SPECIFICATION-BASED INTRUSION DETECTION SYSTEM FOR CARRIER ETHERNET

Pan Jieke, João Redol
Siemens Networks S.A, Lisboa, Portugal
pan.jieke@siemens.com, joao.redol@siemens.com

Miguel Correia
Faculty of Sciences, University of Lisboa, Lisboa, Portugal
mpc@di.fc.ul.pt

Keywords: Specification-based Intrusion Detection, Ethernet, Spanning Tree Protocol, Network Topology, Security.

Abstract: Layered network architectures (OSI, TCP/IP) separate functionality in layers, allowing them to be designed and implemented independently. However, from the security point of view, once a lower layer is compromised, the reliability of the higher layers can be impaired. This paper is about the security of the Data Link Layer, which can affect the reliability of higher layers, like TCP, HTTP and other World-Wide Web protocols. The paper analyzes security-wise a layer 2 protocol – the Spanning Tree Protocol (STP), part of the Ethernet suite – and presents a solution to detect attacks against this protocol using Specification-based Intrusion Detection.

1 INTRODUCTION

The *Internet* and the *World-Wide Web* play an increasingly indispensable role in the modern society. The dependence on networked computers is a phenomenon that can be observed in several areas. This trend could be observed during the last decade and will certainly continue in the future. Nevertheless, due to its popularity, the Internet has become the target of malicious hackers, which cause intrusions and disseminate malware at a world-wide scale.

Layered network architectures, like the OSI and TCP/IP models, separate functionality in layers, where the lower layers provide services for the higher layers. These flexible models provide a form of separation of concerns, which allows the layers to be designed and implemented independently. However, from the security point of view, once a lower layer is compromised, the reliability of the higher layers can also be affected. Most existing network security paradigms and models are concerned with the layers from 3 to 7 of the OSI model, i.e., from the Network to the Application layers. In Internet terms, this means security is mostly concerned with protocols like IP, TCP, HTTP and SOAP, and issues like user authentication, data integrity and confidentiality. Less attention has been paid to the *network infrastructure*, i.e., to layer 1 and 2 protocols. However, attacks against these two layers can "disable" the network, causing unavailability of the higher layer protocols, with a huge impact on a large-scale network, possibly involving innumerable service providers and users.

The work presented in this paper has the purpose of protecting the *Carrier Ethernet* network infrastructure using a *distributed intrusion detection and response system*, since this kind of network technology is being adopted around the world by Internet service providers and other companies[1]. More specifically, the work deals with the *Network Elements – NEs –* Switches/Bridges – and the Data Link Layer (layer 2) protocols. Currently, NEs are reasonably complex hardware and software boxes, which therefore can have vulnerabilities, like those often reported in web servers, browsers, operating systems, etc. So, we consider that a subset of the NEs of a network can be compromised by hackers and launch attacks against the network infrastructure, i.e., they can be malicious. Previous protection schemes for STP did not consider this possibility of NEs being malicious; they essentially discard STP messages (BPDUs) (IEEE, 1998) sent by non-NEs trying to emulate a NE (Marro, 2003; Cisco, 2005a; Cisco, 2005b).

To deal with this problem, we propose that NEs

[1]The adoption of Carrier Ethernet is being pushed by the Metro Ethernet Forum. The site of the forum is at: http://www.metroethernetforum.org

are equipped with a component which provides *network intrusion detection*. Each of these detectors inspects the behavior of other NEs by inspecting the messages received from them. Detection follows a recent approach dubbed *specification-based intrusion detection*, which relies on a specification of the protocol to detect deviations from it (Balepin et al., 2003; Sekar et al., 2002; Uppuluri and Sekar, 2001).

The contribution of this paper is the design of the first network intrusion detection scheme for carrier Ethernets. It focuses on the original link management protocol of the (switched) *Ethernet*, the *Spanning Tree Protocol (STP)*, as a case study (IEEE, 1998). Detection is based on a specification-based intrusion detection scheme, which we extend with annotations, to detect attacks with certain time patterns.

2 THE SYSTEM

The paper proposes a distributed intrusion detection system for *Carrier Ethernet* (Figure 1). Typically, *Carrier* class NEs are used for providing connectivity from end users to ISPs or other service providers. The proposal involves extending each NE with a network intrusion detection component. This extension can be done in two ways:

- by inserting the component in the NE, in case its internal resources (CPU, memory) allow it; or
- by connecting the component to the NE using a network port, and forwarding all BPDUs received by the NE to the component.

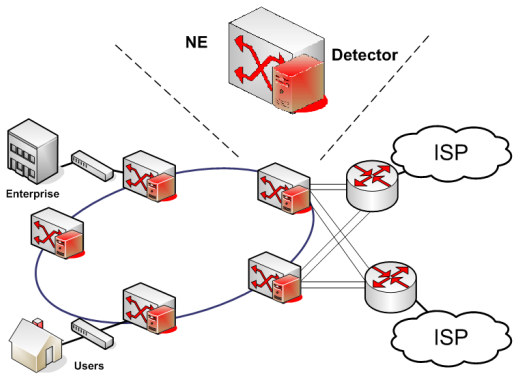

Figure 1: System architecture

We assume a subset of the NEs of a network can be compromised by hackers and launch attacks against the network infrastructure. We say those NEs are *malicious*, while the rest are *correct*. To make the assumption even stronger, if a NE is malicious, its network intrusion detection component will be compromised too. Therefore, a subset of the NEs may not follow the correct specification of STP, and this incorrect behavior has to be detected by the correct NEs/detectors. Previous protection schemes for STP essentially discarded BPDUs sent by non-NEs trying to emulate a NE (Marro, 2003; Cisco, 2005a; Cisco, 2005b).

2.1 STP Attacks

As one of the most commonly used *Ethernet* standard configuration protocols, the *Spanning Tree Protocol (STP)* is an obvious target for attacks, which can compromise the security of all the layers above layer 2 and the network availability. STP is a low level network link management protocol that provides path redundancy while preventing undesirable loops in the network. STP exchanges messages called *Bridge Protocol Data Units (BPDUs)*, which contain the Bridge ID (bridge's unique identifier) of the sender. BPDUs are used to learn about the existence of other bridges and to obtain the information needed to calculate and maintain the spanning tree. A few characteristics render the STP protocol vulnerable to several types of attacks from hackers who have direct physical access to the network equipment. These attacks use the legitimate connection with network equipment to inject attacks in the network. Recall that we say that a *NE is malicious* if it is controlled by a hacker, thus can execute arbitrary attacks against the STP protocol.

A careful analysis of the protocol and the literature (Marro, 2003), lead us to conclude that the attacks possible against STP are essentially the follow:

- ID changing attacks: A NE modifies its bridge identifier to cause (undesirable) topology changes (since the root of the tree is the node with lowest bridge ID).
- Silent attacks: When the network topology is stable every NE continues sending Configuration BPDUs to others to indicate it is alive. A malicious NE can omit sending these BPDUs to cause an undesirable network reconfiguration. When executed repetitively, this attack can impair the availability of the network.
- Faked failure attacks: A NE generates fake Topology Change Notification BPDUs to cause network reconfigurations.
- BPDU flooding attacks: A flood of bogus BPDUs from different addresses is injected in the network, possibly causing undesirable network reconfigurations.
- Invalid BPDU: Sending malformed BPDU, e.g., as an attempt to find a vulnerability in the target NE.

2.2 STP Specification

This paper uses *specification-based intrusion detection* (Balepin et al., 2003; Sekar et al., 2002; Uppuluri and Sekar, 2001) which detect attacks as deviations from a norm. By specifying correct behavior, any other behaviors will be classified as anomalous.

The detector performs network intrusion detection in run-time. The specification of the STP protocol is modeled using a *state machine*. The states of this machine are the states of the protocol, and state transitions are caused by the reception of BPDUs or expiration of timeouts.

Instead of crafting the detailed description of the protocol, an abstract specification of STP is used. Developing a more precise specification requires more effort and has been considered to have a negative impact in specification-based intrusion detection (Sekar et al., 2002). The reason is that real implementations of a protocol tend to have differences between themselves and to deviate from the specification on the standard or RFC. A more abstract specification tends to omit the details that lead to these inconsistencies.

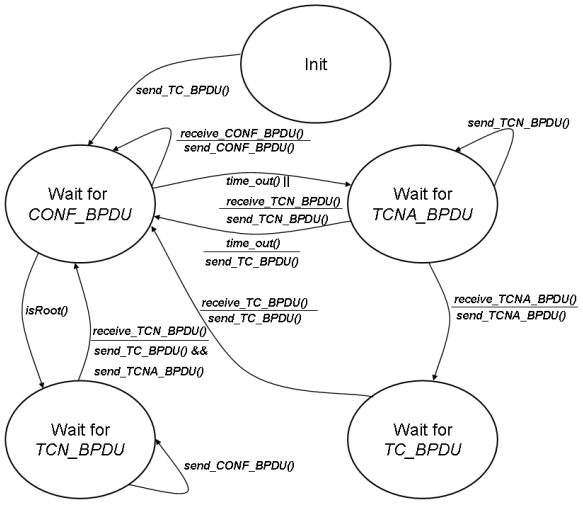

Figure 2: Spanning Tree Protocol state machine

Figure 2 presents the state machine that specifies STP. Each node in the diagram represents a state in which a NE can be at a certain instant. Each arrow represents a transition between two states. Arrows have a label in the form of a fraction: the top part is the event that causes the transition (message reception, timeout), while the bottom is the event generated (message sent).

Annotations The state machine in Figure 2 is an incomplete specification of the STP protocol since it allows behavior that is not acceptable. Examples include a NE omitting sending Configuration BPDUs repeatedly although it is alive, and sending many bogus Topology Change BPDUs.

Distinguishing correct behavior from these deviations requires extending the state machine with *annotations*. Each event *timeout* or *send BPDU* can be annotated with the acceptable time pattern of that event. For simplicity we consider that, given an event e, this pattern is defined simply in terms of the maximum number of repetitions of e per unit of time, $Rmax_e$, but more complex forms of specification might be used. These values can be defined for any event but only some of them are relevant to detect the attacks presented in Section 2.1. Examples are $Rmax_{timeout}$ – for the timeout in the transition between the state Wait_for_CONF_BPDU and the state Wait_TCNA_BPDU – and $Rmax_{tcn}$ – for the send_TCN_BPDU events in state Wait_for_TCNA_BPDU. These values have to be carefully defined in order to avoid detecting false attacks (false positives) and missing attacks (false negatives).

2.3 Intrusion Detection

Consider again the state machine in Figure 2. Each NE stores a representation of the state machine of its neighbors (i.e., of the NEs who are directly connected to it) and has a variable that says which is the current state of a neighbor. The state machine for every NE starts at the *Init* state. The detection algorithm executed by a pair *NE/detector* is the following:

1. When the NE receives a BPDU message: pass it to the detector;
2. When the detector gets the BPDU: check if it is an *expected* BPDU message in the current state and if satisfies the pattern defined for it (i.e., its $Rmax_e$);
3. If the message is not expected, then suspect of the NE that has sent it.

An attack is detected firstly by matching the received message with expected messages in the current state. For example, if a NE is in the Wait_for_CONF_BPDU state, it can not do a transition to the *Init* state directly (see Figure 2). In other words, if a NE is waiting for a configuration BPDU in a stable topology, it is incorrect to send a Topology Change BPDU. Therefore, if the NE in the Wait_for_CONF_BPDU sends a Topology Change BPDU to the others, this will be detected to be an anomaly. The same reasoning can be done to the other states and to the patterns.

Let us now make an argument that the intrusion detection scheme presented above detects the attacks

against STP presented in Section 2.1. All detection actions are made by the detectors in the NEs, but we simplify the presentation by forgetting this separation between a NE and its detector.

- *ID changing attacks.* Recall that each NE stores a representation of the state machine of its neighbors. This representations includes the Bridge ID of the neighbors. If the NE receives two or more BPDUs with different sender IDs from the same port, it detects that its neighbor is malicious (if the ID is modified by an administrator, other NEs should be informed).
- *Silent attacks.* For this type of attacks, when a malicious NE stays silent on purpose, other NEs will generate a *timeout* event, consider it as a legitim NE failure and send Topology Change Notification BPDUs (see Figure 2). So, the specification-based intrusion detection scheme does not detect it to be an attack. One solution would be the elimination of the *timeout* event from the specification, so a timeout would always be suspected of being an attack, but it would generate suspicions for every (non-malicious) failure. The solution is not to detect if the NE simulates a failure but if it does it too many times. This detection is done using the pattern of the event timeout between the states Wait_for_CONF_BPDU and Wait_for_-TCNA_BPDU, i.e., if that timeout expires more than $Rmax_{timeout}$ times in a unit of time then the NE is malicious.
- *Faked failure attacks.* This attack is done by sending a Topology Change Notification BPDU repeatedly, so it is detected if that BPDU is not sent in the state Wait_for_TCNA_BPDU or if it is sent more than $Rmax_{tcn}$ times in a unit of time.
- *BPDU flooding attacks.* These attacks are detected when events that send the BPDUs are done more than the corresponding $Rmax_e$ times in a unit of time.
- *Invalid BPDU.* This attack is directly detected by the specification-based intrusion detection scheme, since an invalid BPDU is precisely one that can not be sent in the current state of the NE.

When a NE is identified as being malicious, it will be logically removed from the network, i.e., its neighbor NEs will disconnect all ports connected to it. Naturally this is only possible if the network has enough redundancy to disconnect an NE and still keep the network connected. Moreover, it requires that the NEs *correlate* their detections (Kruegel et al., 2005), in order to make agreement about which NE(s) is malicious. This correlation and disconnection scheme is still being investigated.

3 CONCLUSION

The *Internet* and the *World-Wide Web* have become more and more important in modern society. However, they also became the target of a legion of malicious hackers. Compromising a low layer, like the Data Link Layer, can affect the reliability of higher layers, like TCP/IP, HTTP, SOAP and other web protocols. This work gives a novel perspective about a security solution for low-level network protocols, based on a *specification-based intrusion detection system* enhanced with annotations. More precisely, we presented a solution for protecting *Carrier Ethernet* by detecting attacks against the STP protocol. We argued that protecting these protocols is crucial to ensure the availability of the network so higher level protocols, like HTTP and other Web-related protocols can be executed normally.

ACKNOWLEDGMENTS

The work presented in the paper is funded by *Siemens Networks, S.A. Portugal*.

REFERENCES

Balepin, I., Maltsev, S., Rowe, J., and Levitt, K. N. (2003). Using specification-based intrusion detection for automated response. In *Proceedings of the International Symposium on Recent Advances in Intrusion Detection*, pages 136–154.

Cisco (2005a). *Spanning Tree PortFast BPDU Guard Enhancement*. Cisco Systems Inc. Document ID 10586.

Cisco (2005b). *Spanning Tree Protocol Root Guard Enhancement*. Cisco Systems Inc. Document ID 10588.

IEEE (1998). *ANSI/IEEE 802.1D-2004 standard - Part 3: Media Access Control (MAC) Bridges*.

Kruegel, C., Valeur, F., and Vigna, G. (2005). *Intrusion Detection and Correlation: Challenges and Solutions*, volume 14 of *Advances in Information Security*. Springer-Verlag.

Marro, G. M. (2003). Attacks at the data link layer. Master's thesis, University of California.

Sekar, R., Gupta, A., Frullo, J., Shanbhag, T., Tiwari, A., Yang, H., and Zhou, S. (2002). Specification-based anomaly detection: a new approach for detecting network intrusions. In *Proceedings of the 9th ACM conference on Computer and communications security*, pages 265–274.

Uppuluri, P. and Sekar, R. (2001). Experiences with specification-based intrusion detection. *Lecture Notes in Computer Science*, 2212:172–189.