Preventing coordinated attacks via alert correlation

J. Garcia, F. Autrel, J. Borrell, Y. Bouzida

S. Castillo, F. Cuppens, G. Navarro

{jgarcia, jborrell, scastillo, gnavarro}@ccd.uab.es,

{fabien.autrel, yacine.bouzida, frederic.cuppens}@enst-bretagne.fr
Main Points

- Introduction
- Classical architectures
- Prevention framework
- Current Development
- Conclusions
1. - Introduction

Coordinated Attacks

- “Combination of actions performed by a malicious adversary to violate the security policy of a target computer system.”

- Networks resources can become an active part of a coordinated attack

- E.g. An attack might start with an intrusion
  ⇒ Nodes have to be monitored

- A global view of the whole system is needed for detection
  ⇒ Collection and combination of events from different nodes
1. - Introduction

Components needed to prevent coordinated attacks

- Sensors (host, application or network based)
- Analyzers (misuse or anomaly based)
- Managers (data consolidation and alert correlation)
- Response units (active or passive reaction)

- Intrusion Detection Systems use these same components to prevent a node getting compromised by an attacker

⇒ We use these components to prevent a compromised node becoming an active part of a coordinated attack.
2. - Classical architectures

Centralized event correlation

- DIDS - University of California, Davis (1991)
- STAT - University of California, Santa Barbara (1992)
2. - Classical architectures

Hierarchical event correlation

- AAFID - CERIAS, Purdue University (1998)
Message passing architecture

⇒ The detection process can be completely distributed
3. - Prevention framework

Sample scenario

- **Victim**
  - Victim PC
  - Query and spoofed resolution for the domain name
  - Flooding spoofed recursive replies

- **Authoritative Nameserver**
  - DoS
  - SYN flooding
  - Flooding of queries for the domain name

- **Attacker**
  - Fake resolution
3. - Prevention framework

Detection Process

Find the set of actions which transforms the system from an initial state $S_0$ to a final state $S_n$.

- action syn-flood($A,H_1,n_s$)
  - pre: remote-access($A,H_1$)
  - send-multiple-tcp-syns($A,H_1,n_s$)
  - post: deny-of-service($H_1$)
- detection: source(Alert,$A$)
  - target(Alert,$H_1$)
  - classification(Alert,'SynFlooding')
  - additional-data(Alert,$n_s$)
3. - Prevention framework

Detection process via alert correlation

- Two actions $A$ and $B$ can be correlated when the realization of $A$ has a positive influence over the realization of $B$ (given that $A$ occurred before $B$):

\[ (E_a \in post(A) \land E_b \in pre(B)) \lor (\neg E_a \in post(A) \land \neg E_b \in pre(B)) \]

- $E_a$ and $E_b$ are unifiable through a unifier $\theta$
3. - Prevention framework

Reaction process via anti-correlation

- Two actions \( A \) and \( B \) are anti-correlated when the realization of \( A \) has a **negative influence** over the realization of \( B \) (given that \( A \) occurred before \( B \)):

\[
\begin{align*}
\text{(not}( E_a ) & \in \text{post}(A) \land E_b \in \text{pre}(B)) \lor (E_a \in \text{post}(A) \land \text{not}(E_b) \in \text{pre}(B)) \\
\text{\( E_a \) and \( E_b \) are unifiable through a unifier } \theta
\end{align*}
\]

\[A\] action undo-deny-of-service\((A,H_1,n_s)\)  
  pre:  deny-of-service\((H_1)\)  
  send-multiple-tcp-resets\((A,H_1,n_s)\)  
  post: not(deny-of-service\((H_1)\))

\[B\] action flooding_spoofed_replies\((A,H_1,H_2,N,IP,n_r)\)  
  pre:  remote-access\((A,H_1)\)  
  send-multiple-spoofed-replies\((A,H_1,H_2,N,IP,n_r)\)  
  wait-recursive-reply\((H_1,H_2,N)\)  
  deny-of-service\((H_2)\)  
  post: legitimate-recursive-query\((H_1,N,IP)\)
3. - Prevention framework

Detection and reaction graph for the sample scenario

counter-measure:
undo-deny-of-service(A,H₂,nₛ)
pre  :  deny-of-service(H₂)
send-multiple-tcp-resets(A,H₂,nₛ)
post: not(deny-of-service(H₂) )

action:
syn-flood(A,H₂,nₛ)
pre :  remote-access(A,H₂)
send-multiple-tcp-syns(A,H₂,nₛ)
post: deny-of-service(H₂)

counter-measure:
block-spoofed-connection(A,H₁,H₂)
pre :  spoofed-connection(A,H₁,H₂)
post: not(spoofed-connection(A,H₁,H₂) )

action:
flooding-queries(A,H₁,H₂,N,n₉)
pre :  remote-access(A,H₁)
send-multiple-queries(A,H₁,N,n₉)
post: wait-recursive-reply(H₁,H₂,N)

objective illegal-recursive-query(H₁,N,IP)
state : legitimate_recursive_query(H₁,N,IP)
not(legitimate_recursive_query(H₁,N,IP))

Correlation

4. - Current Development

Current Development

![Diagram of Elvin publish-subscribe system with Elvin publish-subscribe federated multicast channel and sensors, response units, and events responses.]
5. - Summary

Results of our work

- State of the art about coordinated attack prevention
- Study about alert correlation mechanisms
- Development of a generic framework avoiding bottleneck of centralized architectures using a distributed approach
- Both detection and reaction are performed by using the same formalism
5. - Summary

Future work

► Incorporate fault tolerant mechanisms
► Make a more in-depth study of the format used for alerts
► Incorporate other information about the environment
Thank you! Questions?
Preventing coordinated attacks via alert correlation

J. Garcia, F. Autrel, J. Borrell, Y. Bouzida

S. Castillo, F. Cuppens, G. Navarro

{jgarcia,jborrell,scastillo,gnavarro}@ccd.uab.es,

{fabien.autrel,yacine.bouzida,frederic.cuppens}@enst-bretagne.fr