objectives
  - form front line of intrusion tolerant architecture
  - employ IDS per proxy server
    - » monitor network traffic for external attacks
    - » monitors other proxy servers
- When IDS detects attack or compromised peer
  - it invokes Adaptive Reconfiguration Module (ARM)
  - ARM will reconfigure if necessary

Acceptance Monitors

- Responsibilities
  - Process response from COTS servers and apply acceptance tests
  - Response and result of acceptance tests are forwarded to the Ballot Monitors
  - Detect intrusions in COTS servers
  - Alert ARM

Acceptance Monitors

- What is an acceptance test?
  - check reasonableness of results
  - highly application dependent
  - forms basis for recovery block scheme

Acceptance Monitors

- What is a recovery block?
  - invented by B. Randell in early 70s
  - check out:
    - *The Evolution of the Recovery Block Concept,* Brian Randell and Jie Xu, in Software Fault Tolerance, 1994

```
ensure      acceptance test
by          primary alternate
else by     alternate 2
            .
            .
else by     alternate n
else error
```

Acceptance Monitors

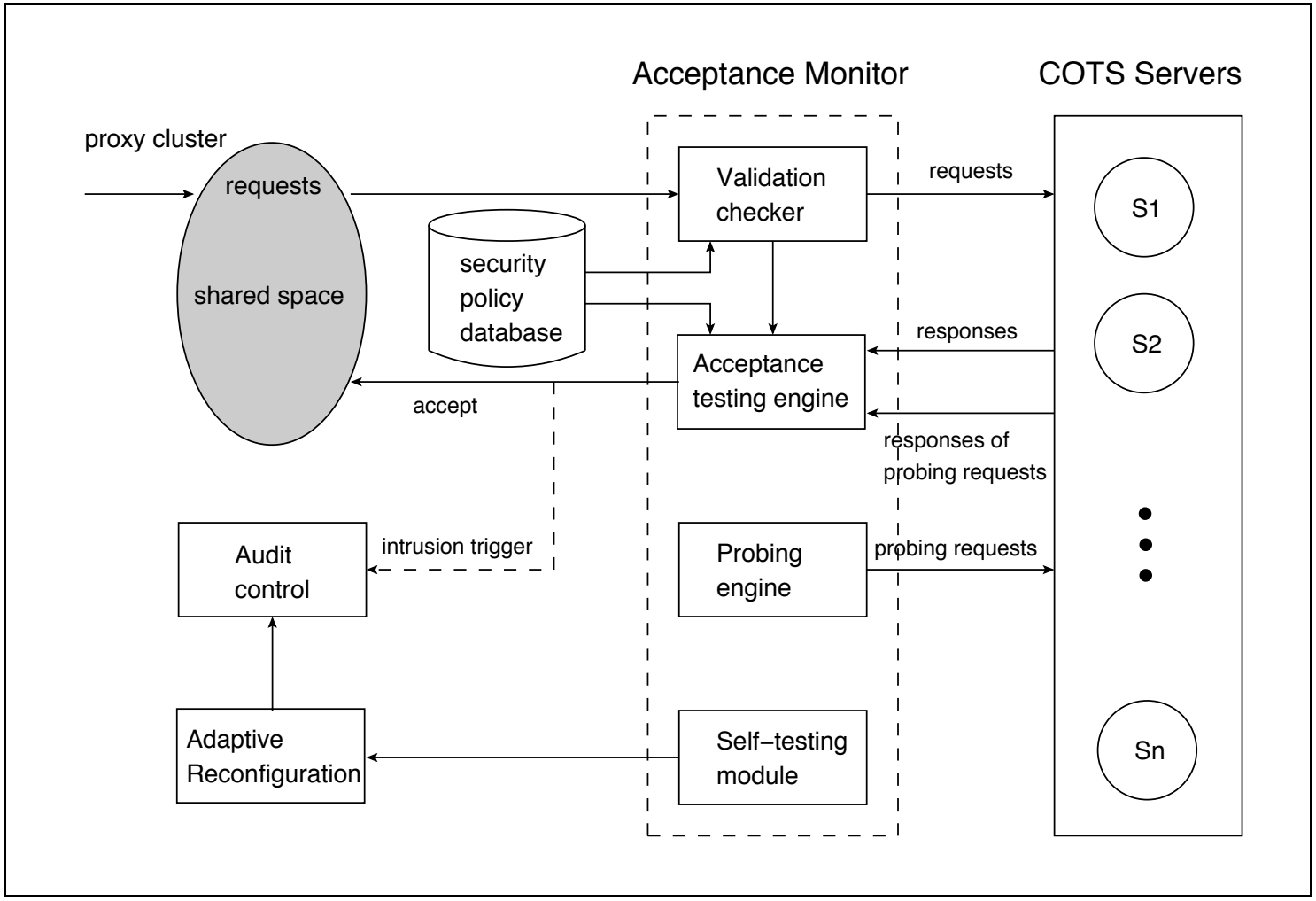


Fig. 4. Architecture of Acceptance Monitor

Acceptance Monitors

- Acceptance Tests
  - Satisfaction of requirements
  - Accounting tests
  - Reasonableness test
  - Computer runtime checks
  - etc.

Ballot Monitors

- This is where the decision about the final response is be made
  - voting/agreeing scheme depends on fault model
    - » e.g., majority, Byz. agreement algo
  - before voting transformations are conducted at three levels
    - » Level 1: Fletcher checksum
    - » Level 2: MD5 checksum
    - » Level 3: Keyed MD5 checksum

Ballot Monitors

- Fletcher's Algorithm
  - developed in late 70s
  - error detection approaching CRC quality
  - algorithm
    - » use two 8-bit check values
    - » first byte is the sum of all values (div. 255)
    - » second byte is the sum of all values of the first check bytes (div. 255)
    - » the checksum itself is not transmitted; instead a value is compute from checksum which causes receiver to produce a checksum of 0

Ballot Monitors

- MD5 checksum
  - cryptographic hash function with 128-bit (16Byte) hash value
  - used to verify data integrity
  - RFC 1321

Ballot Monitors

- Keyed MD5 checksum
  - sender and receiver share secret key *k*
  - sender runs MD5 checksum over concatenation of message *m* and *k*. Then *k* is taken out again.
    - » sender transmits $m + \text{MD5}(m + k)$
  - can use encryption as well
    - » sender picks *k* at random, encrypts it using RSA and receiver's public key; then encrypts with result with own private key; sender transmits
    - » $m + \text{MD5}(m+k) + \text{E}(E(k, rcv\_public), \text{snd\_private})$

Ballot Monitors

- Issues about voting
  - Schemes of mapping voters to processes
  - Guards and "who guards the guards"

Audit Control

- "*Audit is defined as the independent examination of records and activities to ensure compliance with established controls, policy, and operational procedures, and to recommend any indicated changes in controls, policy, or procedures* [9]."
- Individual audit records maintained by
  - each of the following)
  - Proxy Server
  - Acceptance Monitor
  - Ballot Monitor
  - ARM

Audit Control

- What is the purpose Audit Records?
  - derive IDS capabilities
- What data is collected and at which level?
  - generation at high level may allow low level attacks to go undetected
  - generation at low level may generate data overload

Audit Control

- Audit Control Module diagnostics functions
  - maintains results of diagnostic tests
  - maintains test suits (tests and responds)
    - what is *spot-checking*?

Intrusion Detection Systems

- At Proxy Server
  - detect intrusions from outside
  - keep track of usage patters,
    - » e.g., request frequency,
- At Acceptance Monitor
  - detect compromises in COTS servers
  - monitor system activities of COTS servers
    - » e.g., CPU utilization, disk space, paging
    - » software aging tests

Dynamic Reconfiguration

- Strategies depend on
  - fault model, redundancy levels, detection methods etc.
- Different ideas
  - "shifting gears" - change strategies on the fly
  - "graceful degradation" toward critical services
  - adapt reconfiguration to current situation
    - » levels of security threat
    - » configuration as a balance of fault model and available resources

Impact

- The general ideas of the paper collect known fault tolerance and security methods
- The general concepts shown in Figure 1 can be used for many applications
  - most intrusion tolerant systems can be somewhat expressed by high-level mapping of these mechanisms