Forensic Profiling System

P. Kahai, M. Srinivasan, and K. Namuduri

Department of Electrical and Computer Engineering, College of Engineering

Abstract**:** *Incidents related to hacking and network intrusion are on the increase. Most organizations safeguard themselves against cyber attacks by employing security methods such as encryption technologies, network monitoring tools, deploying firewalls and intrusion detection and response mechanisms. Even though prevention mechanisms are in place the vulnerabilities associated with any computer network or security tool can be exploited by hackers to generate attacks. A major drawback in apprehending cyber criminals is lack of efficient attribution mechanisms. This paper proposes a forensic profiling system that accommodates real-time evidence collection as a network feature to address the difficulties involved in collecting evidence against cyber attackers.*

1. Introduction

Configuring security features does not guarantee the information systems absolutely foolproof. Evidence collection, *trace and trap* mechanism and identification of the attacker are as important as intrusion detection when a network attack takes place. Apprehending and prosecuting cyber criminals is complicated because of the intercontinental nature of the cyber space. Early intrusion detection systems (TRIPWIRE, SWATCH) were modeled to detect anomalous activities on a single host. In order to monitor the activities of the entire network, network-based IDS (Snort, e-Trust, NetSTAT and EMERALD (Event Monitoring Enabling Responses to Anomalous Live Disturbances) came into existence. Collaboration between the different intrusion detection and response systems has been the focus of recent research [1], [2]. In this paper we propose a mechanism for real-time forensic evidence where each network element is capable of detecting suspicious activity and provides evidence for the same in the form of log entries indicative of the malicious activity.

2. Proposed Forensic Model

Different nodes in the network are capable of contributing to the security of the entire network if security features are configured and enabled. The forensic profiling mechanism is based on client-server architecture where each node in the network, referred to as a forensic client, is capable of detecting an anomaly and warns a central server, the forensic server about the same in the form of an *alert.* All forensic clients participate in distributed intrusion detection and therefore maintain logs. The forensic client can be a router, a signature analyzer, an IDS, a firewall or a host in the network. The logical architecture for the forensic profiling system is shown in Fig. 1.

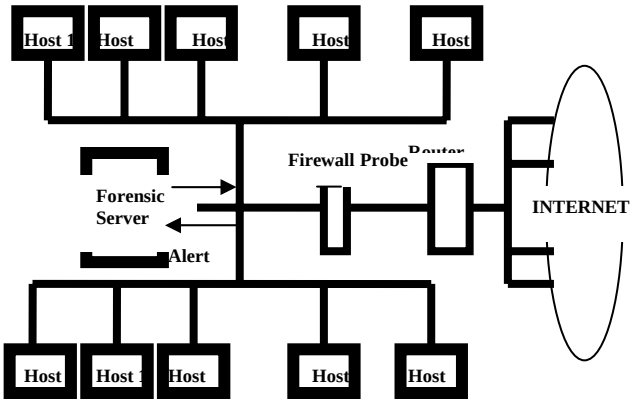

Fig.1 Logical Architecture for the Forensic Profiling System. The forensic clients are composed of hosts, servers, firewalls and the router in the network.

The backbone of the forensic profiling system is the *forensic profile*. A profile is a structure/descriptor that defines an attack in terms of the different events that should occur in order to qualify as a particular attack. The profile thus holds information about an attack in terms of the alerts associated with a particular attack. An *alert* is a message sent by the forensic client to the forensic server which is indicative of a suspicious activity in the client. An unusual event detected by the client triggers an alert in the format that contains the fields *When-Subject-Action-Object* along with the logs for that particular time stamp are also sent to the server. This functionality is achieved by installing an agent, agent alert, in each of the forensic clients which continuously looks for anomalies in terms of drifts in the performance parameters and at the same time scans for keywords in the log files that pertain to suspicious activities. The server is continuously listening for alerts and accordingly generates *probes* in order to gather more information about the received alert.

Probes are queries initiated by the server. When the server receives an alert it looks for a match in the forensic profile database and queries the other forensic clients depending upon the matches found. The forensic server demands information regarding the other alerts that are a subset of the matched profile in the form of a *probe*. The forensic server analyzes the response so received from the clients and if it concludes that an attack is in progress, the forensic server generates a request for sending the log entries

to the forensic clients and eventually an audit trail is embedded in the forensic profile. There is a latent time period when the entire network is behaving normally. During latency the forensic server maintains a database of *latent profiles* of all known attacks. A latent profile may become active when an alert is generated. Fig. 2 depicts the relationship between Alert X received from a forensic client with the forensic profile database that shortlists the active profiles. Also, Alert X is a subset of alerts associated with forensic profiles 2 and 3.

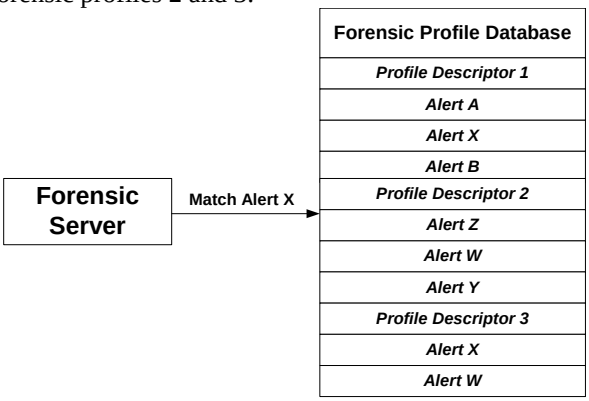

Fig.2 The forensic server scans for Alert X in the forensic profile database. The scan results in transforming Profile Descriptors 1 and 3 as active.

The forensic server builds the entire profile by making use of the alerts received from the forensic clients and the forensic profile database. The forensic server is capable of collecting real time evidence of the attack because of the alerts triggered at different stages of the attack which in turn enables the forensic server to store all the log entries pertaining to the attack.

3. Experimental Results

The experimental set-up constituted of two separate networks. A linux machine was configured as a router that comprised of two network interfaces 192.168.1.0 and 10.10.1.0. An attack was simulated by a host on 10.10.1.0 on to the 192.168.1.0 network. The forensic server and client modules were deployed on the machines that belonged to 192.168.1.0 network (the victim). In order to enhance the logging mechanism, a network-monitoring tool called *iplog* was used. The tool, *iplog* is a TCP/IP traffic logger capable of logging TCP, ICMP and UDP traffic. It also has the ability to detect port scans, null scans, FIN scans, smurf and bogus TCP flags. The log entries generated that eventually led to alerts to the forensic server for SYN-FLOOD attack are summarized in the next section.

3.I Denial of Service Attack (SYN-FLOOD)

The most common form of DOS attack is the SYN-FLOOD attack. The SYN-FLOOD attack works by creating numerous half-open connections. This occurs when the attacking system sends SYN packets to a server with different spoofed IP addresses. This results in a half-open connection as the server sends a SYN-ACK packet but never receives an ACK from the client. A pending connection is written to a buffer of limited size. As the attacking machine creates an ever-increasing number of pending connections, the buffer eventually overflows and the server is no longer able to handle any more connection requests. Thus the server is unable to serve the requests initiated by legitimate users. This leads to increased traffic intensity. If a DOS protection mechanism is in place, the server will suppress further requests and that event is logged in the system logs. The number of requests suppressed depends on the rate limiter. Also, connection attempts by large number of IP addresses (spoofed) increases the frequency of log entries. Thus, the DOS attack profile in its nascent state is as shown in Fig. 3.

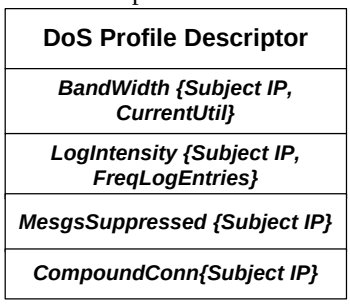

Fig. 3 The forensic profile for Denial of Service attack initiated by SYN Flood.

System logs that indicate DoS protection in progress:

Dec 8 03:25:26 localhost kernel: printk: 525 messages suppressed. .

Dec 8 03:25:31 localhost kernel: printk: 521 messages suppressed. .

Dec 8 03:25:36 localhost kernel: printk: 513 messages suppressed. .

***Iplog* entries that indicate compound connection attempts:**

Dec 8 03:25:52 TCP: http connection attempt to 192.168.1.1 from 139.142.70.51:1722

Dec 8 03:25:52 TCP: http connection attempt to 192.168.1.1 from 220.109.233.12:1047

Dec 8 03:25:52 TCP: http connection attempt to 192.168.1.1 from 95.163.14.119:1972

4. Conclusions

In this paper we have proposed a forensic profiling system for real-time forensic evidence collection. A dedicated server, the forensic server is capable of maintaining an audit trail embedded in the forensic profile. The FPS would help in reducing the time spent on forensic investigation after the attack has occurred.

5. References

[1] C. Kahn, D. Bolinger, D. Schnackenberg, “Common Intrusion Detection Framework”, http://www.isi.edu/gost/cidf/, 1998.

[2] F. Cuppens, A. Mi`ege, “Alert Correlation in a Cooperative Intrusion Detection Framework”, in Proceedings of the 2002 IEEE Symposium on Security and Privacy, May 2002.