SPACEDIVE: A Distributed Intrusion Detection System for Voice-over-IP Environments

Vinita Apte, Yu-Sung Wu, Saurabh Bagchi
Dependable Computing Systems Lab
School of Electrical & Computer Eng
Purdue University
Email: {vapte,yswu,sbagchi}@purdue.edu

Sachin Garg, Navjot Singh
Avaya Labs

Email:{sgarg,singh}@avaya.com

1 Introduction

Voice over IP (VoIP) systems are gaining in popularity as the technology for transmitting voice traffic over IP networks. Along with the anticipated widespread adoption of VoIP systems comes the possibility of security attacks targeted against such systems. The attacks can be thought of as a combination of traditional kinds of security attacks against IP networks and novel attacks enabled by the architecture of VoIP systems.

VoIP applications have soft real time requirements. Second, the attacks can span multiple protocols between different end points and may be spread over arbitrary time periods. Considering a range of attack scenarios seen in practice, we observe that the attack symptom is often detectable only by correlating information from multiple sources. The correlation is required among information from multiple protocols at multiple end points. The correlation may need to be done from sources that are peers, such as, two communicating clients or across peer levels, such as, the communicating clients and the servers.

We propose the design of a system called SPACEDIVE to serve as a correlation-based IDS for VoIP systems. The Snort IDS [2] is well known for its efficiency in examining incoming packets and SPACEDIVE leverages the Snort functionality. To achieve good performance, SPACEDIVE is built *into* Snort using part of its low-level functionality (examining and processing packets) and adding to it (e.g., to build state to support stateful detection) and building completely the high level functionality specific to the VoIP environment.

The contributions of the paper and the advantages of SPACEDIVE can be specified as follows:

1. SPACEDIVE presents the architecture of a hierarchical correlation based IDS that is well suited to detecting attacks in VoIP applications. The ability to match rules remotely makes the system less prone to DoS attacks launched against VoIP components or their hosts.
2. SPACEDIVE provides a language to specify rules for local matching and remote matching. SPACEDIVE's architecture makes the rule matching, efficient and scalable, both essential features for a VoIP system since it has soft real-time requirements.

2 SPACEDIVE Design

2.1 SPACEDIVE Design Hierarchy

The SPACEDIVE design can be decomposed into two parts – the local-level design and the network-level design. Local-level design involves a single VoIP component (client, proxy, etc.) and has the local Rule Matching Engine (RME_L). Network–level design takes into consideration all the components deployed in one domain or across multiple domains and the interactions between them and provides the remote Rule Matching Engine (RME_R)

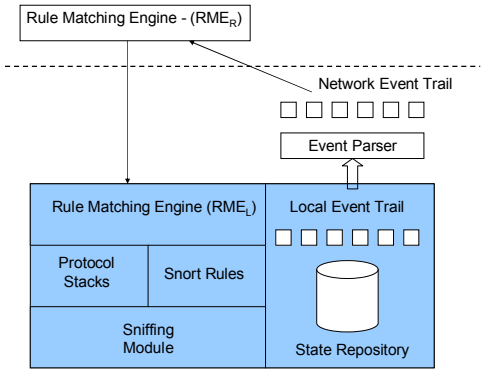

Figure 1: Local-level SPACEDIVE design (Components below the dotted line are local)

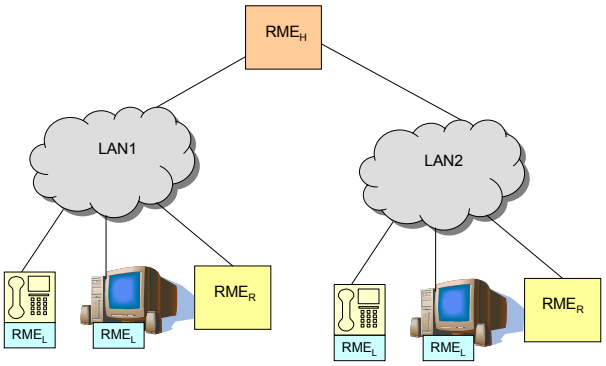

Figure 2: Rule Matching Engine Hierarchy.

2.2 SPACEDIVE Local Level Design

Figure 1 shows the SPACEDIVE components at the local level – the RME_L's. The components below the dashed line represent an instance of SPACEDIVE installed on each VoIP component and integrated with Snort. The *sniffing module* makes use of the `libpcap` library to read packets received over the network.

The State Repository stores the current state of the system. State comprises the status of an ongoing session – i.e. connecting, established, terminated, etc., the status of a node, e.g., if the node has moved, or the reception of a particular type of packet (e.g. a SIP BYE message). The *Event Trail* keeps track of events, specified using the low level rule language. The event trail contains events ordered by session ID. The *Processing Engine* determines whether a pre-defined

event has occurred and records it in the event trail. It also updates the state of the rule variables in the *State Repository*. The *Event Parser* takes the event trail as input and generates a trail of "Network Events". What constitutes a Network Event is specified in the *RME$_R$*, which disseminates the pertinent network event definitions to the RME$_L$'s. The RME$_R$ uses the Network Event Trail to correlate events across the different components of the network.

2.3 SPACEDIVE Network Level Design

At the network level, SPACEDIVE views the system as composed of multiple VoIP domains, each with its own RME$_R$ (Figure 2). The RME$_R$'s perform remote rule matching from network events generated by each RME$_L$ in its domain. We have developed a high-level rule language for specifying network level events in the RMEs.

2.4 Low level rule language

The native rule language of Snort is not well-suited for stateful or cross-protocol detection, described in [1]. Snort provides very limited capability for remembering state both within a session for a given protocol and across protocols. To make up for this, we add constructs to the existing rule language so that it is better-suited for detecting attacks targeted to VoIP environments that span packets in a session and different protocols. A brief description of the new constructs follows-

(a) **var**. This construct is used to set the integer value of a variable in case of a rule match. This is used as a way of keeping state. The `var` construct belongs to the 'options' part of a Snort rule.

(b) **Event.** The event construct is used to create event trails. It tells Snort to record an event when the corresponding rule-match occurs. An event can be triggered on a combination of rule matches according to the following constructs.

(c) And/Or/Not – Logical Constructs. These constructs are used to trigger an event based on logical combinations of rule matches;

(d) **Before/After – Temporal Constructs.** The Before and After constructs are used to trigger events based on a temporal sequence of rule matches.

(e) **Net_Event.** This construct follows the same syntax as 'Event' except that it is used to represent a network event as opposed to a local event.

(f) **Protocol-specific constructs.** To detect certain attacks we need to look into specific fields in the header of a protocol.

3 Demonstration and Results

To realistically emulate a VoIP environment, we have built a testbed with two domains. This enables us to demonstrate intra-domain calls as well as inter-domain calls. Each domain has a SIP gateway, a proxy server, a registrar server, clients and support servers like FTP, DNS, etc. The SIP clients and servers are equipped with the SPACEDIVE IDS. We use the SIP Express Router (ser) [3] for the SIP servers. Ser can be configured as a SIP registrar or proxy server. Our SIP clients are Windows based and use X-Lite [4]. In the testbed, we have the gateway, registrar and proxy server running on the same machine. We deployed RME$_H$ in domain 1 though in practice it can belong to either domain or be in a separate domain altogether.

Next we demonstrate the detection of the Man-in-the Middle attack using SPACEDIVE.

Man in the middle attack : intercepting outgoing calls

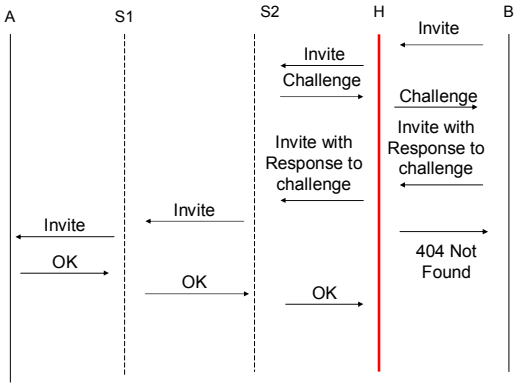

Figure 3: Man in the middle attack

In this attack, we assume H is on the route between S2 and B. The goal for H is to intercept outgoing calls from B. The INVITE messages are authenticated through a challenge-response mechanism. As B places an outgoing call, the attacker H forwards the INVITE messages and the challenge-responses between S2 and B until the authentication phase is completed. Then H fakes a '404 Not Found' message back to B such that B thinks A is not present. In effect a call is established between H and A with H representing itself as B.

This attack can be detected by SPACEDIVE with an end-to-end matching rule for the OK message going correctly all the way from A to B through S1 and S2. The end-to-end matching rule where the passage of a message through several VoIP components is tracked, is an important class of rules supported by SPACEDIVE.

Figure 4 shows the results of the detection of this attack.

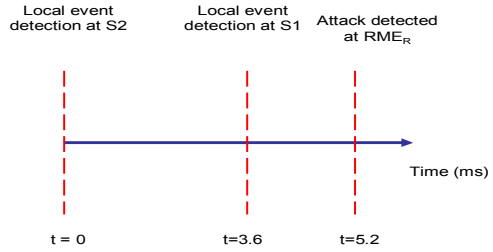

Figure 4: Timeline for Remote Rule Matching (times are not drawn to scale)

References:

[1] Y. Wu, S. Bagchi, S. Garg, N. Singh, and T. Tsai, "SCIDIVE: A Stateful and Cross Protocol Intrusion Detection Architecture for Voice-over-IP Environments." In 2004 International Conference on Dependable Systems and Networks (DSN'04)

[2] The Snort Intrusion Detection System, www.snort.org

[3] SIP Express Router (ser), http://www.iptel.org/ser/

[4] X-Lite, http://xten.com/index.php?menu=X-Series