TOPHAT: Topology based Host-Level Attribution for Multi-Stage Attacks in Enterprise Systems using Software Defined Networks

Subramaniyam Kannan*, Paul Wood*, Larry Deatrick**, Patricia Beane**, Somali Chaterji* and Saurabh Bagchi*

* Purdue University
** Northrop Grumman Corporation

Outline

• Background: Multi Stage Attacks
• Problem Statement: Attribution Problem
• Prior Work
• Our Solution
  ➢ Contribution
  ➢ System Model, Attacker Model
• Solution Variants:
  ➢ Uniform Risk Assignment Algorithm
  ➢ Low Risk Isolation Algorithm
• Results
• Conclusion and Future Work
Background: Multi Stage Attacks (MSA)

- Multi Stage Attacks utilize multiple victim machines in a series, to compromise a target machine deep inside the enterprise network.
- First, the attacker compromises an outward facing service by exploiting a vulnerability in it.
- Then uses the access privilege on that compromised service to proceed to rest of the network in a step by step manner until the crown jewel is compromised.

Problem Statement

Challenge:
- Protecting multi-layer distributed systems, such as those found in enterprise systems against Multi-Stage Attacks (MSA).

Issues:
- Attribution Problem - Difficult to attribute a particular network flow to the corresponding client or attacker at the entry server due to mixing of multiple network flows at the intermediate servers.
- Large Attack Surface - Attack can start at any entry point and propagate in different paths.
Attribution Problem Illustration: Server Setup

![Server Setup Diagram]

... Replication ...

Attribution Problem Illustration

![Attribution Diagram]

IDS Alert: Port Scan
(\text{SA} – \text{Web Server IP}, \text{DA} – \text{DB Server IP})

PHP Command Injection

IDS Alert Generation

Attribution/Un-Mixing/Traceability Problem – Who is the attacker?
Prior Work

**Attack Graph – Alert Inferencing [1]**

- Disadvantage:
  - Difficult to derive and to keep such information updated as systems are dynamic with new vulnerabilities being discovered and new users being added.

**Tagging [2]**

- Disadvantage:
  - Requires application level knowledge or deep packet inspection
  - Computation and space overhead of carrying the tag along

---


---

Our Solution - TOPHAT

- Intrusion Alert
- Guilty by Association
- Risk Tracking
- Moving Target Defense (Shuffling)
- Risk Management

shuffle the clients between the web servers

Attacker repeats the attack

Same alert is generated again
Our Contribution

• TOPHAT can attribute multi stage attacks on a distributed system to a single external source, without relying on attack graphs or modifying server software.
• The MTD-style defense significantly increases the attacker’s effort and can support identification of multiple attackers simultaneously.
• TOPHAT can support high availability for legitimate clients while identifying the attackers in the system.

System Model

• Each layer consists of multiple server instances or replica's for load balancing.
• Each layer is connected to the subsequent layers through open flow switches.
• SDN controller helps in dynamically routing the traffic to the desired server instances.
• Each server instance has an IDS installed.
• TOPHAT is installed in the SDN controller and is invoked whenever a strong alert is generated.
**Attacker Model**

- **Persistent Attacks** – The attacker repeats the attack until it proceeds to the next stage without an alert. Also, the predecessor stages in an attack must be repeated if the attacker is re-connected to a new server.
- **Strong Alerts** – The strong alert is an alert that with high certainty is known to be part of an attack (e.g., brute force attacks, known exploit signatures or other high priority* alerts). The attacker will generate at least one strong alert during a MSA.

*http://manual-snort-org.s3-website-us-east-1.amazonaws.com/node31.html#Snort_Default_Classifications - In Snort, rules are tagged with priority where “high” priority correlates with strong in our solution.

**TOPHAT Variant 1 - Uniform Assignment Algorithm**

- Each client is associated with a risk factor that indicates the likelihood of the client being an attacker.
- The clients are tracked by the source IP address.
- Initially, all the clients have a risk factor of 0.
- The clients are allocated to the available web servers using uniform distribution.
• When an alert is generated, uniform risk is assigned to all the clients present in the same server as the attacker.
• New Risk for each client = \( \frac{1}{NCA} \) (No. of clients allocated in the same server as the attacker).
• All the traffic is dropped by the open flow switch after the risk assignment.
• The clients including the attacker re-establish the connection.

• Once again, the clients are shuffled between the available web servers.
• Now the risk factor of those clients allocated to the same server as the attacker are incremented by \( \frac{1}{NCA} \).
• After sufficient no. of repeated attacks, the attacker’s risk will be maximum in the set without any possible ties.
• The attacker is isolated and his IP address is added to the blacklist.
Tophat Variant 2 - Low Risk Isolation Algorithm

- After each alert generation, 25% of lowest risk clients are moved to a safe server.
- The remaining clients are uniformly shuffled across the remaining processing servers.
- Each repeated attack reveals additional attacker information and increases the probability of the attacker to be identified.
- **Advantage:** The really well-behaved clients are not adversely affected at all.

Experiment Setup

- SDN environment is modeled using C++.

**Evaluation Parameters:**
- **Convergence Time:** Time at which single attacker or all attackers are identified.
- **Percentage of Failed Transactions (PFT(t)):** No. of client disruptions during attacker identification

\[ PFT(t) = \frac{\text{No. of Failed Transactions}}{\text{Total No. of Transactions}} \]
Result: Convergence Over Time

- Uniform converges more quickly in both cases than the LRA as it has 3 servers to use for risk attribution while LRA reserves a server for the safe pool.
- Uniform has a constant PFT as all the clients are disconnected and reassigned till the attacker is identified.
- LRA has a decaying PFT as only those clients remaining in active set are impacted for each attack.

Faster Convergence vs Slower Convergence with better client access.

Result: Attacker Effort

- Attacker effort is measured using the number of times a server must be compromised at any layer by any attacker.
- Total number of exploits increases non-linearly with respect to number of attackers as many attackers will be reset even if they don’t generate an alert themselves due to MTD.

Note: Rest of the results can be found in the paper.
Future Work

- Relaxing the assumption of strong alerts using more probabilistic risk assessment model that adapts the risk based on the quality of alerts.
- Optimize the number of IDS deployed to identify the attacker.
- Honey pot based discovery of the attacker after the isolation.

Conclusion

- TOPHAT solves the problem of attributing an alert to an attacker in a multi-layered system.
- TOPHAT utilizes moving target defense (shuffling) to isolate the attacker.
- TOPHAT provides two algorithms namely Uniform for faster convergence and LRA for improving client connectivity during the attacks.
- TOPHAT also increases the attacker’s effort by requiring multiple re-exploiting of the target system.