

Network Security: An Open-Source Approach

Blain R. Jones

Abstract— This paper attempts to introduce the reader to the Network Security Monitoring (NSM) model and the availability of robust open-source network security utilities to achieve a defensible network. To achieve a secure network, the analyst must fully implement all aspects of the Security Life Cycle. The Security Life Cycle is a process for maintaining an acceptable level of perceived risk of network security. Detection is one component of the process. Open source tools that can be used in the detection phase of the security life cycle include Sguil, MySQL, Snort, Barnyard, Tcpflow, Sncp, Pof, and Ethereal/Tethereal.[1]

Index Terms—

Intrusion Detection System, network security, Open source, Network Security Monitoring, Security Life Cycle

I. INTRODUCTION

WHAT is network security monitoring (NSM)? NSM involves collecting, analyzing, and increasing indications and warnings to detect and respond to intrusions. This operational model was inspired by the United States Air Force's signals intelligence (SIGINT) collection methods and Todd Heberlein's "Network Security Monitor" [2]. SIGINT is a process of collecting information on communication and transforming this information into intelligence [3].

This may sound a lot like an Intrusion Detection System (IDS), but IDS differs in that it does not have any signature to detect against structured attacks that employ stealth, encryption, zero-day exploits. In contrast NSM can collect all network traffic data related to an alert, this allows network security analysis to drilling down through the data to analyze and validate the indicator or warning.[3]

Indications are observable actions which define enemy capabilities and intentions. Within the context of NSM indicators are outputs from products which are created by IDS and are usually referred to as alerts. Trained people who may be referred to as analysts should be engaged in interpreting intrusions. The interpretation of indicators results in warnings. Warnings are human conclusions which indicate to decision makers that a network may have been compromised [4].

Intrusion Detection Systems (IDS) function as a secondary system designed to backup systems such as firewalls, encryption, and authentication. A secondary system is required in case a primary system's rule set is not updated correctly [5].

One can see then that NSM is more than just the IDS model. Many confuse NSM with IDS; however, NSM includes a

variety of functions which can be described as: (1) A product such as Snort performs the collection function. (2) People perform the analysis. (3) Processes guide the escalation. Escalation involves informing decision makers about network intruders.

NSM is just one part of the process of securing a network. By combining NSM and perimeter protection, e.g., firewall, it's possible to achieve a complete security lifecycle. NSM can be used for the assessment, detection, and response aspect of the security lifecycle, and protection can be achieved through perimeter protection devices.

II. SECURITY LIFECYCLE

A process for maintaining an acceptable level of perceived risk is security. It is not an end state, but rather a process. There are four steps in a security lifecycle: assessment, protection, detection, and response.

A. Assessment

Assessment pertains to policies, procedures, laws, regulations, budgeting, and other functions that are preparation for the other three dimensions of the process.

B. Protection

Although protection is the desired end result, in this context protection refers to the process of implementing countermeasures to reduce the likelihood of a compromised network.

C. Detection

Detection is the process of identifying policy violators who are unauthorized to use or perform unacceptable actions on a computer network. This subject is the focus of this paper

D. Response

Response is acting on the information gathered during the detection phase. The activities that occur during the response phase include maintenance on the countermeasures implemented during the protection phases and prosecution of intruders. [4]

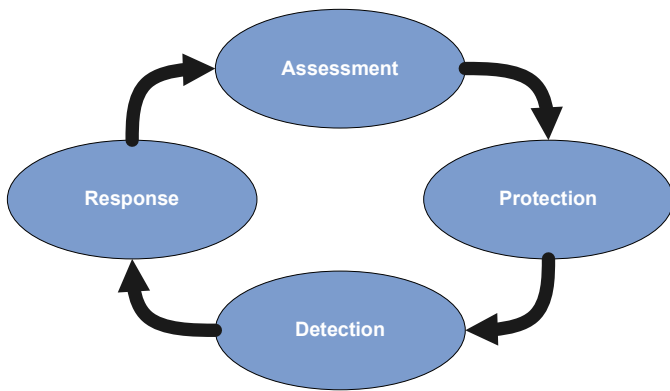


Figure 1 Security Lifecycle

Although it is challenging to defend a network from intruders, it is important to realize that there are open-source tools that can be used in the detection phase of the security lifecycle

III. OPEN-SOURCE

The UNIX philosophy for building tools/utilities is one of cooperation. According to Doug McIlroy “This is the UNIX philosophy: Write programs that do one thing and do it well. Write programs to work together. Write programs to handle text streams, because that is the universal interface.”[6] This is what Sguil (pronounced sgweel) does for NSM; it allows the integration of several cross-platform tools including Tcl/Tk, MySQL, Snort, Barnyard, SANCP, Tcpflow, p0f, and tcpdump. [4]

A. Sguil (<http://sguil.sourceforge.net/>)

There are three main components of Sguil: a plugin to barnyard (op_sguil), a GUI server (sguild), and a GUI client (sguil.tk). Sguil is completely written in Tcl/Tk. Tool Command Language (Tcl) is an interpreted programming language well suited for rapid application development. Tk is the graphic tool kit that generates the Sguil interface (Figure 2) on the analyst’s screen. This analysis interface is what makes Sguil valuable to the end user. Sguil displays to the end user network packet information from a variety of open source utilities in real time. In fact, it has a tab called “real time events.” By default the screen is broken down into three windows with the top window showing the most severe alerts, the middle the less serious alerts, and the bottom the least serious alerts. In the bottom half of the display, the window is broken vertically into two halves. The left side of the display provides host name and Whois database information. The right side of this window provides detailed information for the highlighted alert in the real time event tab. If alerts warrant further investigation, the alert can be moved to the “Escalated Events” tab where the information may be further investigated. A diagram defining how the different Sguil components interact with each other is indicated in figure 3.

ST	CNT	Sensor	sid.cid	Date/Time	Src IP	SPort	Dst IP	DPort	Pr	Event Message
RT	2	reset	1.47783	2005-06-09 13:40:32	61.184.47.150	4784	10.1.1.102	80	6	WEB-UIS ISAPI .ida attempt
RT	2	reset	1.47784	2005-06-09 13:40:32	61.184.47.150	4784	10.1.1.102	80	6	WEB-UIS ISAPI .ida access
RT	1	reset	1.47785	2005-06-09 13:40:32	61.184.47.150	4794	10.1.1.102	80	6	http_inspect_UFENCODING
RT	1	reset	1.47788	2005-06-09 13:40:32	61.184.47.150	4794	10.1.1.102	80	6	http_inspect_APACHEWHITESPA

Figure 2 Sguil Interface

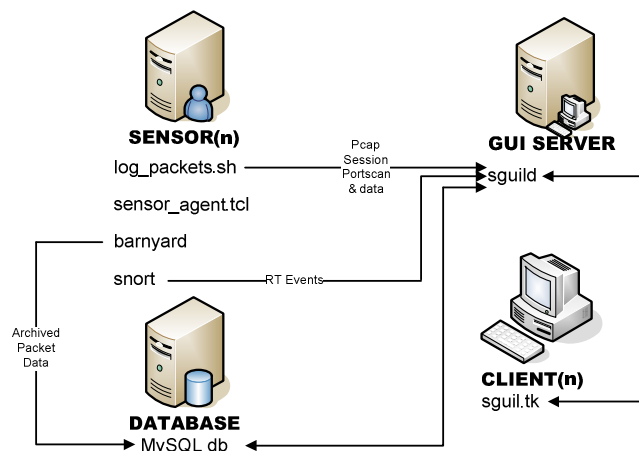


Figure 3 Sguil Component interactions

The analyst can use these components to view Snort events in near real time. There are seven incident categories for classifying Snort events or events may be marked for no further future action required (NA). To retrieve archived events the analyst can create a custom query using SQL or through preformatted queries.[4]

Also, the Sguil package provides a patch for the portscan preprocessor (logs to a pipe delimited file), a patch for the stream4 preprocessor (keepstas type ‘db’ for logging pipe delimited stats to a file), and a tcl script (sensor agent.tcl) for loading the modified outputs into the database. Using these components the analyst may gain immediate access to portscan and session data.[4]

Final components are used for the analysis of the raw data associated with a specific session. Xscriptd is a daemon that monitors requests from Sguil client program. Once queried, it converts raw tcpdump files for packets matching the requested session. Then, the stream is either fed through Tcpflow to create a transcript or the binary data is sent back to the Sguil client to be loaded into Ethereal for further analysis[7].

B. MySQL (<http://www.mysql.com/>)

MySQL is used to store the alerts and packet data gathered from Snort. Using MySQL as the database gives the analyst

the ability to interface with the network data in three different ways. First, Sguil provides an integrated database query tool. To access the query tool, open up the "Query" menu from the main menu bar, then select "Query Builder". At this point it's possible to query the event, sessions or sancp table. Secondly, click the GUI elements to build a valid query for a particular table or simply type the query into the text area. By running queries from the Query Builder, they will show up in the Sguil console, and it will be possible to right-click on the rows to perform all the common Sguil actions. Third, a well versed user of MySQL client may use it to access the information desired. Due to the nature of Sguil being an open-source project, information on the contents of each table can be found in Sguil's documentation.

C. Snort (<http://www.snort.org/>)

Snort is one of the most well-known open-source network-based intrusion detection systems [8]. It also has the capability to collect and view packets. Snort provides the alert data to the Sguil database. There are some modifications to Snort to accommodate Sguil's need for alert and packet data [4, 9]

D. Barnyard (<http://www.snort.org/dl/barnyard/>)

Barnyard is a plug-in for Snort that enables Snort's data to be written out to either a MySQL or Postgres database. In the case of Sguil the events from the Snort log file are taken by Barnyard and inserted into a MySQL database running on the Sguil server. This all happens in near real time.

E. Tcpflow

(<http://www.circlemud.org/~jelson/software/tcpflow/>)

A program that captures data transmitted as part of TCP connections (flow) is Tcpflow. Tcpflow stores captured data using a method that is conducive to protocol analysis or debugging. In contrast, a program like Tcpcmdump provides a summary of packets seen on the wire, but usually does not store the transmitted data. Tcpflow reconstructs the actual data streams and stores each flow in an individual file for future analysis. Due to Tcpflow's ability to understand TCP sequence numbers, it will correctly reconstruct data streams regardless of retransmissions or out-of-sequence delivery [10].

Tcpflow is utilized by Sguil to create transcripts from full content data stored by Snort. It is best to allow Tcpflow to run in its native mode when the application data is that of a file transfer data channel. When sessions are rebuilt, one of the files will contain the binary downloaded by the client [4].

F. Sancp (<http://www.metre.net/sancp.html>)

Sancp is a network security tool which collects statistical information regarding network traffic and the traffic itself in pcap format. The purpose of these functions is auditing, historical analysis, and network activity discovery. Through the use of rules the analyst may distinguish normal from abnormal traffic and support tagging connections with: rule id, node id, and status id. In the context of intrusion detection,

every connection is an event that must be validated through some means. With Sancp rules are used to identify, record, and tag traffic of interest. A new feature since v1.4.0 Connections ('stats') is "tagging" a connection to be loaded into a database for further analysis.[11]

Three types of logging for a connection are controlled using Sancp rules: pcap, stats, and realtime. Pcap refers to packet data captured on the connection in tcpcmdump format, stats refers to a single line summary of an full connection once it is closed, and realtime is a snapshot of stats based on the initial packet, for immediate reporting Stats and realtime contain fields which may be utilized for recording packet statistics, TCP flags, p0f data, and other vitals about how to handle the connection.[11]

G. P0f (<http://lcamtuf.coredump.cx/p0f.shtml>)

A versatile passive operation system identification tool is P0f. P0f attempts to identify the operating system of the hosts it encounters using one of these tests: (1) By default it uses a SYN packet test. P0f does this by watching inbound SYN packets that are entering a local network. (2) P0f with the -A option enables the SYN+ACK test where P0f makes decisions based on SYN ACK packets. Responses are from open remote ports. (3) P0f with the -R option activates the RST+ACK test where P0f makes decisions based on RST ACK packets. Responses are from closed remote ports.

Also, P0f can perform many other activities, and it can detect or measure the following: firewall presence, NAT use (useful for policy enforcement), existence of a load balancer setup, the distance to the remote system and its uptime, intruder's network hookup (DSL, OC3, avian carriers) and the intruder's ISP [12].

P0f provides enhanced detail on traffic from remote hosts for the Sguil project. This is especially useful when the operating system finger printing tool needs to be hidden from the intruder. Remote hosts will not be aware that they are being fingerprinted, because P0f does not emit packets to make its identification decisions.

H. Ethereal/Tetheral (<http://www.ethereal.com/>)

Ethereal is a GUI packet analyzer which seeks to capture network packets and attempts to display that packet's data and include as many details as possible. A network packet analyzer works like a measuring device used to examine what is going on inside a network cable. Similarly yet on a more simple level, a voltmeter is used by an electrician to examine what is happening inside an electric cable[13].

In the past, tools such as Ethereal were either very expensive, proprietary, or both. However, Ethereal has changed all that. Ethereal is perhaps one of the best open source packet analyzers available today.[13]

Tetheral is a command-line twin of Ethereal. It is most useful when Ethereal is not available such as when examining large data captured on a remote sensor in a command line environment. Tetheral's broad range of protocol decoding options simplifies the understanding of certain protocols and supports a wide range of collection-friendly options. The

same holds true of Ethereal.[13]

IV. CONCLUSION

This paper provides an overview of the Network Security Monitoring (NSM) which involves collecting, analyzing, and increasing indications and warnings to detect and respond to intrusions. NSM is just one dimension of the Security Life Cycle. The reader should keep in mind that all phases of the life cycle must be fully implemented by qualified network security managers to achieve a secure network. Even then security is not an end state, but rather a process. As the writer often advises his students, "A network is never secure, only well monitored"

Robust open source tools are available to aid in the detection phase of the Security Life Cycle. These include Sguil, Snort, Tcpflow, Sancp, P0f, Ethereal, and others that were not mentioned in this paper. In the past tools such as these were either very expensive, proprietary, or both. Due to the UNIX philosophy which encourages the development of interoperable tools, it would be expected that additional tools will be added to the list above.

REFERENCES

- [1] H. G. Kayacik and A. N. Zincir-Heywood, "A case study of three open source security management tools," 2003.
- [2] L. T. Heberlein, G. V. Dias, K. N. Levitt, B. Mukherjee, J. Wood, and D. Wolber, "A network security monitor," 1990.
- [3] R. Bejtlich, "Integrating the Network Security Monitoring Mode," *Sys Admin the journal for UNIX and Linux system administrators*, vol. 13, 2004.
- [4] R. Bejtlich, *The Tao of Network Security Monitoring Beyond Intrusion Detection*: Addison-Wesley, 2004.
- [5] J. Davis, E. Hill, L. Spradley, M. Wright, W. Scherer, and Y. Zhang, "Network security monitoring - intrusion detection," 2003.
- [6] P. H. Salus, "A Quarter-Century of Unix," Addison-Wesley, 1994.
- [7] B. Visscher, "SGUIL - The Analyst Console for Network Security Monitoring," 2005.
- [8] G. Lawton, "Open source security: opportunity or oxymoron?," *Computer*, vol. 35, pp. 18-21, 2002.
- [9] J. H. Brian Caswell, "Snort Manual," 2005.
- [10] J. Elson, "tcpflow," 2001.
- [11] J. Curry, "Security Analyst Network Connection Profiler," 2004.
- [12] M. Zalewski, "the new p0f: 2.0.5," 2004.
- [13] "Ethereal," 2005.