Byzantine Tolerant Group Communication Systems

Ziad El Bizri
October 6th, 2004
Intrusion Tolerance by Unpredictable Adaptation
(http://itua.bbn.com/, http://www.perform.csl.uiuc.edu/itua.html)

Supported by the Defense Advanced Research Projects Agency (DARPA)

A joint effort by BBN Technologies, the University of Illinois, the University of Maryland, and Boeing
Main approach

- Develop a robust decentralized intrusion-response mechanism
  - Employs intrusion tolerance in multiple layers: Group Communication System, Gateways and Managers
  - Uses unpredictability in adaptive response
  - Exploits redundancy to tolerate component failures

- Two main assumptions
  - Attack model is *staged*: an attacker can only attack one domain after another
  - Intrusion detection is reliable: corrupt processes can be detected effectively
Architecture
Host

- **Subordinate**
  - Forms a *subordinate group* with other subordinate hosts and the manager host in the security domain
  - In *security advisor* role: collect information, reacts locally to events and reports to the domain manager
  - In *replication management* role: responsible for starting and killing replicas

- **Manager**
  - Forms a *manager group* with all other manager hosts across all security domains
Group communication primitives

- Built on existing secure group communication systems
- Group Membership Protocol
  - Maintains group membership: removing corrupt processes and joining new processes
- Reliable Multicast Protocol
  - Two phase protocol
  - Uses cryptographic primitives
- Total-Ordering Protocol
  - Ensures consistency by providing global sequence numbers
MAFTIA

- Malicious-and Accidental-Fault Tolerance for Internet Applications (http://www.maftia.org)
- A European joint project by University of Newcastle, Universidade de Lisboa, Qinetiq, IBM Zurich, LAAS-CNRS, Saarland University
- Three main areas of work
  - Architecture of MAFTIA
  - Design of mechanisms and protocols
  - Formal verification and assessment
Failure models

- Controlled failure assumptions
  - Failures are bounded

- Arbitrary failure assumptions
  - Byzantine behavior

- Hybrid failure model
  - Some parts of the system can exhibit arbitrary failures, while other parts can be entirely trusted (controlled failure)
  - Every subsystem must be modeled

- Composite failure model
  - Represent failures resulting from different classes of faults
  - Define a set of local techniques to handle distributed failures
Fortress model

- Uses composite failure model
- Recursive use of fault tolerance and fault prevention
  - Removal of internal vulnerabilities (patching)
  - Prevention of attacks (IDS)
  - Intrusion tolerant mechanisms inside the components
Trusted Timely Computing Base

- Small component that can be formally verified
- Trusted: can only exhibit a fail-stop behavior (non Byzantine)
- Provides trusted version of Timely Computing Base services
  - Trusted random number generation
  - Trusted absolute timestamping
  - Trusted block consensus
  - Trusted block equality test
  - Local authentication
  - Distributed authentication
Node architecture

- Two level hierarchy: *participant* level and *site* level
- A *participant-group* is mapped to a *site-group* (containing all the sites of the participants in the participant group)
- Site level
  - Multipoint network module (for multicast communication)
  - Site failure detector (assessing connectivity and correctness of sites)
  - Site membership (creates and maintains membership and view of site-groups)
  - Communication support services module (basic cryptographic primitives)
Node architecture (Cont’d)

- Participant level
  - Participant failure detector module (assess liveness of local participants)
  - Participant membership module (creates and maintains membership and view of participant-groups)
  - Activity support services module (replication and transaction management)
System architecture and Security

- Network (arbitrary failure model)
- Runtime environment (OS, protocol kernel, TTCB)
  - Must be made fail controlled
  - Select an OS that is as trustworthy as possible
  - Patch it (remove known vulnerabilities)
  - Use intrusion detection and countermeasures
  - Protect the host (close unused user accounts, strong passwords, etc…)
  - Protect protocol kernel from buffer overflow and input validation attacks
System architecture and Security (2)

- Site level abstraction must be protected
  - Attacks from OS kernel and the network (obfuscation of the code, protection from buffer overflow and input validation)
  - Joins and leaves of sites have to be secured (TTCB trusted block equality test, TTCB distributed authentication service, all sites must agree before accepting a new site into the group)
  - Communication over the network has to be secured (Encryption, checksum generated by TTCB, key management)

- Participant level
  - Must be built trustworthy (as before)
  - Participant join decision is voted upon by all participants
  - Secure identification using ID/password or secret key
Conclusion

- Two architectures based on Intrusion Tolerance
  ITUA and MAFTIA