

eLearnSecurity Certified Incident Response (eCIR) – Guide Study to Exam

<https://www.linkedin.com/in/joas-antonio-dos-santos>



Summary

| | |
|--|-----------|
| eLearnSecurity Certified Incident Response (eCIR) – Guide Study to Exam..... | 1 |
| Laboratory..... | 10 |
| Content and Exam Reviews..... | 10 |
| Network Packet and Traffic Analysis..... | 10 |
| What is Network Traffic Analysis (NTA)?..... | 10 |
| The key benefits of network traffic analysis..... | 11 |
| The importance of network traffic analysis | 11 |
| What is the purpose of analyzing and monitoring network traffic? | 12 |
| What to look for in a network traffic analysis and monitoring solution..... | 13 |
| Network Topologies:..... | 14 |
| Types of Network:..... | 14 |
| TCP/IP Model and OSI Model: | 15 |
| Computer Network Protocols:..... | 16 |
| IEEE Standards:..... | 23 |
| Networking Devices:..... | 25 |
| Cables in Networking Devices:..... | 26 |
| Types of Ethernet Networks: | 28 |
| Types of Network Connections: | 28 |
| Transmission Meia:..... | 29 |
| Types of Multiplexers: | 30 |
| Collision Detection:..... | 31 |
| Network Layer Services:..... | 31 |
| Mode of Communication:..... | 32 |
| Classes in Computer Networking: | 33 |
| Subnetting: | 35 |
| Methods of Network Security:..... | 38 |
| Wireshark Cheat Sheet..... | 38 |
| Examples to Understand the Power of Wireshark | 38 |
| Installation of Wireshark..... | 39 |
| Getting Started with Filters..... | 40 |
| Follow the White Rabbit Stream | 41 |
| Resolve DNS in Wireshark | 41 |
| Tshark for the Command Line | 41 |
| Build Firewall Rules | 42 |
| Wireshark GeoIP Mapping..... | 42 |

| | |
|---|----|
| Decrypt SSL/TLS sessions | 43 |
| Extract files from PCAP using Export (HTTP or SMB) | 45 |
| Right Hand Status Bar..... | 45 |
| Sample PCAP's are readily available | 45 |
| Setting up your Environment..... | 45 |
| capinfos | 46 |
| Default Columns In a Packet Capture Output..... | 47 |
| Logical Operators..... | 48 |
| Filtering Packets (Display Filters) | 49 |
| Filter Types..... | 50 |
| Wireshark Capturing Modes | 51 |
| Miscellaneous..... | 52 |
| Capture Filter Syntax | 53 |
| Display Filter Syntax..... | 53 |
| Keyboard Shortcuts – Main Display Window..... | 54 |
| Protocols – Values | 56 |
| Common Filtering Commands | 56 |
| Main Toolbar Items..... | 57 |
| Others | 60 |
| Exporting Data | 60 |
| 5.7.1. The “Export Specified Packets” Dialog Box | 60 |
| 5.7.2. The “Export Packet Dissections” Dialog Box..... | 61 |
| 5.7.3. The “Export Selected Packet Bytes” Dialog Box | 66 |
| 5.7.4. The “Export PDUs to File...” Dialog Box | 67 |
| 5.7.5. The “Strip Headers...” Dialog Box..... | 68 |
| 5.7.6. The “Export TLS Session Keys...” Dialog Box..... | 69 |
| 5.7.7. The “Export Objects” Dialog Box..... | 70 |
| ElasticSearch, Kibana and Logstash (ELK)..... | 88 |
| What is the ELK Stack? | 88 |
| Why is ELK So Popular? | 89 |
| Why is Log Analysis Becoming More Important?..... | 90 |
| How to Use the ELK Stack for Log Analysis..... | 91 |
| What’s new? | 92 |
| Installing ELK..... | 93 |

| | |
|--|-----|
| Environment specifications | 94 |
| Installing Elasticsearch | 94 |
| Installing Logstash | 96 |
| Installing Kibana | 97 |
| Installing Beats | 98 |
| Shipping some data | 98 |
| Additional installation guides | 101 |
| Elasticsearch..... | 101 |
| What is Elasticsearch? | 101 |
| Basic Elasticsearch Concepts..... | 102 |
| Index..... | 102 |
| Documents..... | 103 |
| Types | 103 |
| Mapping..... | 104 |
| Shards..... | 104 |
| Replicas..... | 104 |
| Elasticsearch Queries | 105 |
| Boolean Operators | 105 |
| Fields | 105 |
| Ranges..... | 105 |
| Wildcards, Regexes and Fuzzy Searching | 106 |
| URI Search..... | 106 |
| Elasticsearch REST API | 107 |
| Elasticsearch Document API..... | 108 |
| Elasticsearch Search API..... | 108 |
| Elasticsearch Indices API | 108 |
| Elasticsearch Cluster API | 108 |
| Elasticsearch Plugins | 108 |
| Plugin Categories | 109 |
| Installing Elasticsearch Plugins..... | 109 |
| What's next?..... | 110 |
| Logstash | 110 |
| What is Logstash? | 110 |
| Logstash Configuration..... | 111 |
| Input plugins..... | 112 |

| | |
|------------------------------------|------------|
| Filter plugins | 112 |
| Output plugins | 112 |
| Logstash Codecs..... | 113 |
| Configuration example..... | 113 |
| Input..... | 114 |
| Filter | 114 |
| Output | 115 |
| Complete example..... | 115 |
| Logstash pitfalls..... | 115 |
| Monitoring Logstash | 116 |
| What next? | 118 |
| Kibana..... | 118 |
| What is Kibana? | 118 |
| Kibana searches | 119 |
| Kibana searches cheat sheet..... | 120 |
| Kibana autocomplete | 121 |
| Kibana filtering | 121 |
| Kibana visualizations | 122 |
| Visualization types..... | 123 |
| Kibana dashboards | 126 |
| Kibana pages..... | 126 |
| Kibana Elasticsearch index | 127 |
| What's next?..... | 128 |
| Beats..... | 128 |
| What are Beats?..... | 128 |
| Filebeat..... | 129 |
| Packetbeat..... | 129 |
| Metricbeat..... | 130 |
| Winlogbeat | 130 |
| Auditbeat..... | 130 |
| Functionbeat..... | 130 |
| Configuring beats..... | 131 |
| Beats modules..... | 131 |
| Configuration example..... | 132 |
| Configuration best practices | 132 |

| | |
|---|-----|
| What next? | 133 |
| ELK in Production | 133 |
| Don't Lose Log Data | 134 |
| Monitor Logstash/Elasticsearch Exceptions | 135 |
| Keep up with growth and bursts..... | 136 |
| ELK Elasticity | 136 |
| Kafka | 137 |
| Logstash..... | 138 |
| Elasticsearch cluster..... | 139 |
| Run in Different AZs (But Not in Different Regions) | 139 |
| Security..... | 140 |
| Maintainability | 141 |
| Log Data Consistency | 141 |
| Data Retention | 141 |
| Upgrades..... | 142 |
| Summary..... | 142 |
| Common Pitfalls | 143 |
| Elasticsearch..... | 143 |
| Not defining Elasticsearch mapping..... | 143 |
| Capacity Provisioning | 144 |
| Oversized Template..... | 145 |
| Production Fine-tuning | 145 |
| Logstash..... | 146 |
| Logstash configuration file | 146 |
| Memory consumption | 146 |
| Slow processing | 147 |
| Key-Value Filter Plugin | 147 |
| Kibana..... | 148 |
| Elasticsearch connectivity..... | 148 |
| Defining an index pattern | 148 |
| Can not connect to Elasticsearch..... | 148 |
| Bad Kibana searches..... | 149 |
| Advanced settings..... | 150 |
| Beats | 150 |
| YAML configuration files | 151 |

| | |
|--|------------|
| Filebeat – CPU Usage..... | 151 |
| Filebeat – Registry File | 151 |
| Filebeat – Removed or Renamed Log Files..... | 152 |
| Summing it up | 152 |
| Use Cases | 153 |
| Development and troubleshooting | 153 |
| Cloud operations | 154 |
| Application performance monitoring (APM)..... | 155 |
| Security and compliance | 155 |
| 1.Anti-DDoS | 156 |
| 2.SIEM..... | 156 |
| Business Intelligence (BI) | 157 |
| SEO | 158 |
| Integrations..... | 158 |
| Winlogbeat configuration options..... | 159 |
| Configuring our logging pipeline | 161 |
| Analyzing and visualizing event logs | 162 |
| Splunk..... | 168 |
| Monitor Windows event log data with Splunk Enterprise | 168 |
| Why monitor event logs? | 168 |
| Requirements for monitoring event logs..... | 168 |
| Security and other considerations for collecting event log data from remote machines | 169 |
| How the Windows Event Log monitor interacts with Active Directory | 169 |
| Collect event logs from a remote Windows machine | 170 |
| Use a universal or heavy forwarder | 170 |
| Use WMI | 170 |
| Anomalous machine names are visible in event logs on some systems..... | 171 |
| Configure local event log monitoring with Splunk Web..... | 171 |
| Go to the Add Data page | 171 |
| Select the input source | 172 |
| Specify input settings | 172 |
| Review your choices..... | 173 |
| Configure remote event log monitoring with Splunk Web | 173 |
| Use the inputs.conf configuration file to configure event log monitoring | 174 |
| Specify global settings for Windows Event Log inputs..... | 174 |

| | |
|---|-----|
| Event log monitor configuration values | 175 |
| Monitor non-default Windows event logs | 175 |
| Use the Full Name log property in Event Viewer to specify complex Event Log channel names properly..... | 175 |
| Disable an event log stanza..... | 176 |
| Configuration settings for monitoring Windows Event Logs..... | 176 |
| Use the Security event log to monitor changes to files..... | 180 |
| Required data..... | 182 |
| How to use Splunk software for this use case | 182 |
| Monitor Windows host information | 182 |
| Why monitor host information?..... | 183 |
| Requirements..... | 184 |
| Security and remote access considerations | 184 |
| Use the inputs.conf configuration file to configure host monitoring..... | 184 |
| Windows host monitor configuration values | 185 |
| Use Splunk Web to configure host monitoring | 185 |
| Go to the Add Data page | 186 |
| Select the input source | 186 |
| Specify input settings | 186 |
| Review your choices..... | 187 |
| Examples of Windows host monitoring configurations | 187 |
| PowerShell Detections..... | 188 |
| Responding to PowerShell with Automated Playbooks..... | 191 |
| Summary of Logging Types..... | 192 |
| Hunting Analytic..... | 194 |
| Detections..... | 196 |
| How to Enable It? | 198 |
| Registry | 199 |
| PowerShell..... | 199 |
| Enable Logging via Group Policy Objects | 201 |
| Test Yourself | 202 |
| Active Directory Kerberos Attacks Analytic Story..... | 203 |
| Discovery - TA0007..... | 204 |
| Credential Access - TA0006 | 206 |
| Lateral Movement - TA0008 | 212 |
| Privilege Escalation - TA0004 | 216 |

| | |
|---|-----|
| Datasets | 218 |
| Cyber Kill Chain..... | 220 |
| What is the Cyber Kill Chain? | 220 |
| 8 Phases of the Cyber Kill Chain Process..... | 220 |
| Evolution of the Cyber Kill Chain..... | 221 |
| Critiques and Concerns Related to the Cyber Kill Chain | 222 |
| Role of the Cyber Kill Chain in Cybersecurity | 223 |
| Event Correlation | 223 |
| How does IT event correlation work? | 223 |
| How are events correlated with machine learning? | 226 |
| How do you identify patterns in various IT events?..... | 227 |
| What are some of the benefits of IT event correlation?..... | 228 |
| Can you protect IT infrastructure with IT event correlation monitoring? | 229 |
| What are some of the biggest challenges to efficient event correlation? | 229 |
| How do you integrate IT event correlation into SIEM? | 230 |
| How do you get started with event correlation?..... | 230 |
| What is the future of IT event correlation?..... | 232 |
| The Bottom Line: Event correlation makes sense of your infrastructure | 232 |
| Event Correlation Use Cases and Techniques..... | 232 |
| Searches in SIEM with Regex..... | 233 |
| Other Tool | 233 |
| How to Use Regex | 233 |
| The Basics of Regex | 235 |
| Regex Examples | 236 |
| Regexp queryedit | 238 |
| Example requestedit..... | 238 |
| Top-level parameters for regexpedit..... | 238 |
| Parameters for <field>edit | 238 |
| ChatGPT Regex | 239 |
| 1. Simplify SIEM Query Writing with ChatGPT | 239 |
| 2. Generate Security Detections | 241 |
| 3. Write Security Policies | 243 |
| 4. Identify Vulnerabilities | 244 |
| 5. Write Automation Scripts for Daily Operations | 246 |
| 6. Assist Incident Response | 248 |

Laboratory

<https://unicornsec.com/home/siem-home-lab-series-part-1>

<https://www.leveleffect.com/blog/how-to-set-up-your-own-home-lab-with-elk>

https://www.youtube.com/watch?v=YUEMjWk6dvk&ab_channel=HackeXPlorer

<https://github.com/deargle/lab-security-onion>

<https://www.hackingarticles.in/siem-log-monitoring-lab-setup-with-splunk/>

https://www.youtube.com/watch?v=ia9E4x8iVDk&ab_channel=DayCyberwox

https://www.youtube.com/watch?v=CG9QxkWU8hA&ab_channel=CyberInsight

Content and Exam Reviews

<https://haydz.github.io/2020/04/20/elearnircert.html>

<https://www.bencteux.fr/posts/ecir/>

<https://www.linkedin.com/pulse/my-review-ecir-exam-joas-a-santos/>

<https://subtlystoic.medium.com/how-to-prepare-for-the-ecir-exam-5735235b2098>

<https://ivanitlearning.wordpress.com/2020/07/14/review-of-elss-ihrp-course/>

https://www.youtube.com/watch?v=3t1BNAavrIQ&ab_channel=HackeXPlorer

https://www.youtube.com/watch?v=NbXBOalfGfU&ab_channel=I.TSecurityLabs

https://www.youtube.com/watch?v=7DRHt8LN_g&ab_channel=Splunk

Network Packet and Traffic Analysis

What is Network Traffic Analysis (NTA)?

Network traffic analysis (NTA) is a method of monitoring network availability and activity to identify anomalies, including security and operational issues. Common use cases for NTA include:

- Collecting a real-time and historical record of what's happening on your network
- Detecting malware such as ransomware activity
- Detecting the use of vulnerable protocols and ciphers
- Troubleshooting a slow network
- Improving internal visibility and eliminating blind spots

Implementing [a solution that can continuously monitor network traffic](#) gives you the insight you need to optimize network performance, minimize your attack surface, enhance security, and improve the management of your resources. However, knowing how to monitor network traffic is not enough. It's important to also consider the data sources for your network monitoring.

tool; two of the most common are flow data (acquired from devices like routers) and packet data (from SPAN, mirror ports, and network TAPs).

The key benefits of network traffic analysis

With the “it’s not if, it’s when” mindset regarding cyber attacks today, it can feel overwhelming for security professionals to ensure that as much of an organization’s environment is covered as possible. The network is a critical element of their attack surface; gaining visibility into their network data provides one more area they can detect attacks and stop them early.

Benefits of NTA include:

- Improved visibility into devices connecting to your network (e.g. IoT devices, healthcare visitors)
- Meet compliance requirements
- Troubleshoot operational and security issues
- Respond to investigations faster with rich detail and additional network context

A key step of setting up NTA is ensuring you’re collecting data from the right sources. Flow data is great if you are looking for traffic volumes and mapping the journey of a network packet from its origin to its destination. This level of information can help detect unauthorized WAN traffic and utilize network resources and performance, but it can lack rich detail and context to dig into cybersecurity issues.

Packet data extracted from network packets can help network managers understand how users are implementing/operating applications, track usage on WAN links, and monitor for suspicious malware or other security incidents. Deep packet inspection (DPI) tools provide 100% visibility over the network by transforming the raw metadata into a readable format and enabling network and security managers to drill down to the minutest detail.

The importance of network traffic analysis

Keeping a close eye on your network perimeter is always good practice. Even with strong firewalls in place, mistakes can happen and rogue traffic could get through. Users could also leverage methods such as tunneling, external anonymizers, and VPNs to get around firewall rules.

Additionally, the rise of ransomware as [a common attack type](#) in recent years makes network traffic monitoring even more critical. A network monitoring solution should be able to detect activity indicative of ransomware attacks via insecure protocols. Take WannaCry, for example,

where attackers actively scanned for networks with TCP port 445 open, and then used a vulnerability in SMBv1 to access network file shares.

Remote Desktop Protocol (RDP) is another commonly targeted application. Make sure you block any inbound connection attempts on your firewall. Monitoring traffic inside your firewalls allows you to validate rules, gain valuable insight, and can also be used as a source of network traffic-based alerts.

Watch out for any suspicious activity associated with management protocols such as Telnet. Because Telnet is an unencrypted protocol, session traffic will reveal command line interface (CLI) command sequences appropriate for the make and model of the device. CLI strings may reveal login procedures, presentation of user credentials, commands to display boot or running configuration, copying files, and more. Be sure to check your network data for any devices running unencrypted management protocols, such as:

- Telnet
- Hypertext Transport Protocol (HTTP, port 80)
- Simple Network Management Protocol (SNMP, ports 161/162)
- Cisco Smart Install (SMI port 4786)

What is the purpose of analyzing and monitoring network traffic?

Many operational and security issues can be investigated by implementing network traffic analysis at both the network edge and the network core. With the traffic analysis tool, you can spot things like large downloads, streaming or suspicious inbound or outbound traffic. Make sure you start off by monitoring the internal interfaces of firewalls, which will allow you to track activity back to specific clients or users.

NTA also provides an organization with more visibility into threats on their networks, beyond the endpoint. With the rise in mobile devices, IoT devices, smart TV's, etc., you need something with more intelligence than just the logs from firewalls. Firewall logs are also problematic when a network is under attack. You may find that they are inaccessible due to resource load on the firewall or that they've been overwritten (or sometimes even modified by hackers), resulting in the loss of vital forensic information.

Some of the use cases for analyzing and monitoring network traffic include:

- Detection of ransomware activity
- Monitoring data exfiltration/internet activity
- Monitor access to files on file servers or MSSQL databases
- Track a user's activity on the network, though User Forensics reporting
- Provide an inventory of what devices, servers and services are running on the network

- Highlight and identify root cause of bandwidth peaks on the network
- Provide real-time dashboards focusing on network and user activity
- Generate network activity reports for management and auditors for any time period

What to look for in a network traffic analysis and monitoring solution

Not all tools for monitoring network traffic are the same. Generally, they can be broken down into two types: flow-based tools and deep packet inspection (DPI) tools. Within these tools you'll have options for software agents, storing historical data, and intrusion detection systems. When evaluating which solution is right for your organization, consider these five things:

1. **Availability of flow-enabled devices:** Do you have flow-enabled devices on your network capable of generating the flows required by a NTA tool that only accepts flows like Cisco Netflow? DPI tools accept raw traffic, found on every network via any managed switch, and are vendor independent. Network switches and routers do not require any special modules or support, just traffic from a SPAN or port mirror from any managed switch.
2. **The data source:** Flow data and packet data come from different sources, and not all NTA tools collect both. Be sure to look through your network traffic and decide which pieces are critical, and then compare capabilities against the tools to ensure everything you need is covered.
3. **The points on the network:** Consider whether the tool uses agent-based software or agent-free. Also be careful not to monitor too many data sources right out the gate. Instead, be strategic in picking locations where data converges, such as internet gateways or VLANs associated with critical servers.
4. **Real-time data vs. historical data:** Historical data is critical to analyzing past events, but some tools for monitoring network traffic don't retain that data as time goes on. Also check whether the tool is priced based on the amount of data you want to store. Have a clear understanding of which data you care about most to find the option best suited to your needs and budget.
5. **Full packet capture, cost and complexity:** Some DPI tools capture and retain all packets, resulting in expensive appliances, increased storage costs, and much training/expertise to operate. Others do more of the 'heavy lifting,' capturing full packets but extracting only the critical detail and metadata for each protocol. This metadata extraction results in a huge data reduction but still has readable, actionable detail that's ideal for both network and security teams.

[https://www.rapid7.com/fundamentals/network-traffic-analysis/#:~:text=Network%20traffic%20analysis%20\(NTA\)%20is,malware%20such%20as%20ransomware%20activity](https://www.rapid7.com/fundamentals/network-traffic-analysis/#:~:text=Network%20traffic%20analysis%20(NTA)%20is,malware%20such%20as%20ransomware%20activity)

Network Topologies:

| Name | Description |
|--|--|
| <u>Bus Topology</u> | A bus topology, also called a line topology, is a type of network topology in which all network devices are connected through a central RJ-45 network cable or coaxial cable. |
| <u>Ring Topology</u> | A ring topology is a type of network topology in which each device is connected to two other devices on either side using RJ-45 or coaxial cables. |
| <u>Star Topology</u> | A star topology is a network topology in which each element of the network is physically connected to a central node such as a router, hub, or switch. In a star topology, hubs act as servers, and connecting nodes act as clients. |
| <u>Mesh Topology</u> | In a mesh topology, each node is connected to at least one other node and often to multiple nodes. Each node can send and receive messages from other nodes. |
| <u>Tree Topology</u> | A tree topology is a hybrid network topology in which star networks are interconnected by bus networks. Tree networks are hierarchical and each node can have any number of child nodes. |
| <u>Hybrid Topology</u> | A hybrid topology is a type of network topology that uses two or more different network topologies. These topologies can include mixed bus topologies, mesh topologies, ring topologies, star topologies, and tree topologies. |

Types of Network:

| Network Type | Description |
|----------------------------|--|
| <u>PAN</u> | Personal Network is a network consisting of only a small number of devices owned by an individual. |

| Network Type | Description |
|---------------------|--|
| <u>LAN</u> | A local area network is a network that covers a small area (for example, a company's network). |
| <u>WAN</u> | A wide Area Network is a network that includes many devices and covers a large area. Usually collectively owned. |
| <u>MAN</u> | MAN stands for Metropolitan Area Network. It is a computer network that connects a findnumber of LANs to form a larger network so that the computer resources can be shared. |

TCP/IP Model and OSI Model:

| TCP/IP | OSI Model | Protocols |
|--------------------------|---------------------------|---|
| <u>Application Layer</u> | <u>Application Layer</u> | DNS, DHCP, FTP, HTTPS, IMAP, LDAP, NTP, POP3, RTP, RTSP, SSH, SIP, SMTP, SNMP, Telnet, TFTP |
| | <u>Presentation Layer</u> | JPEG, MIDI, MPEG, PICT, TIFF |
| | <u>Session Layer</u> | NetBIOS, NFS, PAP, SCP, SQL, ZIP |
| <u>Transport Layer</u> | <u>Transport Layer</u> | TCP, UDP |
| <u>Internet Layer</u> | <u>Network Layer</u> | ICMP, IGMP, IPsec, IPv4, IPv6, IPX, RIP |

| TCP/IP | OSI Model | Protocols |
|-----------------------------------|--|--|
| <u>Link Layer</u> | <u>Data Link Layer</u> | ARP, ATM, CDP, FDDI, Frame Relay, HDLC, MPLS, PPP, STP, Token Ring |
| | <u>Physical Layer</u> | Bluetooth, Ethernet, DSL, ISDN, 802.11 Wi-Fi |

Computer Network Protocols:

| Network Protocol | Description | Port number of protocol |
|--------------------------------------|---|--------------------------------|
| <u>Ethernet</u> | A family of protocols that specify how devices on the same network segment format and transmit data. | 44818, 2222 |
| <u>Wi-Fi or WLAN</u> | A family of protocols that deal with wireless transmission. | — |
| <u>TCP</u> | Splits data into packets (reassembles later). Error checking is also included, as the acknowledgment is expected to be sent within a specified timeframe. | 22 |
| <u>UDP</u> | User Datagram Protocol | 4096-65535 |
| <u>IP</u> | Every device has an IP address. Packets are “addressed” to ensure they reach the correct user. | — |
| <u>HTTP</u> | Used to access web pages from a web server. | 80 |
| HTTP'S | uses encryption to protect data. | 443 |
| <u>FTP</u> | File Transfer Protocol: Handles file uploads and downloads, transferring data and programs. | 21 |

| Network Protocol | Description | Port number of protocol |
|-----------------------------|---|--------------------------------|
| <u>SMTP</u> | SMTP server has a database of user email addresses. Internet Message Access Protocol: Handles incoming mail. | 587 |
| <u>IMAP</u> | Internet Message Access Protocol: Process incoming mail. | 993 |
| <u>ARP</u> | ARP finds a host's hardware address (also known as MAC (Media Access Control) address) based on its known IP address. | - |
| <u>DNS</u> | DNS is the host name for the IP address translation service. DNS is a distributed database implemented on a hierarchy of name servers. It is an application layer protocol for messaging between clients and servers. | 53 |
| <u>FTPS</u> | FTPS is known as FTP SSL which refers to <u>File Transfer Protocol (FTP)</u> over <u>Secure Sockets Layer (SSL)</u> which is more secure from FTP. FTPS also called as File Transfer Protocol Secure. | 21 |
| <u>POP3</u> | POP3 is a simple protocol that only allows downloading messages from your Inbox to your local computer. | 110 |
| <u>SIP</u> | <u>Session Initiation Protocol</u> was designed by IETF and is described in RFC 3261. It's the protocol of <u>application layer</u> that describes the way to found out Internet telephone calls, video conferences and other multimedia connections, manage them and terminate them. | 5060,5061 |
| <u>SMB</u> | The SMB protocol was developed by Microsoft for direct file sharing over local networks. | 139 |

| Network Protocol | Description | Port number of protocol |
|------------------------------|--|--------------------------------|
| <u>SNMP</u> | SNMP is an application layer protocol that uses UDP port numbers 161/162. SNMP is also used to monitor networks, detect network errors, and sometimes configure remote devices. | 161 |
| <u>SSH</u> | SSH (Secure Shell) is the permissions used by the SSH protocol. That is, a cryptographic network protocol used to send encrypted data over a network. | 22 |
| <u>VNC</u> | VNC stands for Virtual Network Communication. | 5900 |
| <u>RPC</u> | Remote Procedure Call (RPC) is a powerful technique for building distributed client-server based applications. It is based on extending traditional calls to local procedures so that the called procedure does not have to be in the same address space as the calling procedure. | 1024 to 5000 |
| <u>NFS</u> | NFS uses file handles to uniquely identify the file or directory on which the current operation is being performed. Internet Control Message Protocol (ICMP) to provide error control. Used for reporting errors and administrative queries. | 2049 |
| <u>ICMP</u> | Internet Control Message Protocol(ICMP) to provide an error control. It is used for reporting errors and management queries. | — |
| <u>BOOTP</u> | Bootstrap Protocol (BOOTP) is a network protocol used by network management to assign IP addresses to each member of that network in order to join other network devices through a main server. | 67 |

| Network Protocol | Description | Port number of protocol |
|------------------------------|---|--------------------------------|
| <u>DHCP</u> | Dynamic Host Configuration Protocol (DHCP) is an application layer protocol. DHCP is based on a client-server model, based on discoveries, offers, requests, and ACKs. | 68 |
| <u>NAT</u> | Network Address Translation (NAT) is the process of translating one or more local IP addresses into one or more global IP addresses, or vice versa, in order to provide Internet access to local hosts. | 5351 |
| <u>PPP</u> | Point-to-Point Protocol (PPP) is basically an asymmetric protocol suite for various connections or links without framing. H. Raw bit pipe. PPP also expects other protocols to establish connections, authenticate users, and carry network layer data as well. | 1994 |
| <u>RIP</u> | Routing Information Protocol (RIP) is a dynamic routing protocol that uses hop count as a routing metric to find the best path between source and destination networks. | 520 |
| <u>OSPF</u> | Open Shortest Path First (OSPF) is a link-state routing protocol used to find the best path between a source and destination router using its own shortest path first). | 89 |
| <u>EIGRP</u> | Enhanced Interior Gateway Routing Protocol (EIGRP) is a dynamic routing protocol used to find the best path and deliver packets between any two Layer 3 devices. | 88 |
| <u>BGP</u> | Border Gateway Protocol (BGP) is a protocol used to exchange Internet routing information and is used between ISPs in different ASes. | 179 |

| Network Protocol | Description | Port number of protocol |
|-------------------------------|--|--------------------------------|
| <u>STP</u> | Spanning Tree Protocol (STP) is used to create a loop-free network by monitoring the network, tracking all connections, and shutting down the least redundant connections. | 0 to 255 |
| <u>RARP</u> | RARP, stand for Reverse Address Resolution Protocol, is a computer network-based protocol used by client computers to request IP addresses from a gateway server's Address Resolution Protocol table or cache. | — |
| <u>LAPD</u> | The D-channel LAPD or Link Access Protocol is basically the Layer 2 protocol normally required for the ISDN D-channel. It is derived from the LAPB (Link Access Protocol Balanced) protocol. | — |
| <u>IPsec</u> | IP Security (IPSec) is a standard suite of Internet Engineering Task Force (IETF) protocols between two communication points on IP networks to provide data authentication, integrity, and confidentiality. It also defines encrypted, decrypted, and authenticated packets. | 4500 |
| <u>ASCII</u> | ASCII (American Standard Code for Information Interchange) is the standard character encoding used in telecommunications. The ASCII representation “ask-ee” is strictly a 7-bit code based on the English alphabet. ASCII codes are used to represent alphanumeric data. | 9500 |
| <u>EBCDIC</u> | EBCDIC (Extended Binary Encoded Decimal Interchange Code) (pronounced “ehb-suh-dik” or “ehb-kuh-dik”) is an alphanumeric binary code developed by IBM to run large-scale computer systems . | — |

| Network Protocol | Description | Port number of protocol |
|----------------------------------|--|--------------------------------|
| <u>X.25 PAD</u> | X.25 is an International Telecommunication Union Telecommunication Standardization Sector (ITU-T) protocol standard simply for <u>Wide Area Network (WAN)</u> communications that basically describes how the connections among user devices and network devices are established and maintained. | – |
| <u>HDLC</u> | High-Level Data Link Control (HDLC) commonly uses the term “frame” to denote units or logs of units of data that are frequently transmitted or transmitted from one station to another, express. Each frame on the link must start and end with a flag sequence field (F). | – |
| <u>SLIP</u> | SLIP stands for Serial Line Internet Protocol. It is a TCP/IP implementation which was described under RFC 1055 (Request for Comments). | |
| <u>LAP</u> | Link Access Procedure (LAP) is basically considered as an ITU family of <u>Data Link Layer (DLL) protocols</u> that are subsets of <u>High-Level Data Link Control (HDLC)</u> . LAP is particularly derived from IBM's <u>System Development Life Cycle (SDLC)</u> . | – |
| NCP | Network Control Protocol (NCP) is a set of protocols that are part of Point-to-Point Protocol (PPP). | 524 |
| <u>Mobile IP</u> | Mobile IP is a communication protocol (created by extending the Internet Protocol, IP) that allows a user to move from one network to another using the same her IP address. | 434 |
| <u>VOIP</u> | Voice over Internet Protocol (VoIP), is a technology that allowing you to make voice calls over a broadband Internet connection instead of an analog (regular) phone line. Some VoIP services allow you | 5060 |

| Network Protocol | Description | Port number of protocol |
|---------------------------------|---|--------------------------------|
| | to call people using the same service, but others may allow you to call anyone. | |
| <u>LDAP</u> | Lightweight Directory Access Protocol (LDAP) is an internet protocol works on TCP/IP, used to access information from directories. LDAP protocol is basically used to access an active directory. | 389 |
| <u>GRE</u> | GRE or Generic Routing Encapsulation is a tunneling protocol developed by Cisco. It encapsulates IP packets i.e. deliverable inner packets into outer packets. | 47 |
| <u>AH</u> | The HTTP headers Authorization header is a request type header that used to contains the credentials information to authenticate a user through a server. If the server responds with 401 Unauthorized and the WWW-Authenticate header not usually. | 51 |
| <u>ESP</u> | Encapsulation security payload, also abbreviated as ESP plays a very important role in network security. ESP or Encapsulation security payload is an individual protocol in IPSec. | 500 |
| <u>NNTP</u> | Network News Transfer Protocol (NNTP) is the underlying protocol of UseNet, which is a worldwide discussion system which contains posts or articles which are known as news. | 119 |
| <u>RPC-DCOM</u> | <u>DCOM- Distributed Component Object Model</u> — helps remote object via running on a protocol known as the Object Remote Procedure Call (ORPC). | — |
| <u>IRC</u> | Internet Relay Chat (IRC) is an Internet application that was developed by Jakko Oikarinen in Finland. | 6667 |

| Network Protocol | Description | Port number of protocol |
|-------------------------|--|--------------------------------|
| | Chat is the most convenient immediate way to communicate with others via Internet. | |

IEEE Standards:

| Standards | Description |
|------------------|---|
| IEEE 802 | <u>LAN/MAN</u> |
| IEEE 802.1 | LAN/MAN Bridging and management |
| IEEE 802.1s | <u>Multiple spanning tree</u> |
| IEEE 802.1w | <u>Rapid reconfiguration of spanning tree</u> |
| IEEE 802.1x | <u>Port-based network access control</u> |
| IEEE 802.2 | <u>Logical Link Control (LLC)</u> |
| IEEE 802.3 | <u>CSMA/CD access method (Ethernet)</u> |
| IEEE 802.3ae | <u>10 Gigabit Ethernet</u> |
| IEEE 802.4 | Token passing bus access method and Physical layer specifications |

| Standards | Description |
|------------------|--|
| IEEE 802.5 | Token Ring access method and Physical layer specifications |
| IEEE 802.6 | <u>Distributed Queue Dual Bus (DQDB) access method and Physical layer specifications (MAN)</u> |
| IEEE 802.7 | <u>Broadband LAN</u> |
| IEEE 802.8 | <u>Fiber Optic</u> |
| IEEE 802.9 | Isochronous LANs (standard withdrawn) |
| IEEE 802.10 | Interoperable LAN/MAN Security |
| IEEE 802.11 | <u>Wireless LAN MAC and Physical layer specifications</u> |
| IEEE 802.11a | Wireless with speed upto 54 Mbps |
| IEEE 802.11b | <u>Wireless with speed upto 11 Mbps</u> |
| IEEE 802.11g | Wireless with speed upto 54 Mbps |
| IEEE 802.11n | Wireless with speed upto 600 Mbps |
| IEEE 802.12 | Demand-priority access method, physical layer and repeater specifications |

| Standards | Description |
|------------------|---|
| IEEE 802.13 | Not used |
| IEEE 802.14 | Cable modems (proposed standard was withdrawn) |
| IEEE 802.15 | Wireless Personal Area Network (WPAN) |
| IEEE 802.16 | Wireless Metropolitan Area Network (Wireless MAN) |
| IEEE 802.17 | Resilient Packet Ring (RPR) Access |

Networking Devices:

| Device | Description |
|---------------|---|
| Client | Any device, such as a workstation, laptop, tablet, or smartphone, that is used to access a network. |
| Server | Provides resources to network users, including email, web pages, or files. |
| Hub | A Layer 1 device that does not perform any inspection of traffic. A hub simply receives traffic in a port and repeats that traffic out of all the other ports. |
| Switch | A Layer 2 device that makes its forwarding decisions based on the destination Media Access Control (MAC) address . A switch learns which devices reside off which ports by examining the source MAC address. The switch then forwards traffic only to the appropriate port, and not to all the other ports. |
| Router | A Layer 3 device that makes forwarding decisions based on Internet Protocol (IP) addressing. Based on the routing table, the |

| Device | Description |
|-------------------|---|
| | router intelligently forwards the traffic out of the appropriate interface. |
| Multilayer switch | Can operate at both Layer 2 and Layer 3 . Also called a Layer 3 switch, a multilayer switch is a high-performance device that can switch traffic within the LAN and forward packets between subnets. |
| Media | Media can be copper cabling, fiber-optic cabling , or radio waves. Media varies in its cost, bandwidth capacity, and distance limitation. |
| Analog modem | Modem is short for modulator/demodulator . An analog modem converts the digital signals generated by a computer into analog signals that can travel over conventional phone lines. |
| Broadband modem | A digital modem used with high-speed DSL or cable Internet service. Both operate in a similar manner to the analog modem, but use higher broadband frequencies and transmission speeds. |
| Access point (AP) | A network device with a built-in antenna, transmitter, and adapter that provides a connection point between WLANs and a wired Ethernet LAN. APs usually have several wired RJ-45 ports to support LAN clients. Most small office or home office (SOHO) routers integrate an AP. |

Cables in Networking Devices:

| Ethernet Standards (IEEE) | Data Rate | Cable Fiber Type | Maximum Distance (IEEE) |
|--------------------------------------|------------------|------------------------------------|--------------------------------|
| Ethernet (10Base-FL) | 10 Mbps | 50m or 62.5um Multimode @ 850nm | 2km |

| Ethernet Standards (IEEE) | Data Rate | Cable Fiber Type | Maximum Distance (IEEE) |
|---|------------------|----------------------------------|--------------------------------|
| <u>Fast Ethernet (100Base-FX)</u> | 100 Mbps | 50m or 62.5um Multimode @ 1300nm | 2km |
| <u>Fast Ethernet (100Base-SX)</u> | 100 Mbps | 50m or 62.5um Multimode @ 850nm | 300m |
| <u>Gigabit Ethernet (1000Base-SX)</u> | 1000 Mbps | 50m Multimode @ 850nm | 550m |
| <u>Gigabit Ethernet (1000Base-SX)</u> | 1000 Mbps | 62.5um Multimode @ 850nm | 220m |
| <u>Gigabit Ethernet (1000Base-LX)</u> | 1000 Mbps | 50m or 62.5um Multimode @ 1300nm | 550m |
| <u>Gigabit Ethernet (1000Base-LX)</u> | 1000 Mbps | 9um Singlemode @1310nm | 5km |
| <u>Gigabit Ethernet (1000Base-LH)</u> | 1000 Mbps | 9um Singlemode @1550nm | 70km |

Types of Ethernet Networks:

| Speed | Common Name | Informal IEEE Standard Name | Formal IEEE Standard Name | Cable Type, Maximum Length |
|--------------|---|------------------------------------|----------------------------------|-----------------------------------|
| 10 Mbps | Ethernet | 10BASE-T | 802.3 | Copper, 100 m |
| 100 Mbps | Fast Ethernet | 100BASE-T | 802.3u | Copper, 100 m |
| 1000 Mbps | <u>Gigabit Ethernet</u> | 1000BASE-LX | 802.3z | Fiber, 5000 m |
| 1000 Mbps | <u>Gigabit Ethernet</u> | 1000BASE-T | 802.3ab | Copper, 100 m |
| 10 Gbps | <u>10 Gig Ethernet</u> | 10GBASE-T | 802.3an | Copper, 100 m |

Types of Network Connections:

| Type | Description |
|---------------------------------|---|
| <u>Internet</u> | A network of millions of interconnected and cooperatively connected computers is called the Internet. Internet includes people, resources and means of collaboration |
| <u>Intranet</u> | It is an internal private network built within an organization using Internet and World Wide Web standards and products that provides access to corporate information for the organization's employees. |

| Type | Description |
|---------------------------------|--|
| <u>Extranet</u> | This is a type of network that allows external users to access an organization's intranet. |

Transmission Meia:

• Guided Media:

| Type of media | Description |
|--|---|
| <u>Twisted Pair Cable</u> | It is a superimposed winding of two separately insulated conductors. As a rule, several such pairs are grouped together in a protective cover. They are the most widely used transmission media. |
| <u>Coaxial Cable</u> | It has a PVC or Teflon insulating layer and an outer plastic sheath containing two parallel conductors, each with a separate conformal protective cover. |
| <u>Optical Fiber Cable</u> | It uses the concept of light reflection through a glass or plastic core. The core is surrounded by a less dense glass or plastic shell called the cladding. Used to transfer large amounts of data. |
| <u>Stripline</u> | Stripline is a transverse electromagnetic (TEM) transmission line medium invented by Robert M. Barrett at the Air Force Cambridge Research Center in the 1950s. Stripline is the earliest form of planar transmission line. |
| <u>Microstripline</u> | Conductive material is separated from the ground plane by a dielectric layer. |

• Unguided Media:

| Type of media | Description |
|------------------------------------|--|
| <u>Radio waves</u> | These are easy to generate and can penetrate buildings. There is no need to align the transmit and receive antennas. Frequency Range: 3kHz – 1GHz AM radios, FM radios, and cordless phones use radio waves for transmission. |
| <u>Microwaves</u> | Multiplexer types: line-of-sight transmission. H. Transmitting and receiving antennas should be placed properly. The distance a signal travels is directly proportional to the height of the antenna. Frequency Range: 1GHz – 300GHz They are mainly used for mobile telephony and television distribution. |
| <u>Infrared</u> | Infrared is used for short distance communication. Obstacles cannot be penetrated. This prevents interference between systems. Frequency Range: 300GHz – 400THz It is used in TV remote controls, wireless mice, keyboards, printers, etc. |

Types of Multiplexers:

| Type | Description |
|--|--|
| <u>Frequency Division Multiplexing (FDM)</u> | The frequency spectrum is divided into logical channels and each user has exclusive access to his channel. It transmits signals in several different frequency ranges and multiple video channels over a single cable. Each signal is modulated onto a different carrier frequency and the carrier frequencies are separated by guard bands. |
| <u>Time Division Multiplexing (TDM)</u> | Each user gets full bandwidth for a short period of time on a regular basis. The entire channel is dedicated to her one user, but only for a short time. |
| <u>Wavelength Division Multiplexing</u> | This is the same as FDM but applied to fiber, with the difference that here the operating frequency is much higher, |

| Type | Description |
|------|--|
| | actually in the optical range. Due to its extremely high bandwidth, fiber optic has great potential. |

Collision Detection:

| Type | Description |
|---|--|
| <u>Carrier Sense</u> <u>Multiple Access with Collision Detection (CSMA/CD)</u> | In this method, after sending a frame, the station monitors the media to see if the transmission was successful. If successful, the transmission is terminated, otherwise the frame is retransmitted. |
| <u>Carrier Sense</u> <u>Multiple Access with Collision Avoidance (CSMA/CA)</u> | The basic idea behind CSMA/CA is that stations must be able to receive while transmitting in order to detect collisions from different stations. A collision in a wired network nearly doubles the energy of the received signal, allowing stations to detect a potential collision. |
| <u>ALOHA</u> | It was developed for wifi, but can also be used for shared media. Multiple stations can transmit data at the same time, which can lead to collisions and data corruption. |

Network Layer Services:

| Type | Description |
|------------------------------------|---|
| <u>Packetizing</u> | The process of encapsulating data (also called payload) received from upper layers of the network into network layer packets at the source and decapsulating the payload from the network layer packets at the destination is called packetization. |

| Type | Description |
|---|---|
| <u>Routing</u> and Forwarding | These are two other services provided by the network layer. A network has many routes from a source to a destination. The network layer sets some strategies for finding the best possible route. This process is called routing. |

Mode of Communication:

| Types | Description |
|---|--|
| <u>Simplex Mode</u> | In simplex mode, communication is one-way, like one-way. Only one of the two devices on the link can transmit, the other can only receive. Simplex mode allows data to be sent in one direction using the full capacity of the channel. |
| <u>Half-Duplex Mode</u> | In half-duplex mode, each station can both transmit and receive, but not at the same time. When one device transmits, the other device can only receive and vice versa. Half-duplex mode is used when simultaneous communication in both directions is not required. |
| <u>Full-Duplex Mode</u> | <p>In full-duplex mode, both stations can transmit and receive at the same time. In full-duplex mode, signals in one direction share the capacity of the link with signals in the other direction. This sharing can be done in two ways:</p> <ul style="list-style-type: none"> • Either the link must contain two physically separate transmission paths, one for sending and the other for receiving. • Or the capacity is divided between signals traveling in both directions. |

Classes in Computer Networking:

| CLAS S | LEADI NG BITS | NET ID BITS | HOS T ID BITS | NO. OF NETWO RKS | ADDRE SSES PER NETWO RK | START ADDR ESS | END ADDRESS |
|--|---------------|--------------------|-----------------------|-------------------------|--------------------------|----------------|---------------------|
| <u>CLA SS A</u> | 0 | 8 | 24 | 27 (128) | 24 2 (16,777, 216) | 0.0.0.0 | 127.255.25 5.255 |
| <u>CLA SS B</u> | 10 | 16 | 16 | 14 2 (16,384) | 16 2 (65,536) | 128.0. 0.0 | 191.255.25 5.255 |
| <u>CLA SS C</u> | 110 | 24 | 8 | 21 2 (2,097,1 52) | 8 2 (256) | 192.0. 0.0 | 223.255.25 5.255 |
| <u>CLA SS D</u> | 1110 | NOT DEFI NED | NOT DEFI NED | NOT DEFINED | NOT DEFINED | 224.0. 0.0 | 239.255.25 5.255 |
| <u>CLA SS E</u> | 1111 | NOT DEFI NED | NOT DEFI NED | NOT DEFINED | NOT DEFINED | 240.0. 0.0 | 255.255.25 5.255 |
| Subnet Address or Subnet ID Using/16Prefix | | | 1st Usable IP Address | | Last usable IP Address | | Broadcast Address |
| 120.0.0.0/24 | | | 120.0.0.1 | | 120.0.255.254 | | 120.0.255.255 |

| Subnet Address or Subnet ID Using/16Prefix | 1st Usable IP Address | Last usable IP Address | Broadcast Address |
|---|------------------------------|-------------------------------|--------------------------|
| 120.1.0.0/24 | 120.1.0.1 | 120.1.255.254 | 120.1.255.255 |
| 120.2.0.0/24 | 120.2.0.1 | 120.2.255.254 | 120.2.255.255 |
| 120.3.0.0/24 | 120.3.0.1 | 120.3.255.254 | 120.3.255.255 |
| 120.100.0.0/24 | 120.100.0.1 | 120.100.255.254 | 120.100.255.255 |
| 120.101.0.0/24 | 120.101.0.1 | 120.101.255.254 | 120.101.255.255 |
| 120.200.0.0/24 | 120.200.0.1 | 120.200.255.254 | 120.200.255.255 |
| 120.201.0.0/24 | 120.201.0.1 | 120.201.255.254 | 120.201.255.255 |
| 120.253.0.0/24 | 120.253.0.1 | 120.253.255.254 | 120.253.255.255 |
| 120.254.0.0/24 | 120.254.0.1 | 120.254.255.254 | 120.254.255.255 |
| 120.255.0.0/24 | 120.255.0.1 | 120.255.255.254 | 120.255.255.255 |

Subnetting:

| Private IP Address with Subnet Mask | | Private IP Range | Private IP Range denoted in CIDR | |
|-------------------------------------|-----------------|--------------------------------|----------------------------------|--------------------------|
| 10.0.0.0 255.0.0.0 | | 10.0.0.0 to 10.255.255.255 | 10.0.0.0/8 | |
| 172.16.0.0 255.240.0.0 | | 172.16.0.0 to 172.31.255.255 | 172.16.0.0/12 | |
| 192.168.0.0 255.255.0.0 | | 192.168.0.0 to 192.168.255.255 | 192.168.0.0/16 | |
| CIDR | SUBNET MASK | WILDCARD MASK | # OF IP ADDRESSES | # OF USABLE IP ADDRESSES |
| /32 | 255.255.255.255 | 0.0.0.0 | 1 | 1 |
| /31 | 255.255.255.254 | 0.0.0.1 | 2 | 2* |
| /30 | 255.255.255.252 | 0.0.0.3 | 4 | 2 |
| /29 | 255.255.255.248 | 0.0.0.7 | 8 | 6 |
| /28 | 255.255.255.240 | 0.0.0.15 | 16 | 14 |
| /27 | 255.255.255.224 | 0.0.0.31 | 32 | 30 |
| /26 | 255.255.255.192 | 0.0.0.63 | 64 | 62 |

| CIDR | SUBNET MASK | WILDCARD MASK | # OF IP ADDRESSES | # OF USABLE IP ADDRESSES |
|------|-----------------|---------------|-------------------|--------------------------|
| /25 | 255.255.255.128 | 0.0.0.127 | 128 | 126 |
| /24 | 255.255.255.0 | 0.0.0.255 | 256 | 254 |
| /23 | 255.255.254.0 | 0.0.1.255 | 512 | 510 |
| /22 | 255.255.252.0 | 0.0.3.255 | 1,024 | 1,022 |
| /21 | 255.255.248.0 | 0.0.7.255 | 2,048 | 2,046 |
| /20 | 255.255.240.0 | 0.0.15.255 | 4,096 | 4,094 |
| /19 | 255.255.224.0 | 0.0.31.255 | 8,192 | 8,190 |
| /18 | 255.255.192.0 | 0.0.63.255 | 16,384 | 16,382 |
| /17 | 255.255.128.0 | 0.0.127.255 | 32,768 | 32,766 |
| /16 | 255.255.0.0 | 0.0.255.255 | 65,536 | 65,534 |
| /15 | 255.254.0.0 | 0.1.255.255 | 131,072 | 131,070 |
| /14 | 255.252.0.0 | 0.3.255.255 | 262,144 | 262,142 |
| /13 | 255.248.0.0 | 0.7.255.255 | 524,288 | 524,286 |

| CIDR | SUBNET MASK | WILDCARD MASK | # OF IP ADDRESSES | # OF USABLE IP ADDRESSES |
|------|-------------|-----------------|-------------------|--------------------------|
| /12 | 255.240.0.0 | 0.15.255.255 | 1,048,576 | 1,048,574 |
| /11 | 255.224.0.0 | 0.31.255.255 | 2,097,152 | 2,097,150 |
| /10 | 255.192.0.0 | 0.63.255.255 | 4,194,304 | 4,194,302 |
| /9 | 255.128.0.0 | 0.127.255.255 | 8,388,608 | 8,388,606 |
| /8 | 255.0.0.0 | 0.255.255.255 | 16,777,216 | 16,777,214 |
| /7 | 254.0.0.0 | 1.255.255.255 | 33,554,432 | 33,554,430 |
| /6 | 252.0.0.0 | 3.255.255.255 | 67,108,864 | 67,108,862 |
| /5 | 248.0.0.0 | 7.255.255.255 | 134,217,728 | 134,217,726 |
| /4 | 240.0.0.0 | 15.255.255.255 | 268,435,456 | 268,435,454 |
| /3 | 224.0.0.0 | 31.255.255.255 | 536,870,912 | 536,870,910 |
| /2 | 192.0.0.0 | 63.255.255.255 | 1,073,741,824 | 1,073,741,822 |
| /1 | 128.0.0.0 | 127.255.255.255 | 2,147,483,648 | 2,147,483,646 |
| /0 | 0.0.0.0 | 255.255.255.255 | 4,294,967,296 | 4,294,967,294 |

Methods of Network Security:

| Method | Description |
|--|--|
| <u>Authentication</u> | Verify a user's identity, usually by asking them to enter a password or biometric identifier. |
| <u>Encryption</u> | Encrypt data with a key, that is, the same key is required to decrypt the data. This is how HTTPS works. |
| <u>Firewalls</u> | Protect the network from unauthorized access. |
| <u>MAC Address Filtering</u> | Allow devices to access or be prevented from accessing the network based on their physical address embedded in the device's network adapter. |

<https://www.geeksforgeeks.org/computer-network-cheat-sheet/>

Wireshark Cheat Sheet

Examples to Understand the Power of Wireshark

Wireshark can be useful for many different tasks, whether you are a network engineer, security professional or system administrator. Here are a few example use cases:

Troubleshooting Network Connectivity

- Visually understand packet loss
- Review TCP retransmission
- Graph high latency packet responses

Examination of Application Layer Sessions (even when encrypted by SSL/TLS see below)

- View full HTTP session, seeing all headers and data for both requests and responses
- View Telnet sessions, see passwords, commands entered and responses
- View SMTP or POP3 traffic, reading emails off the wire

Troubleshoot DHCP issues with packet level data

- Examine DHCP client broadcast

- DHCP offer with address and options
- Client requests for offered address
- Ack of server acknowledging the request

Extract files from HTTP sessions

- [Export objects](#) from HTTP such as javascript, images, or even executables.

Extract file from SMB sessions

- Similar to the HTTP export option but able to extract files transferred over SMB, the ever present Microsoft File Sharing protocol.

Detect and Examination of Malware

- Detect anomalous behaviour that could indicate malware
- Search for unusual domains or IP address endpoints
- Use IO graphs to discover regular connections (beacons) to command and control servers
- Filter out the "normal" and find the unusual
- Extract large DNS responses and other oddness which may indicate malware

Examination of Port Scans and Other Vulnerability Scan types

- Understand what network traffic the vulnerability scanner is sending
- Troubleshoot vulnerability checks to understand false positives and false negatives

These examples only scratch the surface of the possibilities. Continue reading through the tutorial and start getting more from this powerful tool.

Installation of Wireshark

Wireshark will run on a variety of operating systems and is not difficult to get up and running. We will touch on Ubuntu Linux, Centos and Windows.

Install on Ubuntu or Debian

```
~# apt-get update

~# apt-get install wireshark tshark
```

Getting the latest version of Wireshark has a number of benefits. Many new features are released with major updates such as new protocol parsing and other features. There is

a PPA available for Ubuntu, add the repository and update packages to ensure you are getting a more recent release.

```
~# add-apt-repository ppa:wireshark-dev/stable  
~# apt-get update
```

Install on Fedora or CentOS

```
~# yum install wireshark-gnome
```

Install on Windows

Head over to the [Wireshark Download](#) page, grab the installation executable and run it to install. Pretty straight forward, you will also be installing a packet capture driver. This allows the network card to enter promiscuous mode.

Getting Started with Filters

After running an initial capture you will see the standard layout and the packet details that can be viewed through the interface.

When you have captured an HTTP session, stop the capture and try playing with a few basic filters and the **Analyze | Follow | HTTP Stream** options.

The filters are easy to read and self-explanatory. You enter these expressions into the filter bar (or on the command line if using tshark). A primary benefit of the filters is to remove the noise (traffic you don't want to see). As seen here, you can filter on MAC address, IP address, Subnet or protocol. The easiest filter is to type `http` into the filter bar. The results will now only show HTTP (tcp port 80) traffic.

IP Address Filter Examples

```
ip.addr == 192.168.0.5  
  
!(ip.addr == 192.168.0.0/24)
```

Protocol Filter Examples

```
tcp
```

```

udp

tcp.port == 80 || udp.port == 80

http

not arp and not (udp.port == 53)

```

Try generating a filter combination that shows all non HTTP and HTTPS traffic leaving your local system that is not destined for the local network. This is a good way to find software (malware even) that is communicating with the Internet using unusual protocols.

Follow the White Rabbit Stream

Once you have several packets showing HTTP, select one and then select **Analyze | Follow | HTTP Stream** from the drop-down menu. This will show you an assembled HTTP session. In this new window, you see the HTTP request from the browser and HTTP response from the web server. Goal! You are now winning at Wireshark. Continue reading our Wireshark Tutorial for more advanced tips.

```

Wireshark - Follow HTTP Stream (tcp.stream eq 5) - wireshark_enp7s0_20180519172906_LtkX06
POST /phpMyAdmin/index.php HTTP/1.1
Host: 192.168.1.92
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.1.92/phpMyAdmin/
Content-Type: application/x-www-form-urlencoded
Content-Length: 273
Cookie: phpMyAdmin=0b460e3d045e07a4c7b19adc61db43f52a6d7d47; pma_lang=en-utf-8; pma_charset=utf-8; pma_theme=original; pma_fontsize=82%;5
Connection: keep-alive
Upgrade-Insecure-Requests: 1

phpMyAdmin=0b460e3d045e07a4c7b19adc61db43f52a6d7d47&phpMyAdmin=0b460e3d045e07a4c7b19adc61db43f52a6d7d47&pma_username=admin&pma_password=admin&server=
phpMyAdmin=0b460e3d045e07a4c7b19adc61db43f52a6d7d47&lang=en-utf-8&convcharset=utf-8&token=c6fa291545bf559592b60032365940HTTP/1.1 302 Found
Date: Sat, 19 May 2018 07:29:15 GMT
Server: Apache/2.2.8 (Ubuntu) DAV/2
X-Powered-By: PHP/5.2.4-2ubuntu5.10
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: private, max-age=10800, pre-check=10800
Set-Cookie: pma_fontsize=deleted; expires=Fri, 19-May-2017 07:29:14 GMT; path=/phpMyAdmin/

```

http request from browser in red

and look what we find in the POST body

http response from web server in blue

Resolve DNS in Wireshark

By default, Wireshark won't resolve the network address that it is displaying in the console. Only showing IP addresses, by changing an option in the preferences, you can enable the resolution of IP addresses to network names. This will slow down the display of packets, as it also does when using `tcpdump`, due to the DNS resolution that has to take place. It is important to understand if you are doing a live capture, the DNS requests from your Wireshark host will be additional traffic that you might be capturing.

Edit | Preferences | Name Resolution | Enable Network Name Resolution

Tshark for the Command Line

If you haven't had a play with `tshark`, take a look at our [tshark tutorial and filter examples](#). This program is often overlooked but is a great way to capture application layer sessions on a remote system. The advantage over `tcpdump` is you can capture and view application

layer sessions on the fly, as the protocol decoders included in Wireshark are also available to `tshark`.

Build Firewall Rules

A quick way to generate command line firewall rules, this can save a few minutes Googling for different firewall syntax. Select a rule, and head up to the Tools | Firewall ACL Rules. Different firewall products such as Cisco IOS (standard and extended), `ipfilter`, `ipfw`, `iptables`, `pf` and even Windows firewall using `netsh`.

```
# IPv4 source address.
iptables --append INPUT --in-interface eth0 --source 66.175.211.5/32 --jump DROP

# IPv4 destination address.
iptables --append INPUT --in-interface eth0 --source 66.175.214.247/32 --jump DROP

# Source port.
iptables --append INPUT --in-interface eth0 --protocol udp --source-port 53 --jump DROP

# Destination port.
iptables --append INPUT --in-interface eth0 --protocol udp --source-port 44407 --jump DROP

# IPv4 source address and port.
iptables --append INPUT --in-interface eth0 --protocol udp --source 66.175.211.5/32 --source-port 53 --jump DROP

# IPv4 destination address and port.
iptables --append INPUT --in-interface eth0 --protocol udp --source 66.175.214.247/32 --source-port 44407 --jump DROP

# MAC source address.
iptables --append INPUT --in-interface eth0 --mac-source 84:78:ac:57:a8:41 --jump DROP

# MAC destination address.
iptables --append INPUT --in-interface eth0 --mac-source f0:2e:01:6c:d5:e0 --jump DROP
```

Generate Firewall Rules based on captured packet

Create rules for ▾

Wireshark GeoIP Mapping

As long as Wireshark has been compiled with GeoIP support and you have the [Free Maxmind databases](#) available, you can resolve IP addresses to locations. Look at About | Wireshark to see what has been compiled with the version you are using. If GeoIP is listed, ensure you have the GeoLite City, Country, and ASNum databases in a directory on your system running Wireshark. Point to the location of the databases in Edit | Preferences | Name Resolution.

Test it by loading a capture and selecting Statistics | Endpoints | IPv4. The columns on the right show the location and ASN information for the IP address.

| Endpoints - test4 | | | | | | | | | | | | | | |
|-------------------|-----------|--------|------------|----------|-------------|----------|-------------------|-------------------------------------|--|-----------|------------|-----------|--|--|
| | IPv4 · 16 | | IPv6 · 5 | | TCP · 65653 | | UDP · 79 | | | | | | | |
| | packets | bytes | tx packets | tx bytes | rx packets | rx bytes | city | country | as number | | latitude | longitude | | |
| 1.181 | 131.402 | 7643 k | 141 | 27 k | 131.261 | 7615 k | Boynton, VA | United States | AS8075 Microsoft Corporation | | 36.664799 | -78.3714 | | |
| 65.146 | 64 | 21 k | 31 | 5810 | 33 | 15 k | Mountain View, CA | United States | AS15169 Google LLC | | 37.419201 | -122.057 | | |
| 1.5 | 12 | 1440 | 6 | 954 | 6 | 486 | Newark, NJ | United States | AS63949 Linode, LLC | | 40.735699 | -74.1724 | | |
| 2.5 | 148 | 18 k | 74 | 11 k | 74 | 6436 | Newark, NJ | United States | AS63949 Linode, LLC | | 40.735699 | -74.1724 | | |
| 11.5 | 34 | 3621 | 17 | 2409 | 17 | 1212 | Cedar Knolls, NJ | United States | AS63949 Linode, LLC | | 40.822102 | -74.4563 | | |
| 14.247 | 131.8000 | 7718 k | 131.459 | 7658 k | 341 | 59 k | Cedar Knolls, NJ | United States | AS63949 Linode, LLC | | 40.822102 | -74.4563 | | |
| 33.4 | 16 | 1548 | 8 | 862 | 8 | 686 | Newark, NJ | United States | AS63949 Linode, LLC | | 40.735699 | -74.1724 | | |
| 78.32 | 20 | 2144 | 10 | 964 | 10 | 1180 | Clifton, NJ | United States | AS14061 DigitalOcean, LLC | | 40.832600 | -74.1306 | | |
| 56.78 | 2 | 114 | 1 | 60 | 1 | 54 | Rebel, 10 | Korea, Republic of | AS36351 SoftLayer Technologies Inc. | | 36.000000 | 127.350 | | |
| 548.163 | 2 | 114 | 1 | 1 | 1 | 1 | Beon, 13 | Korea, Republic of | AS4766 Korea Telecom | | 37.739202 | 127.442 | | |
| 8.74 | 2 | 114 | 1 | 1 | 1 | 1 | United States | AS36351 SoftLayer Technologies Inc. | | 37.750999 | -97.8219 | | | |
| 6.46 | 1 | 90 | 0 | 0 | 0 | 0 | Hunt Holly, NJ | United States | AS701 MCI Communications Services, Inc. d/b/a Verizon Business | | 39.994301 | -74.7918 | | |
| 107.101 | 26 | 3004 | 15 | 656 | 9 | 512 | Cedar Knolls, NJ | United States | AS4739 Internode Pty Ltd | | -35.169601 | 149.128 | | |
| 69.4 | 12 | 1168 | 6 | 656 | 6 | 512 | Cedar Knolls, NJ | United States | AS63949 Linode, LLC | | 40.822102 | -74.4563 | | |

GeoIP data for the IP address Endpoints

A further function of the GeoIP feature is to filter traffic based on location using the `ip.geoip` display filter.

For example, use this filter to exclude traffic from an ASN. ASN 63949 is the Linode block, so the filter now displays only IP traffic not coming from this netblock.

```
ip and not ip.geoip.asnum == 63949
```

Of course, you can apply the [same filter to city and country based queries](#). Doing this removes noise from your capture display and allows you to focus on the packets you care about.

Decrypt SSL/TLS sessions

One way of decrypting SSL/TLS sessions is to use the Private Key from the server that is being connected to by the client. Using this key, you can decrypt the session and view the protocol under the SSL/TLS layer. For example, in a browser session you could see the plain text HTTP.

You are not always going to have access to the servers private key. Hence, there is another option for easily viewing the browser SSL/TLS traffic from your local system. If Firefox or Chrome are loaded using a special environment variable, the individual SSL/TLS session symmetric keys will be logged to a file that Wireshark can read. With these keys, Wireshark can show you the session fully decrypted for the win!

1. Configure the Environment Variable

Linux / Mac

```
export SSLKEYLOGFILE=~/sslkeylogfile.log
```

Windows

Under advanced system settings, select [Environment Variables](#) and add the variable name `SSLKEYLOGFILE` with the variable value as the path to where you want the file saved.

2. Configure Wireshark

From the drop-down menu select Edit | Preferences | Protocols | SSL | (Pre)-Master-Secret Log Filename -- Browse to the log file you placed in your environment variable.

Start a capturing on your local system.

3. Restart Firefox or Chrome

After browsing to a HTTPS site. The log file should start to increase in size as it logs the symmetric session keys.

Review the Wireshark session previously started. You should see something resembling the image below showing the decrypted sessions. Take note of the decrypted packets in the tab in the bottom pane.

tcp.stream eq 16

| No. | Time | Source | Destination | Protocol | Length | Info |
|-----|-------------|---------------|---------------|----------|--------|--|
| 502 | 9.211593439 | 192.168.1.10 | 151.101.1.198 | TCP | 74 | 53502 - 443 [SYN] Seq=0 Win=29200 |
| 529 | 9.221223818 | 151.101.1.198 | 192.168.1.10 | TCP | 74 | 443 - 53502 [SYN, ACK] Seq=0 Ack=1 Win=29200 |
| 530 | 9.221249199 | 192.168.1.10 | 151.101.1.198 | TCP | 66 | 53502 - 443 [ACK] Seq=1 Ack=1 Win=29200 |
| 557 | 9.235197935 | 192.168.1.10 | 151.101.1.198 | TLSv1.2 | 583 | Client Hello |
| 589 | 9.243759343 | 151.101.1.198 | 192.168.1.10 | TCP | 96 | 443 - 53502 [ACK] Seq=1 Ack=518 Win=29200 |
| 601 | 9.245710958 | 151.101.1.198 | 192.168.1.10 | TLSv1.2 | 2946 | Server Hello, Certificate |
| 602 | 9.245710963 | 192.168.1.10 | 151.101.1.198 | TCP | 66 | 53502 - 443 [ACK] Seq=518 Ack=288 |

Frame 753: 451 bytes on wire (3608 bits), 451 bytes captured (3608 bits) on interface 0

► Ethernet II, Src: Giga-Byt_cf:a6:3a (40:8d:5c:cf:a6:3a), Dst: AsustekC_5d:a7:c8 (34:97:f6:5d:a7:c8)

► Internet Protocol Version 4, Src: 192.168.1.10, Dst: 151.101.1.198

► Transmission Control Protocol, Src Port: 53502, Dst Port: 443, Seq: 3491, Ack: 4785, Len: 385

► [3 Reassembled TCP Segments (3265 bytes): #751(1440), #752(1440), #753(385)]

Frame (451 bytes) Reassembled TCP (3265 bytes) Decrypted SSL (3236 bytes)

Frame (frame), 451 bytes

Wireshark - Follow SSL Stream (tcp.stream eq 16) - wireshark_enp7s0 20180519193236_CArSKy

GET /morphus.cnet.1591.js HTTP/1.1
Host: mtrx.go.sonobi.com
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: */*
Accept-Language: en-GB,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: https://www.cnet.com/
Cookie: __uin_t1=10664471760336337415; __uis=b0cccf14-46c9-11e8-aef0-000acd2a; lotame-131957:";"sonobi-lotame-146534:";"sonobi-lotame-131797:";"sonobi-lotame-3807:";"sonobi-lotame-131961:";"sonobi-lotame-131986:";"sonobi-lotame-393450:";"sonobi-lotame-131987:";"sonobi-lotame-294994:";"sonobi-lotame-381741:";"sonobi-lotame-386297:";"sonobi-lotame-374849:";"sonobi-lotame-336866:";"sonobi-lotame-3964; lotame-3822:";"sonobi-lotame-131799:";"sonobi-lotame-362396:";"sonobi-lotame-131982:";"sonobi-lotame-393451; lotame-3805:";"sonobi-lotame-381217:";"sonobi-lotame-131796:";"sonobi-lotame-371243:";"sonobi-lotame-294996; lotame-2924:";"sonobi-lotame-187851:";"sonobi-lotame-131958:";"sonobi-lotame-390462:";"sonobi-lotame-131979; lotame-264952:";"sonobi-lotame-330565:";"sonobi-lotame-198548:";"sonobi-lotame-381733:";"sonobi-lotame-3934; lotame-264954:";"sonobi-lotame-287829:";"sonobi-lotame-390452:";"sonobi-lotame-264951:";"87fb69345d-lotame-381739:";"sonobi-lotame-386302:";"sonobi-lotame-131800:";"sonobi-lotame-136189:";"sonobi-lotame-3904; lotame-125478:";"sonobi-lotame-263807:";"sonobi-lotame-131978:";"sonobi-lotame-336871:";"sonobi-lotame-1317; lotame-382709:";"sonobi-lotame-381214:";"sonobi-lotame-131981:";"sonobi-lotame-393448:";"sonobi-lotame-2185; lotame-217028:";"87fb69345d-lotame-65413:";"sonobi-lotame-132003:";"sonobi-lotame-382408:"]; __uin_bw=3bc18; __uin_td=8bf1eccb-b325-4cc8-a37b-97903420d29; __uin_pp=Szc0YEkPeK3y; __uin_rx=RX-aafdfb9b-612b-47bb-8123-de922fba4e01a61d0; __uin_sv=30c2a364f0baedd595e1fa0c061c3133; __uin_gm=ChU17YDjrvM4fdKEJSD1dyR-db-tQESKwgxEidSWC1hY; __uin_tb=ccebb50d-feda-4094-ad5d-e4a0edf3e680; __uin_pl=niwTnu2BqfqMlqJxWnZLQU0bpgtlwKotafpJ1JIUYC0=; Connection: keep-alive

HTTP/1.1 200 OK
x-amz-id-2: RDmlNgaQs40BzX41fDSjJS08eqyPzRvozN0Phc9o23rDwmFScZFIUE73SqohkgU0IW+ColyH0=
x-amz-request-id: 0A0049E482344B5D
Last-Modified: Thu, 03 May 2018 17:31:26 GMT

Another way to view the session is to use the analysis drop down and follow the stream. If the session has successfully been decrypted you will see the option for SSL under Stream.

Analysis | Follow | Stream | SSL

Use caution when logging these keys and pcaps. Someone with access to the key log file and your pcap might very well find your passwords and authentication cookies within the pcap.

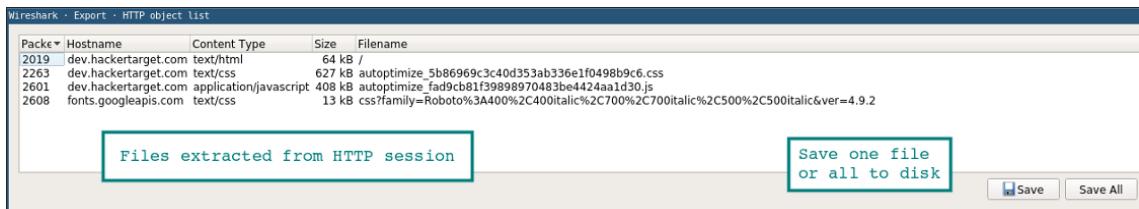
Another option for getting at the underlying HTTP traffic is using [Burp Suite](#) with its CA loaded in your browser. In this case, the proxy decrypts the connection on the client side and then establishes a new SSL/TLS session to the server. There are many ways to man in the middle (mitm) yourself, these are two of the most straightforward.

Extract files from PCAP using Export (HTTP or SMB)

It is quite easy to extract files from a Wireshark capture using the export option.

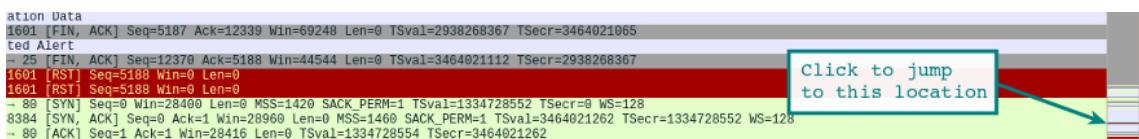
File | Export Objects | HTTP

The new Window will show any files that were found. In this new Window you can save the individual files or save them all to a folder. A similar method can be used to extract files from SMB sessions. This is the Microsoft Server Message Block protocol that allows Windows File Sharing.



Right Hand Status Bar

Quickly jump to packets based on the color of the main display. For instance, to find Red - Errors see the red line noted in the right hand side status bar and jump to that location with a click.



Sample PCAP's are readily available

You may be getting started with Wireshark and looking for interesting packet captures to explore, the [Wireshark Samples](#) page is a great place to start. Due to the fact, there are enough sample protocols to keep you busy for months and several worm / exploit samples for those digging into Network Security Monitoring.

Setting up your Environment

A handy tip is to remember the default console is highly configurable. You may add or remove columns, even adding something as simple as a UTC time column. Which might be immediately useful if you are looking at historical pcaps.

The columns can be configured by going to Edit | Preferences | Appearance | Columns. In this area, you can also change the layout, font, and colors.

This video has good configuration tips for the environment. Including troubleshooting tips and configurations for identifying issues through TCP sequence numbers.

capinfos

A handy command line tool that comes packaged with Wireshark is the `capinfos` binary. This command will produce a summary of a `pcap` with statistics, start / finish times and other details.

Run it as below or use the table option `-T` to produce tab separated output that can be imported into a spreadsheet or parsed on the command line.

```
test@ubuntu:~$ capinfos test.pcap

File name:          test.pcap

File type:          Wireshark/tcpdump/... - pcap

File encapsulation: Ethernet

File timestamp precision: microseconds (6)

Packet size limit:   file hdr: 262144 bytes

Number of packets:  341 k

File size:          449 MB

Data size:          444 MB

Capture duration:   3673.413779 seconds

First packet time:  2018-12-01 11:26:53.521929

Last packet time:   2018-12-01 12:28:06.935708

Data byte rate:     120 kBps

Data bit rate:      967 kbps

Average packet size: 1300.72 bytes

Average packet rate: 93 packets/s
```

```
SHA256:  
989388128d676c329ccdbdec4ed221ab8ecffad81910a16f473ec2c2f54c5d6e  
  
RIPEMD160:  
0742b6bbc79735e57904008d6064cce7eb95abc9  
  
SHA1:  
d725b389bea044d6520470c8dab0de1598b01d89  
  
Strict time order: True  
  
Number of interfaces in file: 1
```

<https://hackertarget.com/wireshark-tutorial-and-cheat-sheet/>

Default Columns In a Packet Capture Output

When analyzing network traffic with a packet capture tool such as Wireshark, the output typically includes a list of columns that provide important information about each captured packet. Here are the default columns in Wireshark, along with their names and descriptions:

| Column Name | Description |
|-------------|---|
| No. | The packet number in the capture file. |
| Time | The time the packet was captured. |
| Source | The source IP address of the packet. |
| Destination | The destination IP address of the packet. |
| Protocol | The protocol used by the packet (e.g., TCP, UDP, ICMP). |
| Length | The length of the packet in bytes. |
| Info | A summary of the packet contents, such as the HTTP request method or response code. |

These columns provide essential information that can help you analyze network traffic and troubleshoot network issues. For example, the source and destination columns can help you identify where the traffic is coming from and where it's going. The protocol column can help you determine which

application or service is generating the traffic, while the length column can help you identify unusually large or small packets that may indicate a problem. Finally, the info column can help you quickly identify important packets, such as those containing HTTP requests or responses.

It's worth noting that you can customize the columns displayed in Wireshark to suit your needs. For example, you can add columns to display additional information such as the TCP sequence number or the DNS query name. To add or remove columns, simply right-click on the column header and select "Column Preferences". From there, you can select the columns you want to display and adjust their order and width as needed.

Logical Operators

Logical operators are used in various programming languages and tools to combine two or more conditions or expressions to form a single, more complex condition. They are frequently used in network analysis and troubleshooting tools, such as packet capture tools, to filter and search for specific packets or network traffic. Here are the three main logical operators and their descriptions:

| Operator | Name | Description |
|----------|------|---|
| && | AND | Evaluates to true if and only if both conditions are true. For example, in a packet capture filter, the expression ip.src == 192.168.1.1 && tcp.port == 80 would match only packets with a source IP address of 192.168.1.1 AND a TCP destination port of 80. |
| | OR | Evaluates to true if either or both conditions are true. For example, in a packet capture filter, the expression `ip.src == 192.168.1.1 |
| ! | NOT | Inverts the result of the condition. For example, in a packet capture filter, the expression !tcp.port == 80 would match packets that do NOT have a TCP destination port of 80. |

These logical operators can be combined to create more complex conditions that are used to filter and search for specific packets or network traffic. By using these operators effectively, network analysts and troubleshooters can quickly isolate and diagnose network problems.

Filtering Packets (Display Filters)

Packet filtering, also known as display filtering, is the process of selecting specific packets from a captured network traffic for further analysis. Display filters in packet capture tools such as Wireshark provide a way to filter packets based on specific criteria such as protocol, IP address, port number, etc. Here are some common display filters along with their descriptions and examples:

| Filter | Operator | Description | Example |
|--------------|-----------------|--|------------------------------|
| ip.addr | == | Matches packets where the IP address matches the specified value. This can be used to filter packets based on either the source or destination IP address. | ip.addr == 192.168.1.1 |
| tcp.port | == | Matches packets where the TCP port number matches the specified value. This can be used to filter packets based on either the source or destination port. | tcp.port == 80 |
| udp.port | == | Matches packets where the UDP port number matches the specified value. This can be used to filter packets based on either the source or destination port. | udp.port == 53 |
| http | | Matches packets where the protocol is HTTP. This can be used to filter HTTP traffic, which can be useful for analyzing web traffic. | http |
| dns | | Matches packets where the protocol is DNS. This can be used to filter DNS traffic, which can be useful for analyzing DNS queries and responses. | dns |
| icmp | | Matches packets where the protocol is ICMP. This can be used to filter ICMP traffic, which can be useful for analyzing ping requests and responses. | icmp |
| frame.number | <, >, <=, >= | Matches packets with a frame number less than, greater than, less than or equal to, or greater than or equal to the specified value. | frame.number >= 100 |
| frame.time | <, >, <=, >= | Matches packets with a frame time less than, greater than, less than or equal to, or greater than or equal to the specified value. | frame.time >= 03-10 10:00:00 |

By using these filters effectively, network analysts and troubleshooters can quickly isolate and diagnose network problems. These are just a few examples of the many display filters available in packet capture tools such as Wireshark. Understanding these filters and how to use them can greatly improve the efficiency and accuracy of network analysis and troubleshooting.

Filter Types

Packet filtering is a crucial aspect of network analysis and troubleshooting. It allows network analysts to selectively view and analyze network traffic based on specific criteria. Packet capture tools such as Wireshark provide various types of filters for packet filtering. Here are some common types of filters along with their names and descriptions:

| Filter Type | Name | Description |
|---------------------|------------------------------|---|
| Capture Filter | BPF (Berkeley Packet Filter) | A low-level filter applied to a live capture or when saving a capture. This filter is written in BPF syntax and is used to capture only the packets of interest. |
| Display Filter | Wireshark filter | A high-level filter applied to an already captured packet capture. This filter is written in Wireshark's display filter syntax and is used to selectively view the packets of interest. |
| Protocol Filter | Protocol filter | A filter applied to isolate packets that belong to a specific protocol family, such as IP, TCP, UDP, DNS, etc. |
| Time Filter | Time-based filter | A filter applied to isolate packets that occurred within a specific range or at a specific time interval. |
| Endpoint Filter | Endpoint filter | A filter applied to isolate packets that belong to a specific endpoint, such as a specific IP address or MAC address. |
| Conversation Filter | Conversation filter | A filter applied to isolate packets that belong to a specific conversation between two endpoints, such as a specific IP address or port. |
| Expert Info Filter | Expert Info filter | A filter applied to isolate packets with specific expert info messages generated by Wireshark, such as warnings or errors. |

By using these filters effectively, network analysts and troubleshooters can quickly isolate and diagnose network problems. These are just a few examples of the many filter types available in packet capture tools such as Wireshark.

Understanding these filters and how to use them can greatly improve the efficiency and accuracy of network analysis and troubleshooting.

Wireshark Capturing Modes

Wireshark, like many other packet capture tools, supports various capturing modes that provide different levels of access and control over the captured traffic. Here are some common Wireshark capturing modes along with their names and descriptions:

| Capturing Mode | Name | Description |
|----------------------|----------------------|--|
| Promiscuous Mode | Promiscuous Mode | Captures all network traffic seen by the network interface card (NIC), including traffic not intended for the host machine. This mode is useful for capturing all traffic on a network segment or for analyzing switch or router behavior. |
| Non-Promiscuous Mode | Non-Promiscuous Mode | Captures only traffic intended for the host machine, such as traffic addressed to its MAC address. This mode is useful for capturing traffic from a specific machine and can help reduce the amount of captured traffic. |
| Monitor Mode | Monitor Mode | Captures all wireless network traffic on a specific wireless channel. This mode is used for wireless network analysis and troubleshooting. |
| Remote Capture | Remote Capture | Captures network traffic on a remote machine using the Wireshark Remote Packet Capture Protocol (RPCAP). This mode is useful for capturing traffic on machines that are not physically accessible or for capturing traffic on remote networks. |
| File-based Capture | File-based Capture | Captures network traffic to a file for later analysis. This mode is useful for capturing large amounts of traffic over a longer period or for capturing traffic on systems with limited resources. |

By using these capturing modes effectively, network analysts and troubleshooters can gain better control over the captured traffic and reduce the amount of irrelevant traffic in the capture file. Understanding these modes and how to use them can greatly improve the efficiency and accuracy of network analysis and troubleshooting.

Miscellaneous

Here are some miscellaneous features and capabilities of Wireshark along with their names and descriptions:

| Feature | Name | Description |
|-------------------------|-------------------------|---|
| Protocol Decode | Protocol Decode | Wireshark can decode and display the contents of a captured packet according to the protocol specification. This feature allows network analysts to view the details of a specific protocol and can help identify protocol-specific issues. |
| Conversation Statistics | Conversation Statistics | Wireshark can display conversation statistics between two endpoints including the number of packets, bytes, and average packet size. This feature is useful for analyzing the performance of specific connections for identifying potential network issues. |
| Coloring Rules | Coloring Rules | Wireshark can apply color-coded highlighting to packets based on specific criteria, such as protocol or source/destination IP address. This feature allows network analysts to quickly identify relevant packets in a large capture file. |
| Follow TCP Stream | Follow TCP Stream | Wireshark can reconstruct the contents of a TCP stream and display it in a separate window. This feature is useful for analyzing the contents of a specific connection and can help identify application-layer issues. |
| Exporting Data | Exporting Data | Wireshark can export captured packets to various formats, including CSV, JSON, and XML. This feature allows network analysts to manipulate captured data in external tools or to share it with others who do not have access to Wireshark. |
| Plugins | Plugins | Wireshark supports various plugins that can extend its functionality and capabilities. These plugins can be developed by third parties or by the Wireshark community and can be used to enhance specific features or add new ones. |

By using these features and capabilities effectively, network analysts and troubleshooters can gain greater insight into network behavior and diagnose network problems more efficiently. Understanding these features and how to use them can greatly improve the efficiency and accuracy of network analysis and troubleshooting.

Capture Filter Syntax

A capture filter is a filter used to limit the amount of traffic captured by Wireshark when capturing packets in real-time. Here are some common capture filter syntax elements along with their names and descriptions:

| Syntax Element | Name | Description |
|----------------------------|--------------|---|
| Protocol | Protocol | Filters packets based on the protocol type, such as TCP, UDP, or ICMP. |
| Source/Destination Address | Host | Filters packets based on the source or destination IP address. |
| Port Number | Port | Filters packets based on the port number, such as TCP port 80 for HTTP traffic. |
| Network Address Range | Net | Filters packets based on a range of IP addresses, such as 192.168.0.0/24 for all hosts in the 192.168.0.0 subnet. |
| Logical Operators | and, or, not | Used to combine multiple filter expressions or to negate a filter expression. |
| Expression Grouping | () | Used to group multiple filter expressions to control the order of evaluation. |
| Primitive | Primitive | Specifies a basic condition to filter packets, such as src or dst for source or destination IP address, respectively. |

By using these syntax elements effectively, network analysts and troubleshooters can create capture filters that target specific traffic and reduce the amount of irrelevant traffic captured. Understanding these syntax elements and how to use them can greatly improve the efficiency and accuracy of network analysis and troubleshooting.

Display Filter Syntax

A display filter is a filter used to selectively display packets from a packet capture file. Here are some common display filter syntax elements along with their names and descriptions:

| Syntax Element | Name | Description |
|----------------------------|---------------------|--|
| Protocol | Protocol | Filters packets based on the protocol type, such as TCP, UDP, etc. |
| Source/Destination Address | ip.addr | Filters packets based on the source or destination IP address. |
| Port Number | tcp.port, udp.port | Filters packets based on the port number, such as tcp.port == 80 for HTTP traffic. |
| Network Address Range | ip.src, ip.dst, net | Filters packets based on a range of IP addresses, such as ip.src == 192.168.0.0/24 for all packets with a source IP address in the 192.168.0.0 subnet. |
| Time | frame.time | Filters packets based on the time of capture, such as frame.time > "2022-01-01 00:00:00" to display packets captured after a specific date and time. |
| Logical Operators | and, or, not | Used to combine multiple filter expressions or to negate a filter expression. |
| Expression Grouping | () | Used to group multiple filter expressions to control the order of evaluation. |
| Field Comparison | ==, !=, <, > | Used to compare fields within a packet, such as tcp.len > 100 to display packets with a TCP payload length greater than 100 bytes. |

By using these syntax elements effectively, network analysts and troubleshooters can create display filters that selectively display packets and highlight relevant information. Understanding these syntax elements and how to use them can greatly improve the efficiency and accuracy of network analysis and troubleshooting.

Keyboard Shortcuts – Main Display Window

Here are some common keyboard shortcuts for the main display window in Wireshark:

| Shortcut | Description |
|------------------|--|
| Ctrl + E | Expand all packets |
| Ctrl + F | Open the display filter dialog |
| Ctrl + G | Find the next packet matching the current display filter |
| Ctrl + L | Open the capture filter dialog |
| Ctrl + R | Apply or reapply the current display filter |
| Ctrl + Shift + R | Clear the current display filter |
| Ctrl + Shift + T | Toggle the time display format between absolute and relative |
| Ctrl + Shift + X | Exit Wireshark |
| Ctrl + + | Increase the font size |
| Ctrl + - | Decrease the font size |
| Ctrl + 1-9 | Toggle the display of specific packet details columns |
| Tab | Switch focus between the packet list and packet details panes |
| Shift + Tab | Switch focus in reverse order |
| Enter | Expand or collapse a packet in the packet list |
| Spacebar | Start or stop packet capture |
| F1 | Open the Wireshark help documentation |
| F3 | Find the next occurrence of a search term in the packet details pane |
| F11 | Toggle full-screen mode |

By using these keyboard shortcuts, network analysts and troubleshooters can navigate through packets, filter traffic, and customize the display quickly and efficiently.

Protocols – Values

ether, fddi, ip, arp, rarp, decnet, lat, sca, moprc, mopdl, tcp and udp

Common Filtering Commands

Common Filtering Commands

| USAGE | FILTER SYNTAX |
|--------------------------------------|---|
| Wireshark Filter by IP | ip.add == 10.10.50.1 |
| Filter by Destination IP | ip.dest == 10.10.50.1 |
| Filter by Source IP | ip.src == 10.10.50.1 |
| Filter by IP range | ip.addr >= 10.10.50.1 and ip.addr <=10.10.50.100 |
| Filter by Multiple Ips | ip.addr == 10.10.50.1 and ip.addr == 10.10.50.100 |
| Filter out IP address | ! (ip.addr == 10.10.50.1) |
| Filter subnet | ip.addr == 10.10.50.1/24 |
| Filter by port | tcp.port == 25 |
| Filter by destination port | tcp.dstport == 23 |
| Filter by ip address and port | ip.addr == 10.10.50.1 and Tcp.port == 25 |
| Filter by URL | http.host == "host name" |
| Filter by time stamp | frame.time >= "June 02, 2019 18:04:00" |
| Filter SYN flag | Tcp.flags.syn == 1 and tcp.flags.ack ==0 |

| USAGE | FILTER SYNTAX |
|-----------------------------------|-------------------------------|
| Wireshark Beacon Filter | wlan.fc.type_subtype = 0x08 |
| Wireshark broadcast filter | eth.dst == ff:ff:ff:ff:ff:ff |
| Wireshark multicast filter | (eth.dst[0] & 1) |
| Host name filter | ip.host = hostname |
| MAC address filter | eth.addr == 00:70:f4:23:18:c4 |
| RST flag filter | tcp.flag.reset == 1 |

Main Toolbar Items

| Main toolbar items | Toolbar Icon | Toolbar Item | Menu Item | Description |
|--------------------|--------------|----------------------|-----------|---|
| | Start | Capture → Start | | Uses the same packet capturing options as previous session, or uses defaults if no options were set |
| | Stop | Capture → Stop | | Stops currently active capture |
| | Restart | Capture → Restart | | Restarts active capture session |
| | Options... | Capture → Options... | | Opens “Capture Options” dialog box |

| Main toolbar items | | | |
|--------------------|-----------------|-----------------------|--|
| | Open... | File → Open... | Opens “File open” dialog box to load a cap viewing |
| | Save As... | File → Save As... | Save current capture file |
| | Close | File → Close | Close current capture file |
| | Reload | View → Reload | Reloads current capture file |
| | Find Packet... | Edit → Find Packet... | Find packet based on different criteria |
| | Go Back | Go → Go Back | Jump back in the packet history |
| | Go Forward | Go → Go Forward | Jump forward in the packet history |
| | Go to Packet... | Go → Go to Packet... | Go to specific packet |

| Main toolbar items | | | |
|--------------------|-----------------------------|------------------------------------|---|
| | Go To First Packet | Go → First Packet | Jump to first packet of the capture file |
| | Go To Last Packet | Go → Last Packet | Jump to last packet of the capture file |
| | Auto Scroll in Live Capture | View → Auto Scroll in Live Capture | Auto scroll packet list during live capture |
| | Colorize | View → Colorize | Colorize the packet list (or not) |
| | Zoom In | View → Zoom In | Zoom into the packet data (increase the font size) |
| | Zoom Out | View → Zoom Out | Zoom out of the packet data (decrease the font size) |
| | Normal Size | View → Normal Size | Set zoom level back to 100% |
| | Resize Columns | View → Resize Columns | Resize columns, so the content fits to the window width |

Others

Here is a summary of some of the other important features of Wireshark that we have not covered yet:

| Feature | Description |
|------------------------|---|
| Expert info | Wireshark's expert system provides information on various issues and anomalies within captured traffic, such as retransmissions, duplicate packets, and protocol errors. |
| Statistics | Wireshark provides various statistics and graphs for captured traffic, such as protocol hierarchy, conversation statistics, and packet length distribution. |
| Customization | Wireshark offers a high degree of customization, including the ability to create custom columns, protocols, and dissector plugins. |
| Exporting | Wireshark can export captured traffic to various formats, such as plain text, CSV, XML, and pcapng. |
| Command-line interface | Wireshark can be used through its command-line interface, tshark, which provides similar functionality as the GUI but with greater automation and scripting capabilities. |

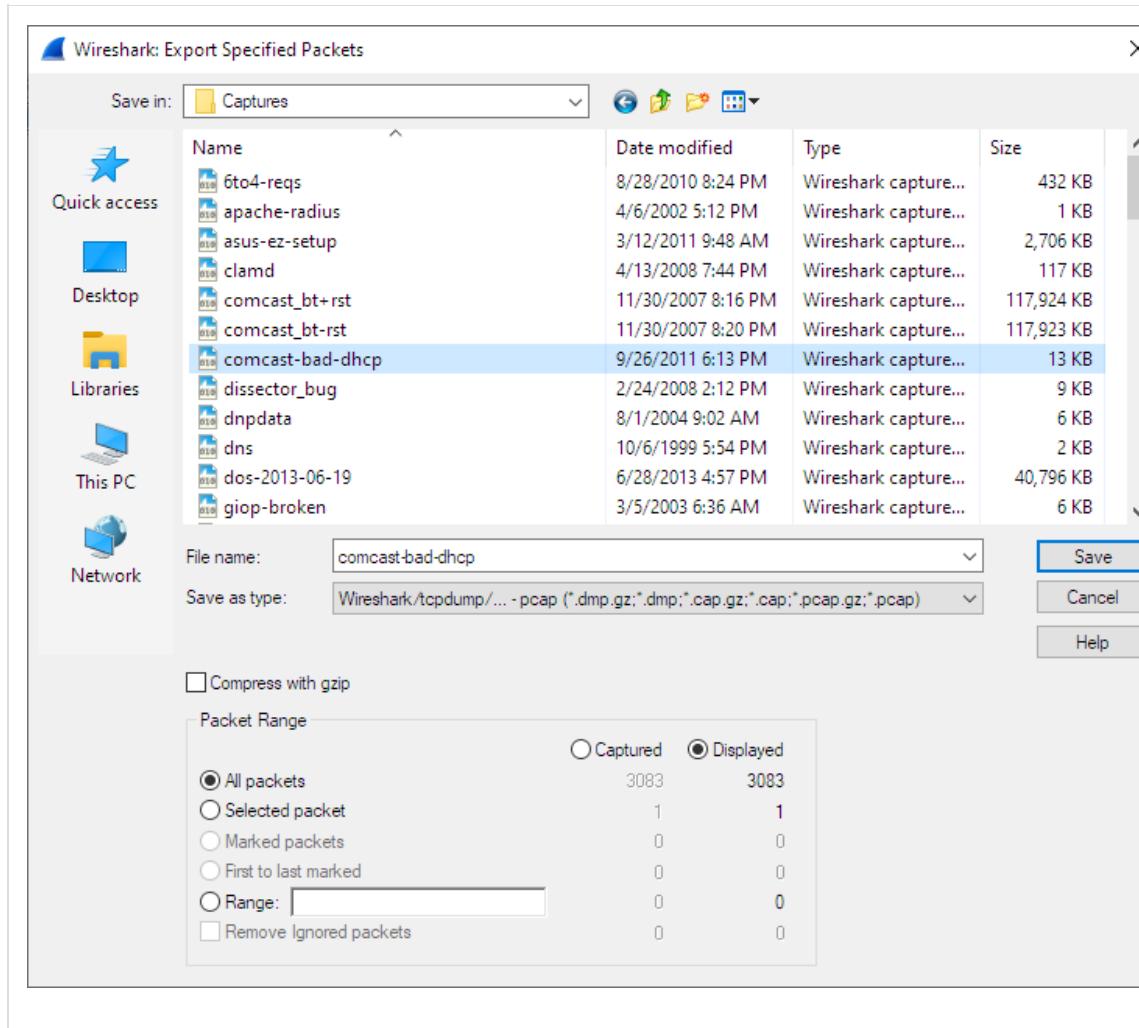
<https://www.codelivly.com/wireshark-cheatsheet/>

Exporting Data

Wireshark provides a variety of options for exporting packet data. This section describes general ways to export data from the main Wireshark application. There are many other ways to export or extract data from capture files, including processing [tshark](#) output and customizing Wireshark and TShark using Lua scripts.

5.7.1. The “Export Specified Packets” Dialog Box

Figure 5.10. The “Export Specified Packets” dialog box

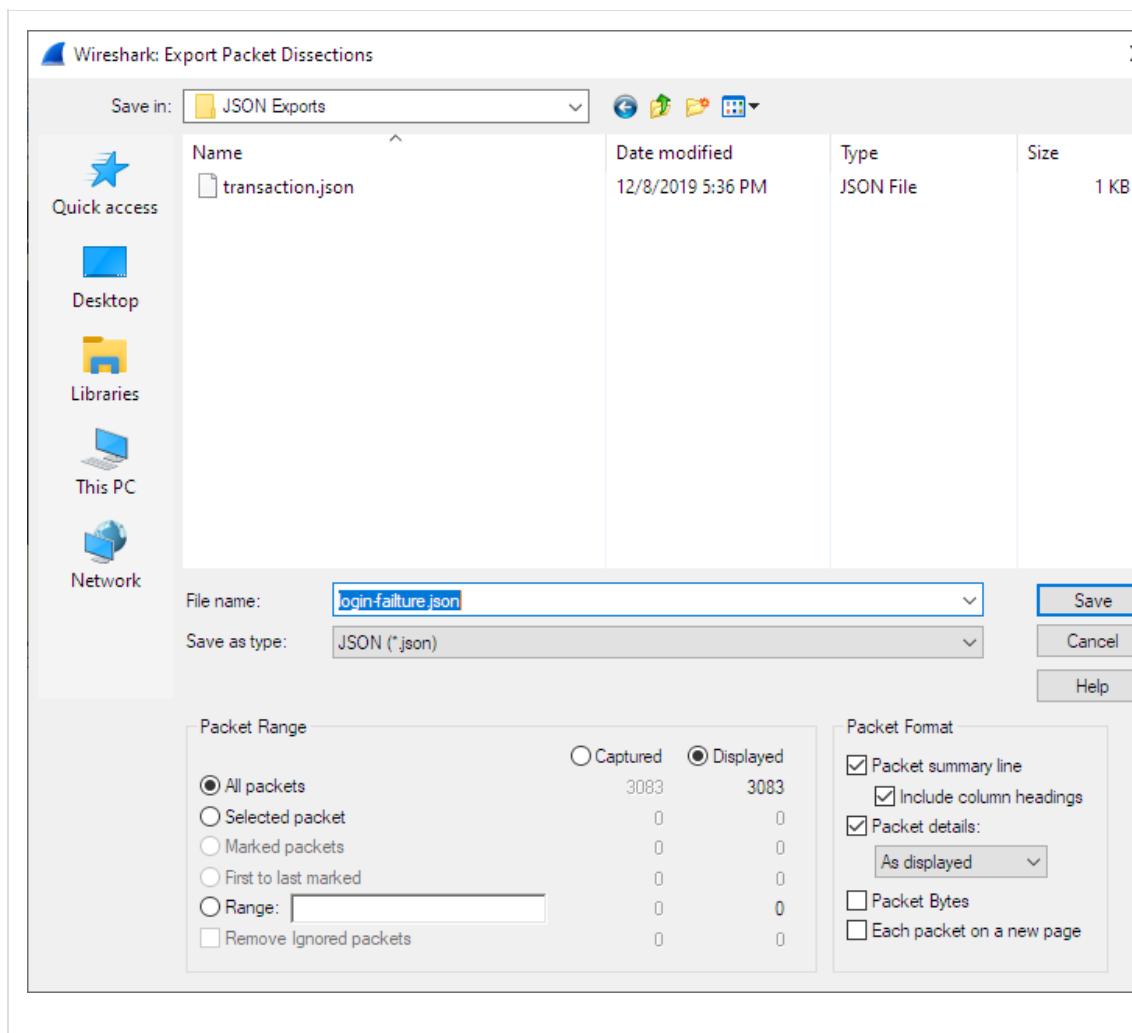


This is similar to the “[Save](#)” dialog box, but it lets you save specific packets. This can be useful for trimming irrelevant or unwanted packets from a capture file. See [Packet Range](#) for details on the range controls.

5.7.2. The “Export Packet Dissections” Dialog Box

This lets you save the packet list, packet details, and packet bytes as plain text, CSV, JSON, and other formats.

Figure 5.11. The “Export Packet Dissections” dialog box



The format can be selected from the “Export As” drop-down and further customized using the “[Packet Range](#)” and “[Packet Format](#)” controls. Some controls are unavailable for some formats, notably CSV and JSON. The following formats are supported:

- Plain text as shown in the main window
- [Comma-separated values \(CSV\)](#)
- [C-compatible byte arrays](#)
- [PSML \(summary XML\)](#)
- [PDML \(detailed XML\)](#)
- [JavaScript Object Notation \(JSON\)](#)

Here are some examples of exported data:

Plain text.

| No. | Time | Source | Destination |
|----------|--------|--------|-------------|
| Protocol | Length | SSID | Info |

```

1 0.000000      200.121.1.131      172.16.0.122
TCP      1454          10554 → 80 [ACK] Seq=1 Ack=1
Win=65535 Len=1400 [TCP segment of a reassembled PDU]

Frame 1: 1454 bytes on wire (11632 bits), 1454 bytes
captured (11632 bits)
Ethernet II, Src: 00:50:56:c0:00:01, Dst:
00:0c:29:42:12:13
Internet Protocol Version 4, Src: 200.121.1.131
(200.121.1.131), Dst: 172.16.0.122 (172.16.0.122)
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
    Differentiated Services Field: 0x00 (DSCP: CS0, ECN:
Not-ECT)
        Total Length: 1440
        Identification: 0x0141 (321)
        Flags: 0x0000
            ...0 0000 0000 0000 = Fragment offset: 0
        Time to live: 106
        Protocol: TCP (6)
        Header checksum: 0xd390 [validation disabled]
[Header checksum status: Unverified]
        Source: 200.121.1.131 (200.121.1.131)
        Destination: 172.16.0.122 (172.16.0.122)
[Source GeoIP: PE, ASN 6147, Telefonica del Peru
S.A.A.]
        Transmission Control Protocol, Src Port: 10554, Dst Port:
80, Seq: 1, Ack: 1, Len: 1400

```

Tip

If you would like to be able to [import](#) any previously exported packets from a plain text file it is recommended that you do the following:

- Add the “Absolute date and time” column.
- Temporarily hide all other columns.
- Disable the Edit → Preferences → Protocols → Data “Show not dissected data on new Packet Bytes pane” preference. More details are provided in [Section 11.5, “Preferences”](#)
- Include the packet summary line.
- Exclude column headings.
- Exclude packet details.
- Include the packet bytes.

CSV.

```
"No.", "Time", "Source", "Destination", "Protocol", "Length", "SID",
"Info", "Win Size"
"1", "0.000000", "200.121.1.131", "172.16.0.122", "TCP", "1454",
" ", "10554 > 80 [ACK] Seq=1 Ack=1 Win=65535 Len=1400
[TCP segment of a reassembled PDU]", "65535"
"2", "0.000011", "172.16.0.122", "200.121.1.131", "TCP", "54",
" ", "[TCP ACKed unseen segment] 80 > 10554 [ACK] Seq=1
Ack=11201 Win=53200 Len=0", "53200"
```

```

"3", "0.025738", "200.121.1.131", "172.16.0.122", "TCP", "1454"
,"","[TCP Spurious Retransmission] 10554 > 80 [ACK]
Seq=1401 Ack=1 Win=65535 Len=1400 [TCP segment of a
reassembled PDU]", "65535"
"4", "0.025749", "172.16.0.122", "200.121.1.131", "TCP", "54", "
,",[TCP Window Update] [TCP ACKed unseen segment] 80 >
10554 [ACK] Seq=1 Ack=11201 Win=63000 Len=0", "63000"
"5", "0.076967", "200.121.1.131", "172.16.0.122", "TCP", "1454"
,"","[TCP Previous segment not captured] [TCP Spurious
Retransmission] 10554 > 80 [ACK] Seq=4201 Ack=1
Win=65535 Len=1400 [TCP segment of a reassembled
PDU]", "65535"

```

JSON.

```

{
  "_index": "packets-2014-06-22",
  "_type": "doc",
  "_score": null,
  "_source": {
    "layers": {
      "frame": {
        "frame.encap_type": "1",
        "frame.time": "Jun 22, 2014 13:29:41.834477000
PDT",
        "frame.offset_shift": "0.000000000",
        "frame.time_epoch": "1403468981.834477000",
        "frame.time_delta": "0.450535000",
        "frame.time_delta_displayed": "0.450535000",
        "frame.time_relative": "0.450535000",
        "frame.number": "2",
        "frame.len": "86",
        "frame.cap_len": "86",
        "frame.marked": "0",
        "frame.ignored": "0",
        "frame.protocols": "eth:ethertype:ipv6:icmpv6",
        "frame.coloring_rule.name": "ICMP",
        "frame.coloring_rule.string": "icmp || icmpv6"
      },
      "eth": {
        "eth.dst": "33:33:ff:9e:e3:8e",
        "eth.dst_tree": {
          "eth.dst_resolved": "33:33:ff:9e:e3:8e",
          "eth.dst.oui": "3355647",
          "eth.addr": "33:33:ff:9e:e3:8e",
          "eth.addr_resolved": "33:33:ff:9e:e3:8e",
          "eth.addr.oui": "3355647",
          "eth.dst.lg": "1",
          "eth.lg": "1",
          "eth.dst.ig": "1",
          "eth.ig": "1"
        },
        "eth.src": "00:01:5c:62:8c:46",
        "eth.src_tree": {
          "eth.src_resolved": "00:01:5c:62:8c:46",
          "eth.src.oui": "348",
          "eth.src.oui_resolved": "Cadant Inc."
        }
      }
    }
  }
}
```

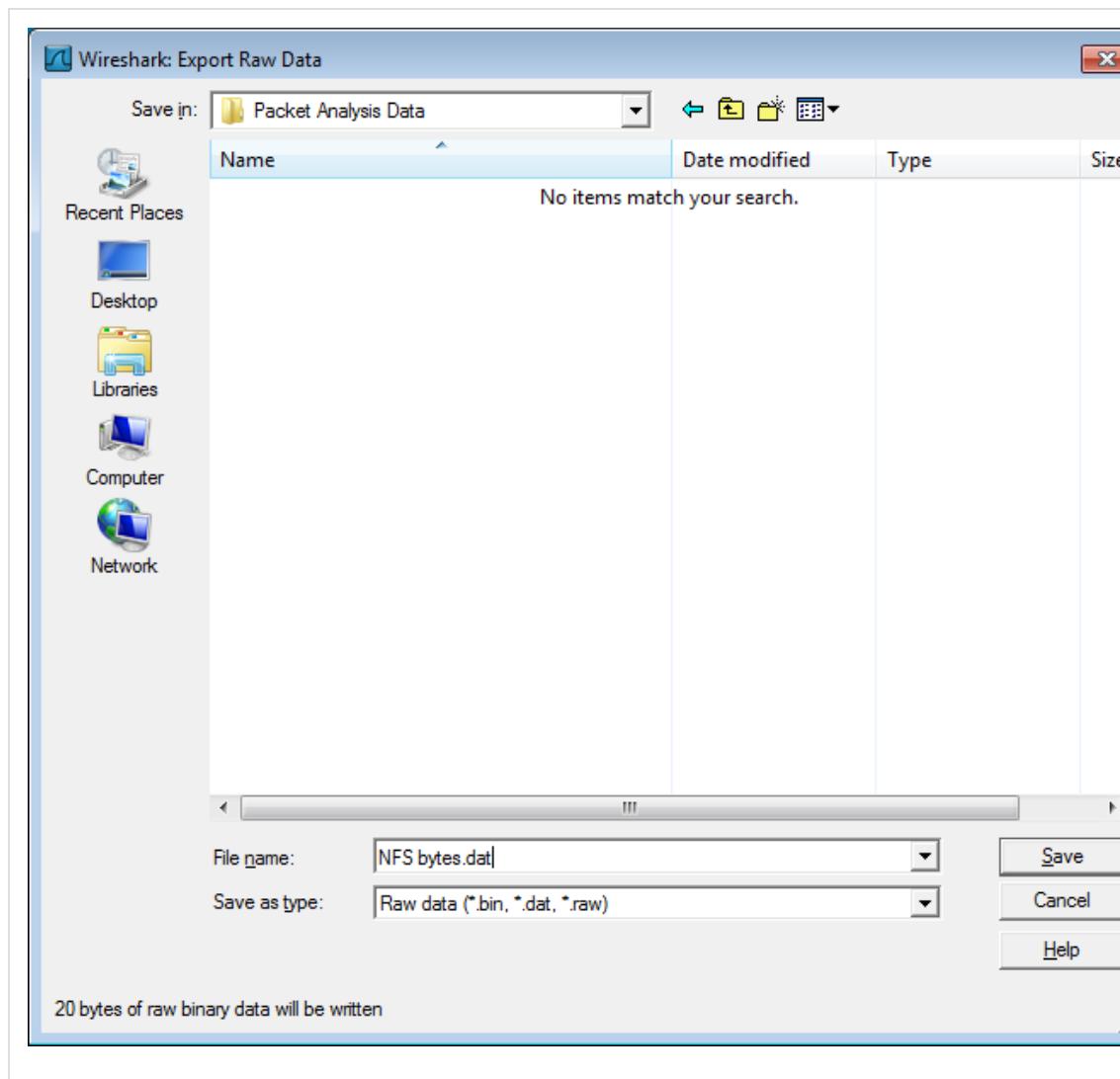
```
        "eth.addr": "00:01:5c:62:8c:46",
        "eth.addr_resolved": "00:01:5c:62:8c:46",
        "eth.addr.oui": "348",
        "eth.addr.oui_resolved": "Cadant Inc.",
        "eth.src.lg": "0",
        "eth.lg": "0",
        "eth.src.ig": "0",
        "eth.ig": "0"
    },
    "eth.type": "0x000086dd"
},
"ipv6": {
    "ipv6.version": "6",
    "ip.version": "6",
    "ipv6.tclass": "0x00000000",
    "ipv6.tclass_tree": {
        "ipv6.tclass.dscp": "0",
        "ipv6.tclass.ecn": "0"
    },
    "ipv6.flow": "0x00000000",
    "ipv6.plen": "32",
    "ipv6.nxt": "58",
    "ipv6.hlim": "255",
    "ipv6.src": "2001:558:4080:16::1",
    "ipv6.addr": "2001:558:4080:16::1",
    "ipv6.src_host": "2001:558:4080:16::1",
    "ipv6.host": "2001:558:4080:16::1",
    "ipv6.dst": "ff02::1:ff9e:e38e",
    "ipv6.addr": "ff02::1:ff9e:e38e",
    "ipv6.dst_host": "ff02::1:ff9e:e38e",
    "ipv6.host": "ff02::1:ff9e:e38e",
    "ipv6.geoip.src_summary": "US, ASN 7922, Comcast
Cable Communications, LLC",
    "ipv6.geoip.src_summary_tree": {
        "ipv6.geoip.src_country": "United States",
        "ipv6.geoip.country": "United States",
        "ipv6.geoip.src_country_iso": "US",
        "ipv6.geoip.country_iso": "US",
        "ipv6.geoip.src_asnum": "7922",
        "ipv6.geoip.asnum": "7922",
        "ipv6.geoip.src_org": "Comcast Cable
Communications, LLC",
        "ipv6.geoip.org": "Comcast Cable
Communications, LLC",
        "ipv6.geoip.src_lat": "37.751",
        "ipv6.geoip.lat": "37.751",
        "ipv6.geoip.src_lon": "-97.822",
        "ipv6.geoip.lon": "-97.822"
    }
},
"icmpv6": {
    "icmpv6.type": "135",
    "icmpv6.code": "0",
    "icmpv6.checksum": "0x00005b84",
    "icmpv6.checksum.status": "1",
    "icmpv6.reserved": "00:00:00:00",
    "icmpv6.nd.ns.target_address":
"2001:558:4080:16:be36:e4ff:fe9e:e38e",
```

```
        "icmpv6.opt": {
            "icmpv6.opt.type": "1",
            "icmpv6.opt.length": "1",
            "icmpv6.opt.linkaddr": "00:01:5c:62:8c:46",
            "icmpv6.opt.src_linkaddr": "00:01:5c:62:8c:46"
        }
    }
}
]
```

5.7.3. The “Export Selected Packet Bytes” Dialog Box

Export the bytes selected in the “Packet Bytes” pane into a raw binary file.

Figure 5.12. The “Export Selected Packet Bytes” dialog box



File name

The file name to export the packet data to.

Save as type

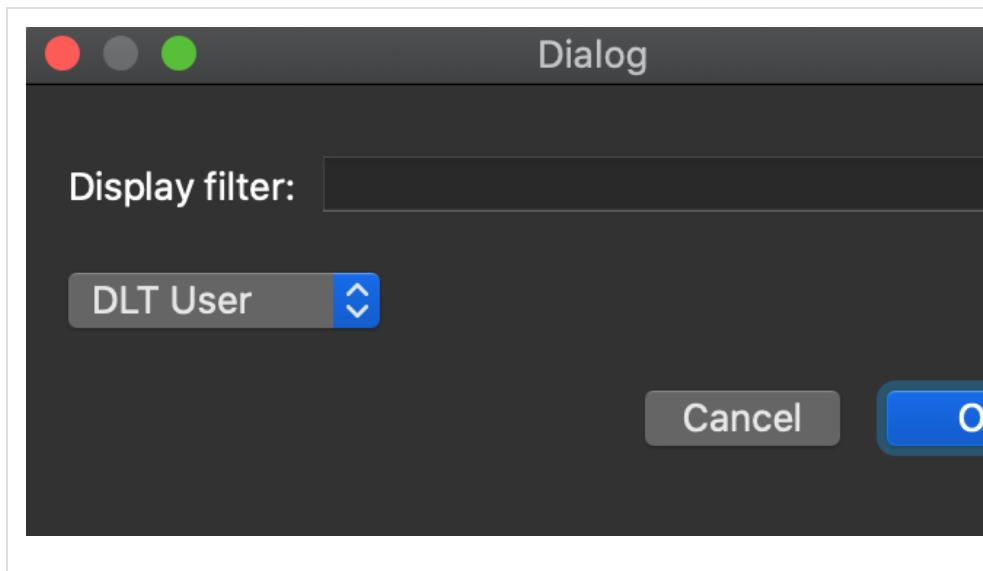
The file extension.

5.7.4. The “Export PDUs to File...” Dialog Box

The “Export PDUs to File...” dialog box allows you to filter the captured Protocol Data Units (PDUs) and export them into the file. It allows you to export reassembled PDUs avoiding lower layers such as HTTP without TCP, and decrypted PDUs without the lower protocols such as HTTP without TLS and TCP.

1. In the main menu select File → Export PDUs to File.... Wireshark will open a corresponding dialog [Figure 5.13, “Export PDUs to File window”](#).

Figure 5.13. Export PDUs to File window



2. To select the data according to your needs, optionally type a filter value into the **Display Filter** field. For more information about filter syntax, see the [Wireshark Filters](#) man page.
3. In the field below the **Display Filter** field you can choose the level from which you want to export the PDUs to the file. There are seven levels:
 - a. **DLT User**. You can export a protocol, which is framed in the user data link type table without the need to reconfigure the DLT user table. For more information, see the [How to Dissect Anything](#) page.
 - b. **DVB-CI**. You can use it for the Digital Video Broadcasting (DVB) protocol.
 - c. **Logcat** and **Logcat Text**. You can use them for the Android logs.

- d. **OSI layer 3.** You can use it to export PDUs encapsulated in the IPSec or SCTP protocols.
- e. **OSI layer 4.** You can use it to export PDUs encapsulated in the TCP or UDP protocols.
- f. **OSI layer 7.** You can use it to export the following protocols: CredSSP over TLS, Diameter, protocols encapsulated in TLS and DTLS, H.248, Megaco, RELOAD framing, SIP, SMPP.

Note

As a developer you can add any dissector to the existing list or define a new entry in the list by using the functions in `epan/exported_pdu.h`.

4. To finish exporting PDUs to file, click the OK button in the bottom-right corner. This will close the originally captured file and open the exported results instead as a temporary file in the main Wireshark window.
5. You may save the temporary file just like any captured file.

See [Section 5.3, “Saving Captured Packets”](#) for details.

Note

The file produced has a `Wireshark_Upper_PDU` encapsulation type that has somewhat limited support outside of Wireshark, but is very flexible and can contain PDUs for any protocol for which there is a Wireshark dissector.

5.7.5. The “Strip Headers...” Dialog Box

The “Strip Headers...” dialog box allows you to filter known encapsulation types on whatever protocol layer they appear and export them into a new capture file, removing lower-level protocols. It allows you to export reassembled packets and frames without lower layers such as GPF, GRE, GSE, GTP-U, MPLS, MPE, PPP, and more. If Wireshark has performed decryption, then you can export decrypted IP from protocols like IEEE 802.11 or IPSec without having to save encryption keys.

The procedure is similar to that of [Section 5.7.4, “The “Export PDUs to File...” Dialog Box”](#):

1. In the main menu select File → Strip Headers.... Wireshark will open a corresponding dialog.
2. To select the data according to your needs, optionally type a filter value into the `Display Filter` field. For more information about filter syntax, see the [Wireshark Filters](#) man page.

3. In the field below the **Display Filter** field you can choose the encapsulation type you want to find and export to the file. There are two encapsulations supported:
 - a. **Ethernet**. You can use it to export Ethernet encapsulated in other protocols.
 - b. **IP**. You can use it to export IPv4 and IPv6 encapsulated in other protocols.

Note

As a developer you can add encapsulations to the list by using the functions in `epan/exported_pdu.h`.

4. To finish exporting to file, click the OK button in the bottom-right corner. This will close the originally captured file and open the exported results instead as a temporary file in the main Wireshark window.
5. You may save the temporary file just like any captured file. See [Section 5.3, “Saving Captured Packets”](#) for details.

Note

The new capture files produced have standard encapsulation types and can be read in nearly any tool.

5.7.6. The “Export TLS Session Keys...” Dialog Box

Transport Layer Security (TLS) encrypts the communication between a client and a server. The most common use for it is web browsing via HTTPS.

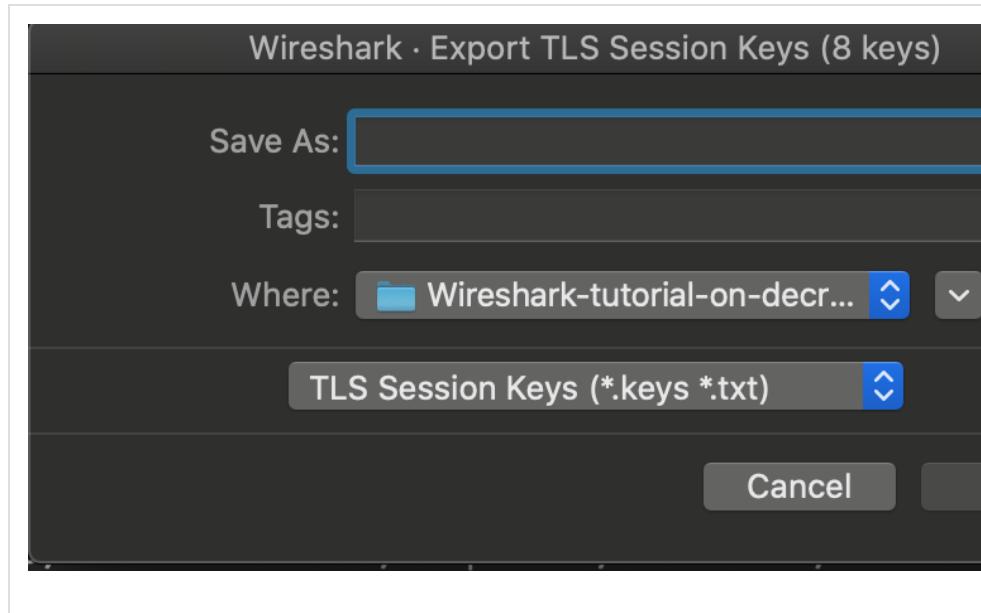
Decryption of TLS traffic requires TLS secrets. You can get them in the form of stored session keys in a "key log file", or by using an RSA private key file. For more details, see the [TLS wiki page](#).

The File → Export TLS Session Keys... menu option generates a new "key log file" which contains TLS session secrets known by Wireshark. This feature is useful if you typically decrypt TLS sessions using the RSA private key file. The RSA private key is very sensitive because it can be used to decrypt other TLS sessions and impersonate the server. Session keys can be used only to decrypt sessions from the packet capture file. However, session keys are the preferred mechanism for sharing data over the Internet.

To export captured TLS session keys, follow the steps below:

1. In the main menu select File → Export TLS Session Keys.... Wireshark will open a corresponding dialog [Figure 5.14, “Export TLS Session Keys window”](#).

Figure 5.14. Export TLS Session Keys window



2. Type the desired file name in the **Save As** field.
3. Choose the destination folder for your file in the **Where** field.
4. Press the Save button to complete the export file procedure.

5.7.7. The “Export Objects” Dialog Box

This feature scans through the selected protocol’s streams in the currently open capture file or running capture and allows the user to export reassembled objects to the disk. For example, if you select HTTP, you can export HTML documents, images, executables, and any other files transferred over HTTP to the disk. If you have a capture running, this list is automatically updated every few seconds with any new objects seen. The saved objects can then be opened or examined independently of Wireshark.

Figure 5.15. The “Export Objects” dialog box

Wireshark - Export - HTTP object list

| Packet | Hostname | Content Type | Size | Filename |
|--------|-----------------------|--------------------------|-------------|-----------------|
| 54 | www.msftncsi.com | text/plain | 14 bytes | ncsi.txt |
| 132 | api.bing.com | text/html | 1,305 bytes | qsml.aspx?que |
| 163 | api.bing.com | text/html | 1,346 bytes | qsml.aspx?que |
| 177 | api.bing.com | text/html | 1,369 bytes | qsml.aspx?que |
| 198 | api.bing.com | text/html | 1,398 bytes | qsml.aspx?que |
| 212 | google.com | text/html | 219 bytes | / |
| 226 | www.google.com | text/html | 231 bytes | / |
| 1858 | www.google.com | text/html | 1,058 bytes | url?sa=t&rct= |
| 1904 | www.bluproducts.com | text/html | 19 kB | / |
| 1955 | www.bluproducts.com | text/css | 7,321 bytes | default_iceme |
| 1972 | www.bluproducts.com | text/css | 331 bytes | default_notjs.c |
| 2109 | www.bluproducts.com | text/css | 63 kB | widgetkit-2410 |
| 2136 | www.bluproducts.com | application/x-javascript | 4,707 bytes | core-816de4c |
| 2139 | www.bluproducts.com | application/x-javascript | 657 bytes | caption-5e0b3 |
| 2280 | www.bluproducts.com | application/x-javascript | 20 kB | widgetkit-34c2 |
| 2390 | www.bluproducts.com | application/x-javascript | 18 kB | cufon-yui-1d10 |
| 2545 | www.bluproducts.com | application/x-javascript | 95 kB | mootools-core |
| 2560 | www.bluproducts.com | application/x-javascript | 93 kB | jquery-7ae67c |
| 2689 | www.bluproducts.com | application/x-javascript | 4,784 bytes | core.js |
| 2728 | platform.linkedin.com | text/javascript | 3,768 bytes | in.js |
| 2743 | www.bluproducts.com | text/css | 132 kB | template-897f |
| 2784 | www.bluproducts.com | application/x-javascript | 22 kB | template-3f20 |
| 2898 | www.bluproducts.com | image/png | 19 kB | facebook.png |
| 2990 | www.bluproducts.com | image/png | 22 kB | Twitter.png |
| 3060 | www.bluproducts.com | image/png | 44 kB | googleplus.pn |
| 3066 | s.amazon-adsystem.com | image/gif | 43 bytes | iui37d=3p-hbg |
| 3145 | www.bluproducts.com | image/png | 19 kB | mail.png |

Text Filter:

[Help](#) [Save All](#) [Close](#) [Save](#)

Columns:

Packet

The packet number in which this object was found. In some cases, there can be multiple objects in the same packet.

Hostname

The hostname of the server that sent this object.

Content Type

The content type of this object.

Size

The size of this object in bytes.

Filename: The filename for this object. Each protocol generates the filename differently. For example, HTTP uses the final part of the URI and IMF uses the subject of the email.

Inputs:

Text Filter

Only displays objects containing the specified text string.

Help

Opens this section of the “User’s Guide”.

Save All

Saves all objects (including those not displayed) using the filename from the filename column. You will be asked what directory or folder to save them in.

Close

Closes the dialog without exporting.

Save

Saves the currently selected object as a filename you specify. The default filename to save as is taken from the filename column of the objects list.

https://www.wireshark.org/docs/wsug_html_chunked/ChIOExportSection.html

When reviewing packet captures (pcaps) of suspicious activity, security professionals may need to export objects from the pcaps for a closer examination.

This tutorial offers tips on how to export different types of objects from a pcap. The instructions assume you understand network traffic fundamentals. We will use [these](#) pcaps of network traffic to practice extracting objects using Wireshark. The instructions also assume you have customized your Wireshark column display as previously demonstrated in [this](#) tutorial.

Warning: Most of these pcaps contain Windows malware, and this tutorial involves examining these malicious files. Since these files are Windows malware, I recommend doing this tutorial in a non-Windows environment, like a MacBook or Linux host. You could also use a virtual machine (VM) running Linux.

This tutorial covers the following areas:

- Exporting objects from HTTP traffic
- Exporting objects from SMB traffic
- Exporting emails from SMTP traffic
- Exporting files from FTP traffic

Exporting Objects from HTTP Traffic

The first pcap for this tutorial, **extracting-objects-from-pcap-example-01.pcap**, is available [here](#). Open the pcap in Wireshark and filter on **http.request** as shown in Figure 1.

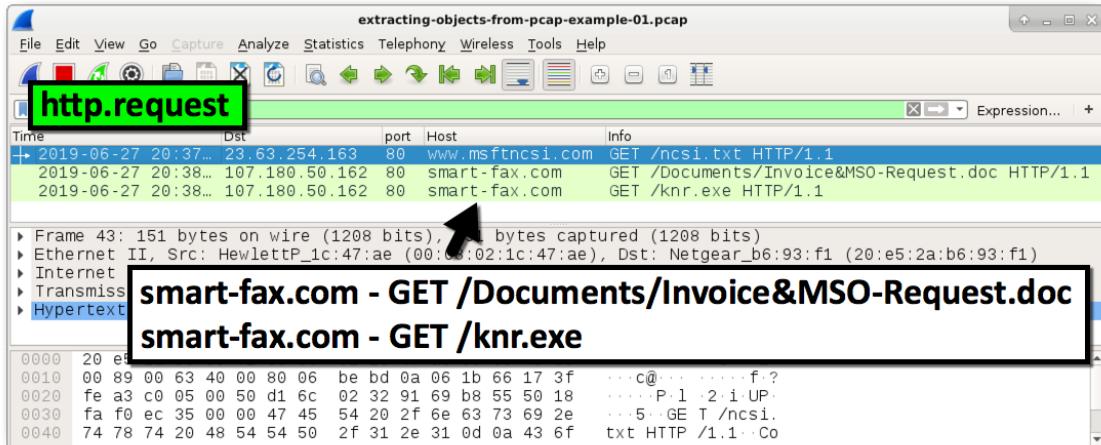


Figure 1. Filtering on the tutorial's first pcap in Wireshark.

After filtering on **http.request**, find the two GET requests to **smart-fax[.]com**. The first request ends with **.doc**, indicating the first request returned a Microsoft Word document. The second request ends with **.exe**, indicating the second request returned a Windows executable file. The HTTP GET requests are listed below.

- **smart-fax[.]com** - GET /Documents/Invoice&MSO-Request.doc
- **smart-fax[.]com** - GET /knr.exe

We can export these objects from the HTTP object list by using the menu path: **File --> Export Objects --> HTTP...** Figure 2 show this menu path in Wireshark.

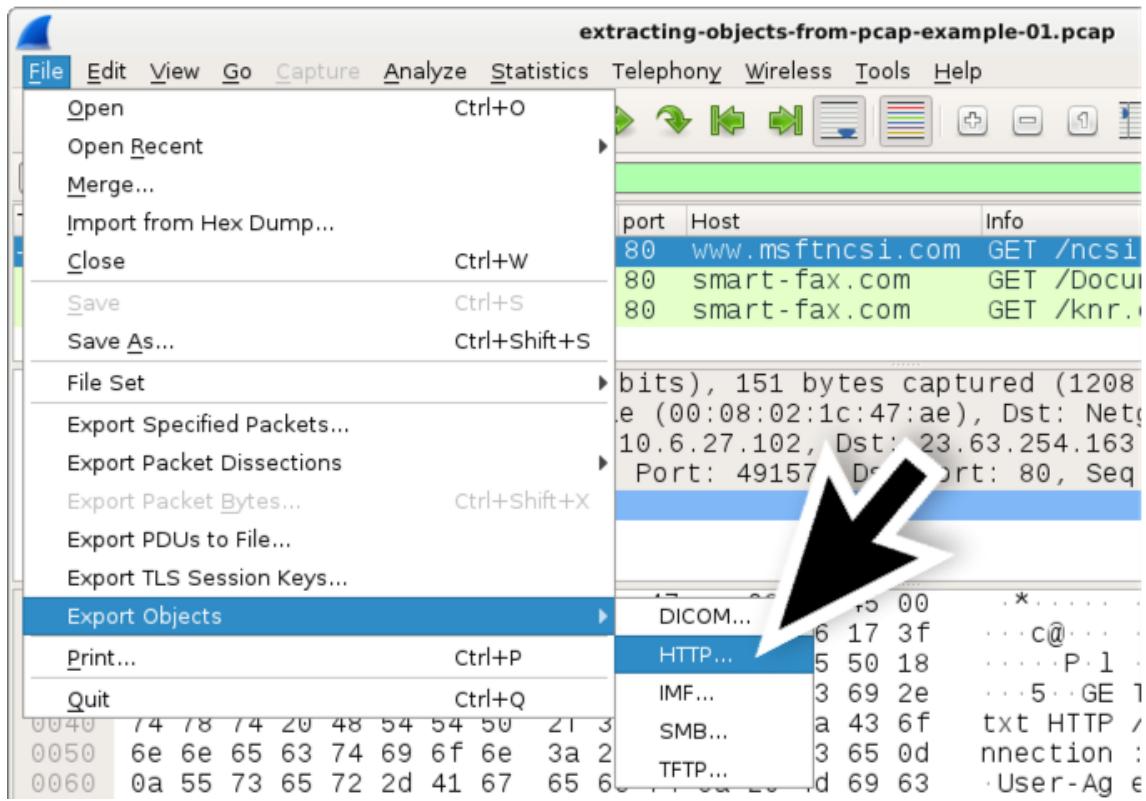


Figure 2. Exporting HTTP objects in Wireshark.

This menu path results in an Export HTTP object list window as shown in Figure 3. Select the first line with **smart-fax[.]com** as the hostname and save it as shown in Figure 3. Select the second line with **smart-fax[.]com** as the hostname and save it as shown in Figure 4.

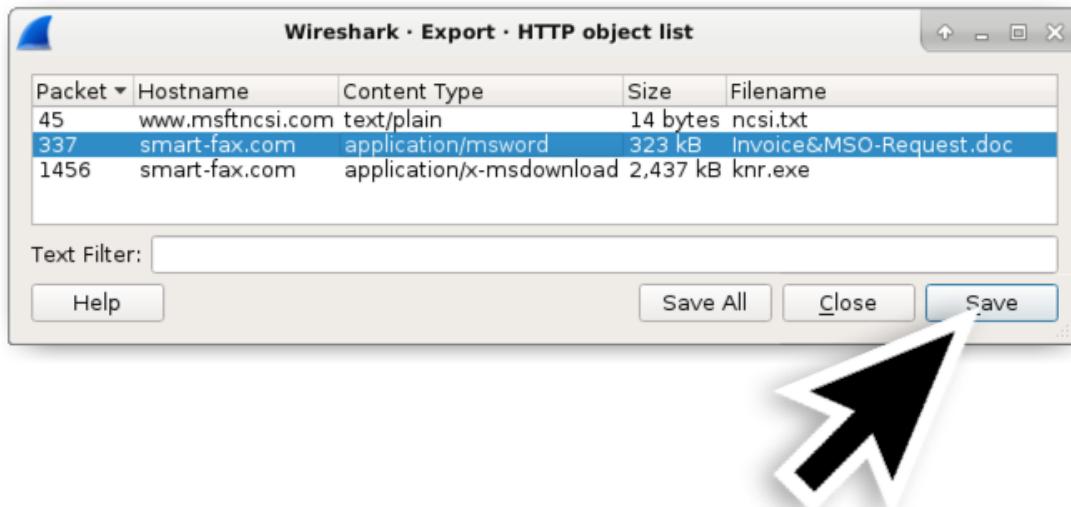


Figure 3. Saving the suspected Word document from the HTTP object list.

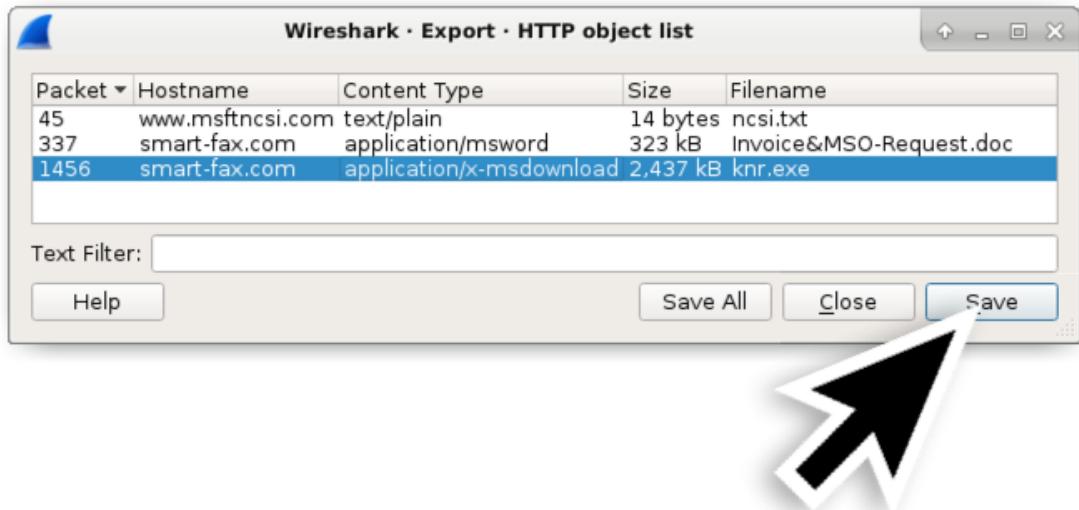


Figure 4. Saving the suspected Windows executable file from the HTTP object list.

Of note, the Content Type from the HTTP object list shows how the server identified the file in its HTTP response headers. In some cases, Windows executables are intentionally labeled as a different type of file in an effort to avoid detection. Fortunately, the first pcap in this tutorial is a very straight-forward example.

Still, we should confirm these files are what we think they are. In a MacBook or Linux environment, you can use a terminal window or command line interface (CLI) for the following commands:

- **file [filename]**
- **shasum -a 256 [filename]**

The **file** command returns the type of file. The **shasum** command will return the file hash, in this case the SHA256 file hash. Figure 5 shows using these commands in a CLI on a Debian-based Linux host.

```
Terminal - debian-user@debian-host: ~/Desktop
File Edit View Terminal Tabs Help
debian-user@debian-host:~$ cd Desktop/
debian-user@debian-host:~/Desktop$ file Invoice\&MSO-Request.doc
Invoice&MSO-Request.doc: Composite Document File V2 Document, Little Endian, Os: Windows,
Version 6.3, Code page: 1252, Template: Normal.dotm, Last Saved By: Administrator, Revision Number: 2, Name of Creating Application: Microsoft Office Word, Create Time/Date: Thu Jun 27 19:24:00 2019, Last Saved Time/Date: Thu Jun 27 19:24:00 2019, Number of Pages: 1, Number of Words: 0, Number of Characters: 1, Security: 0
debian-user@debian-host:~/Desktop$ file knr.exe
knr.exe: PE32 executable (GUI) Intel 80386, for MS Windows
debian-user@debian-host:~/Desktop$ shasum -a 256 Invoice\&MSO-Request.doc
f808229aa516ba134889f81cd699b8d246d46d796b55e13bee87435889a054fb  Invoice&MSO-Request.doc
debian-user@debian-host:~/Desktop$ shasum -a 256 knr.exe
749e161661290e8a2d190b1a66469744127bc25bf46e5d0c6f2e835f4b92db18  knr.exe
debian-user@debian-host:~/Desktop$
```

Figure 5. Determining the file type and hash of our two objects exported from the pcap.

The commands and their results from Figure 5 are listed below:

```
$ file Invoice\&MSO-Request.doc
Invoice&MSO-Request.doc: Composite Document File V2
Document, Little Endian, Os:Windows, Version 6.3,
Code page: 1252, Template: Normal.dotm, Last Saved
By: Administrator, Revision Number: 2, Name of
Creating Application: Microsoft Office Word, Create
Time/Date: Thu Jun 27 19:24:00 2019, Last Saved
Time/Date: Thu Jun 27 19:24:00 2019, Number of
Pages: 1, Number of Words: 0, Number of Characters:
1, Security: 0

$ file knr.exe
knr.exe: PE32 executable (GUI) Intel 80386, for MS
Windows

$ shasum -a 256 Invoice\&MSO-Request.doc
f808229aa516ba134889f81cd699b8d246d46d796b55e13bee87
435889a054fb Invoice&MSO-Request.doc

$ shasum -a 256 knr.exe
749e161661290e8a2d190b1a66469744127bc25bf46e5d0c6f2e
835f4b92db18 knr.exe
```

The information above confirms our suspected Word document is in fact a Microsoft Word document. It also confirms the suspected Windows executable file is indeed a Windows executable. We can check the SHA256 hashes against VirusTotal to see if these files are detected as malware. We could also do a Google search on the SHA256 hashes to possibly find additional information.

In addition to Windows executable or other malware files, we can also extract web pages. Our second pcap for this tutorial, ***extracting-objects-from-pcap-example-02.pcap*** (available [here](#)) contains traffic of someone entering login credentials on a fake PayPal login page.

When reviewing network traffic from a phishing site, we might want to see what the phishing web page looks like. We can extract the initial HTML page using the Export HTTP object menu as shown in Figure 6. Then we can view it through a web browser in an isolated environment as shown in Figure 7.

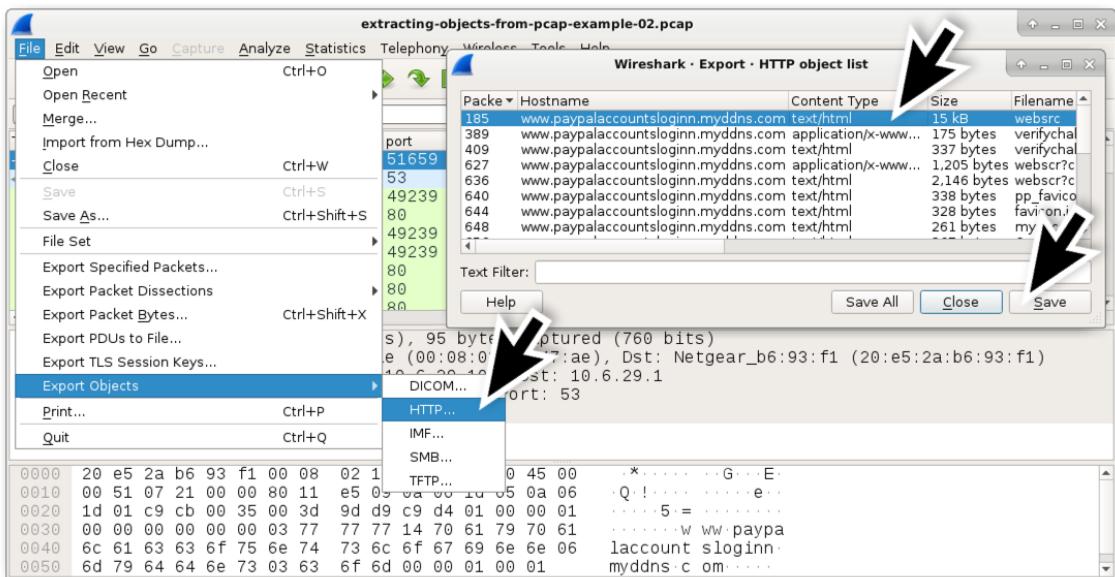


Figure 6. Exporting a fake PayPal login page from our second pcap.

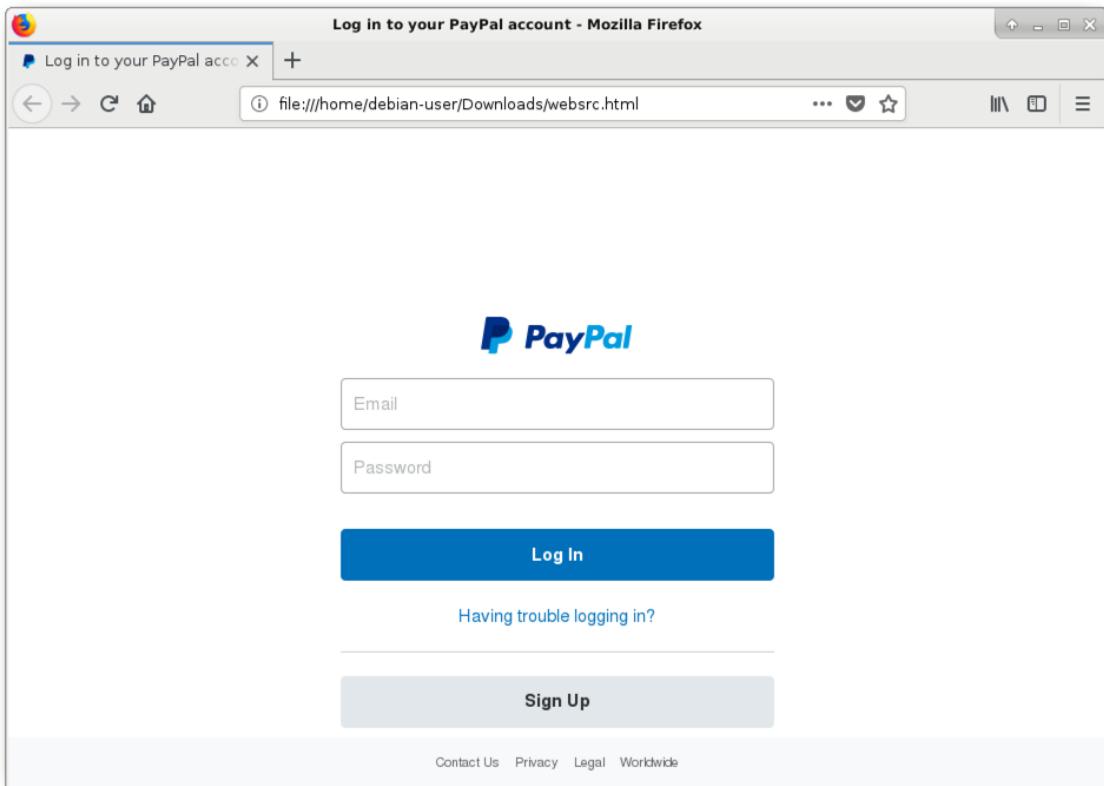


Figure 7. The exported fake PayPal login page viewed in a web browser.

Exporting Objects from SMB Traffic

Some malware uses Microsoft's Server Message Block (SMB) protocol to spread across an Active Directory (AD)-based network. A banking Trojan known as Trickbot added a worm module [as early as July 2017](#) that uses an exploit based on [EternalBlue](#) to spread across a network over SMB. We continue to find indications of this Trickbot worm module today.

Our next pcap represents a Trickbot infection that used SMB to spread from an infected client at 10.6.26.110 to its domain controller at 10.6.26.6. The pcap, [extracting-objects-from-pcap-example-03.pcap](#), is available [here](#). Open the pcap in Wireshark. Use the menu path **File --> Export Objects --> SMB...** as shown in Figure 8.

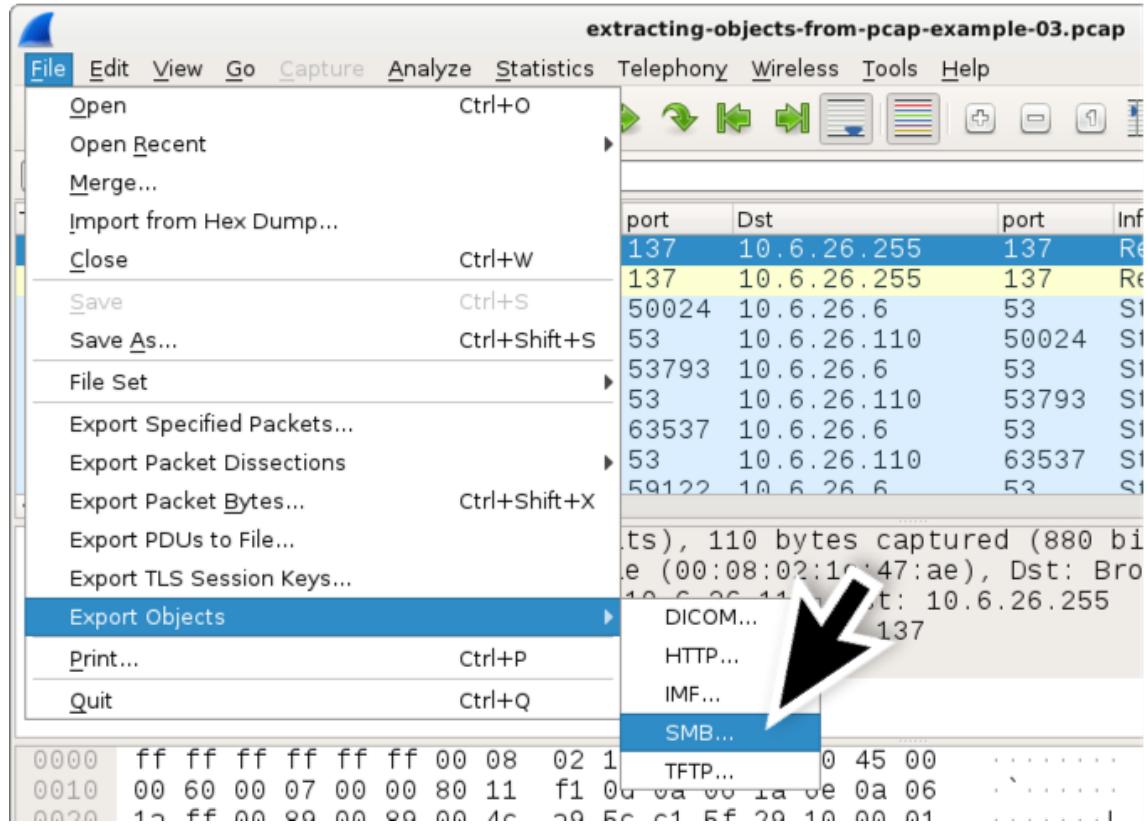


Figure 8. Getting to the Export SMB objects list.

This brings up an Export SMB object list, listing SMB objects you can export from the pcap as shown below in Figure 9.

| Packet | Hostname | Content Type | Size | Filename |
|--------|--------------------------------------|---|----------|---------------------------|
| 408 | \Cliffstone-DC.cliffstone.net\sysvol | FILE (22/22) R [100.00%] | 22 bytes | \cliffstone.net\Policies\ |
| 691 | \Cliffstone-DC.cliffstone.net\sysvol | FILE (22/22) R [100.00%] | 22 bytes | \cliffstone.net\Policies\ |
| 6991 | \10.6.26.6\IPC\$ | PIPE (Not Implemented) (0/0) W [0.00%] 0 bytes | 0 bytes | samr |
| 7058 | \10.6.26.6\C\$ | FILE (712704/712704) W [100.00%] | 712 kB | \WINDOWS\d0p2nc6ka |
| 7936 | \10.6.26.6\C\$ | FILE (115712/115712) W [100.00%] | 115 kB | \WINDOWS\oiku9bu68c |
| 8084 | \10.6.26.6\IPC\$ | PIPE (Not Implemented) (0/0) W [0.00%] 0 bytes | 0 bytes | \svctrl |
| 8168 | \10.6.26.6\IPC\$ | PIPE (Not Implemented) (0/0) W [0.00%] 0 bytes | 0 bytes | \svctrl |

Figure 9. The export SMB object list.

Notice the two entries near the middle of the list with \\10.6.26.6\C\$ as the Hostname. A closer examination of their respective Filename fields

indicates these are two Windows executable files. See Table 1 below for details.

| Packet number | Hostname | Content Type | Size | Filename |
|---------------|------------------|---------------------------------------|--------|--|
| 7058 | \\"10.6.26.6\C\$ | FILE (712704/712704) W [100.0%] | 712 kB | \WINDOWS\d0p2nc6ka3f_fixhohlycj4ovqfcy_s |
| 7936 | \\"10.6.26.6\C\$ | FILE (115712/115712) W [100.0%] | 115 kB | \WINDOWS\oiku9bu68cxqenfmcsos2aek6t07_ |

Table 1. Data from the Export SMB objects list on the two Windows executable files.

In the Content Type column, we need [100.00%] to export a correct copy of these files. Any number less than 100 percent indicates there was some data loss in the network traffic, resulting in a corrupt or incomplete copy of the file. These Trickbot-related files from the pcap have SHA256 file hashes as shown in Table 2.

| SHA256 hash |
|--|
| 59896ae5f3edcb999243c7bfd0b17eb7fe28f3a66259d797386ea470c010040 |
| cf99990bee6c378cbf56239b3cc88276eec348d82740f84e9d5c343751f82560 |

Table 2. SHA256 file hashes for the Windows executable files.

Exporting Emails from SMTP Traffic

Certain types of malware are designed to turn an infected Windows host into a spambot. These spambots send hundreds of spam messages or malicious emails every minute. In some cases, the messages are sent using unencrypted SMTP, and we can export these messages from a pcap of the infection traffic.

One such example is from our next pcap, [**extracting-objects-from-pcap-example-04.pcap**](#) (available [here](#)). In this pcap, an infected Windows client sends [sextortion spam](#). Open the pcap in Wireshark, filter on **smtp.data.fragment**, and you should see 50 examples of subject lines as shown in Figure 10. This happened in five seconds of network traffic from a single infected Windows host.

| Time | Dst | port | Info |
|---------------------|---------------|------|---|
| 2019-06-01 08:37:40 | 98.137.159.26 | 25 | from: Your Life<YourLife36@7162.com>, subject: Hurry up and pay! |
| 2019-06-01 08:37:40 | 98.137.159.26 | 25 | from: Your Life<YourLife56@1598.com>, subject: Your password! - |
| 2019-06-01 08:37:40 | 98.137.159.26 | 25 | from: Your Life<YourLife92@8738.com>, subject: No longer private |
| 2019-06-01 08:37:40 | 98.137.159.26 | 25 | from: Your Life<YourLife29@8738.com>, subject: You better read t |
| 2019-06-01 08:37:41 | 98.137.159.26 | 25 | from: Your Life<YourLife71@2261.com>, subject: I won't warn you |
| 2019-06-01 08:37:41 | 98.137.159.26 | 25 | from: Your Life<YourLife69@2844.com>, subject: Infected your com |
| 2019-06-01 08:37:41 | 98.137.159.26 | 25 | from: Your Life<YourLife39@0083.com>, subject: Recorded you mast |
| 2019-06-01 08:37:41 | 98.137.159.26 | 25 | from: Your Life<YourLife42@0083.com>, subject: I got everything! |
| 2019-06-01 08:37:41 | 98.137.159.26 | 25 | from: Your Life<YourLife56@7082.com>, subject: Read carefully! . |
| 2019-06-01 08:37:41 | 98.137.159.26 | 25 | from: Your Life<YourLife91@3903.com>, subject: I can destroy eve |
| 2019-06-01 08:37:41 | 98.137.159.26 | 25 | from: Your Life<YourLife02@7678.com>, subject: Your password! - |
| 2019-06-01 08:37:41 | 98.137.159.26 | 25 | from: Your Life<YourLife59@3510.com>, subject: Data of you and y |
| 2019-06-01 08:37:41 | 98.137.159.26 | 25 | from: Your Life<YourLife01@6073.com>, subject: Pay! - 12345, (t |
| 2019-06-01 08:37:41 | 98.137.159.26 | 25 | from: Your Life<YourLife01@6073.com>, subject: Pay! - tomcat, (|
| 2019-06-01 08:37:41 | 98.137.159.26 | 25 | from: Your Life<YourLife97@6073.com>, subject: Videos of you! - |
| 2019-06-01 08:37:41 | 106.10.248.75 | 25 | from: Your Life<YourLife53@3510.com>, subject: I seen everything |
| 2019-06-01 08:37:41 | 98.137.159.26 | 25 | from: Your Life<YourLife70@3953.com>, subject: Your private data |
| 2019-06-01 08:37:42 | 98.137.159.26 | 25 | from: Your Life<YourLife01@7417.com>, subject: You got owned! - |
| 2019-06-01 08:37:42 | 98.137.159.26 | 25 | from: Your Life<YourLife36@2231.com>, subject: Take care next ti |
| 2019-06-01 08:37:42 | 98.137.159.26 | 25 | from: Your Life<YourLife59@2231.com>, subject: Your life about to |
| 2019-06-01 08:37:42 | 98.137.159.26 | 25 | from: Your Life<YourLife29@5316.com>, subject: I won't warn you |
| 2019-06-01 08:37:42 | 98.137.159.26 | 25 | from: Your Life<YourLife15@6425.com>, subject: Your privacy! - J |
| 2019-06-01 08:37:42 | 98.137.159.26 | 25 | from: Your Life<YourLife42@2656.com>, subject: Safe your privacy |
| 2019-06-01 08:37:42 | 98.137.159.26 | 25 | from: Your Life<YourLife67@2656.com>, subject: Everyone will kno |
| 2019-06-01 08:37:42 | 98.137.159.26 | 25 | from: Your Life<YourLife63@2266.com>, subject: Your private data |
| 2019-06-01 08:37:42 | 98.137.159.26 | 25 | from: Your Life<YourLife63@2266.com>, subject: Your private data |

Figure 10. Filtering for email senders and subject lines in Wireshark.

You can export these messages using the menu path **File --> Export Objects --> IMF...** as shown in Figure 11. IMF stands for Internet Message Format, which is saved as a name with an .eml file extension.

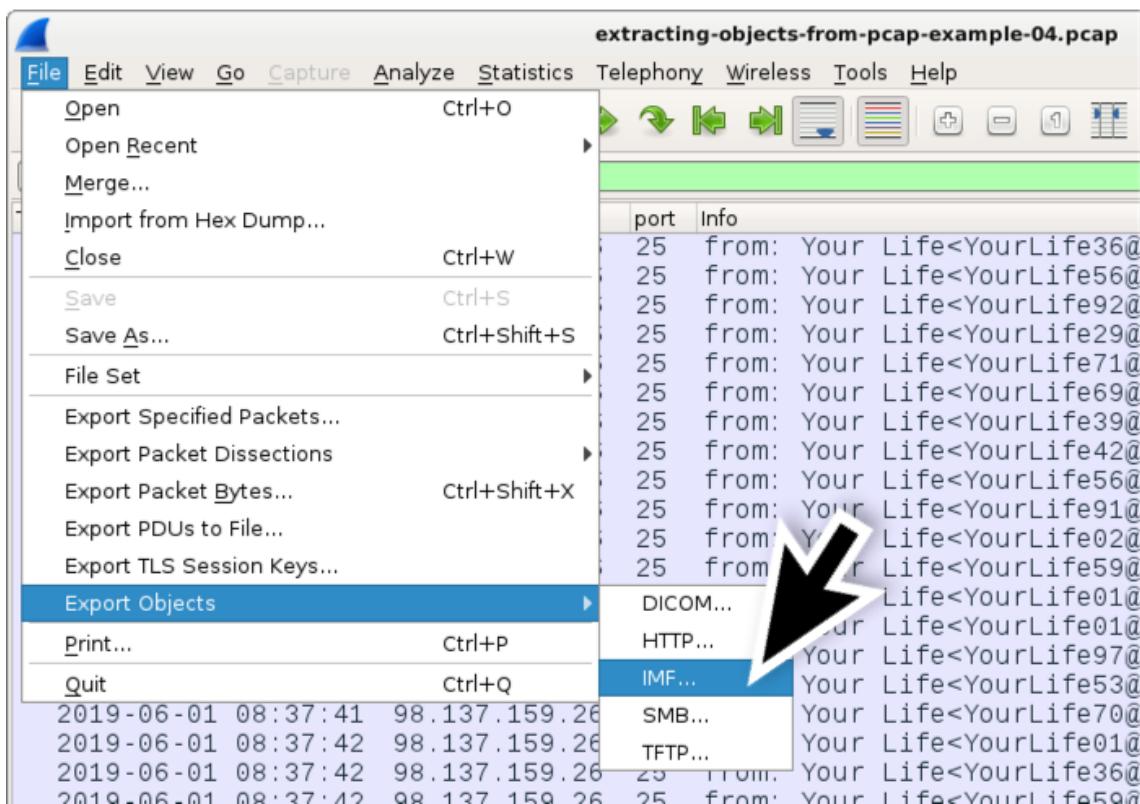


Figure 11. Exporting emails from a pcap in Wireshark.

The sextortion spam messages are all listed with an .eml file extension in the IMF object list as shown in Figure 12.

| Packet | Hostname | Content Type | Size | Filename |
|--------|---------------------|--------------|-------------|---|
| 338 | YourLife36@7162.com | EML file | 2,458 bytes | Hurry up and pay! - ganjaman.eml |
| 551 | YourLife56@1598.com | EML file | 2,461 bytes | Your password! - dontscrew.eml |
| 617 | YourLife92@8738.com | EML file | 2,462 bytes | No longer private! - 123456789.eml |
| 630 | YourLife29@8738.com | EML file | 2,465 bytes | You better read this! - asda123456.eml |
| 714 | YourLife71@2261.com | EML file | 2,473 bytes | I won't warn you again! - 93irish89.eml |
| 788 | YourLife69@2844.com | EML file | 2,461 bytes | Infected your computer! - dcjDik.eml |
| 814 | YourLife39@9083.com | EML file | 2,471 bytes | Recorded you masturbating! - 0910465419.eml |
| 840 | YourLife42@9083.com | EML file | 2,456 bytes | I got everything! - 121212.eml |
| 905 | YourLife56@7082.com | EML file | 2,453 bytes | Read carefully! - dayit.eml |
| 949 | YourLife91@3903.com | EML file | 2,468 bytes | I can destroy everything! - 12345678.eml |
| 966 | YourLife02@7678.com | EML file | 2,452 bytes | Your password! - nafd111.eml |
| 999 | YourLife59@3510.com | EML file | 2,466 bytes | Data of you and your family! - peace.eml |
| 1023 | YourLife01@6073.com | EML file | 2,443 bytes | Pay! - 12345.eml |
| 1030 | YourLife01@6073.com | EML file | 2,444 bytes | Pay! - tomcat.eml |
| 1052 | YourLife97@6073.com | EML file | 2,455 bytes | Videos of you! - 122577.eml |
| 1120 | YourLife53@3510.com | EML file | 2,462 bytes | I seen everything! - xbdmiy4.eml |
| 1130 | YourLife70@3953.com | EML file | 2,457 bytes | Your private data! - 2645885.eml |
| 1180 | YourLife01@7417.com | EML file | 2,452 bytes | You got owned! - üüghu.eml |

Figure 12. List of spam messages in the IMF object list.

After they are exported, these .eml files can be reviewed with an email client like Thunderbird, or they can be examined in a text editor as shown in Figure 13.

```

Received: from ffsrja ([29.140.42.132]) by 19322.com with MailEnable ESMTP; Sat, 1 Jun 2019 08:35:10 +0000
Received: (qmail 19322 invoked by uid 193);
From: Your Life<YourLife36@7162.com>
To: ikwlngpoh@yahoo.com
Subject: Hurry up and pay! - ganjaman
Date: Sat, 1 Jun 2019 08:35:10 +0000
Message-ID: <193221.597207@19322.com>
Mime-Version: 1.0
Content-type: text/plain; charset=utf-8;

Hi!

I know that: ganjaman - is your password!

Your computer was infected with my private malware, RAT, (Remote Administration Tool).

The malware gave me full access and control over your computer, I got access to all your files and data.

For a long time I was spying on you through your webcam and recorded MANY EMBARRASSING videos.

I collected all your private data, pictures, documents, videos, absolutely everything about you.

To not leave any traces, I removed my malware after that.

I can send the videos to all your contacts (email, social network) and publish all your private data online.

Only you can prevent me from doing this!

To stop me, pay exactly 1600$ in bitcoin (BTC).
If you don't know how to buy bitcoin, go to: www.paxful.com ( there are over 300 ways )

```

Figure 13. Using a text editor to view an .eml file exported from the pcap.

Exporting files from FTP Traffic

Some malware families use FTP during malware infections. Our next pcap has malware executables retrieved from an FTP server followed

by information from the infected Windows host sent back to the same FTP server.

The next pcap is ***extracting-objects-from-pcap-example-05.pcap*** and is available [here](#). Open the pcap in Wireshark. Filter on ***ftp.request.command*** to review the FTP commands as shown in Figure 14. You should find a username (USER) and password (PASS) followed by requests to retrieve (RETR) five Windows executable files: ***q.exe***, ***w.exe***, ***e.exe***, ***r.exe***, and ***t.exe***. This is followed by requests to store (STOR) html-based log files back to the same FTP server approximately every 18 seconds.

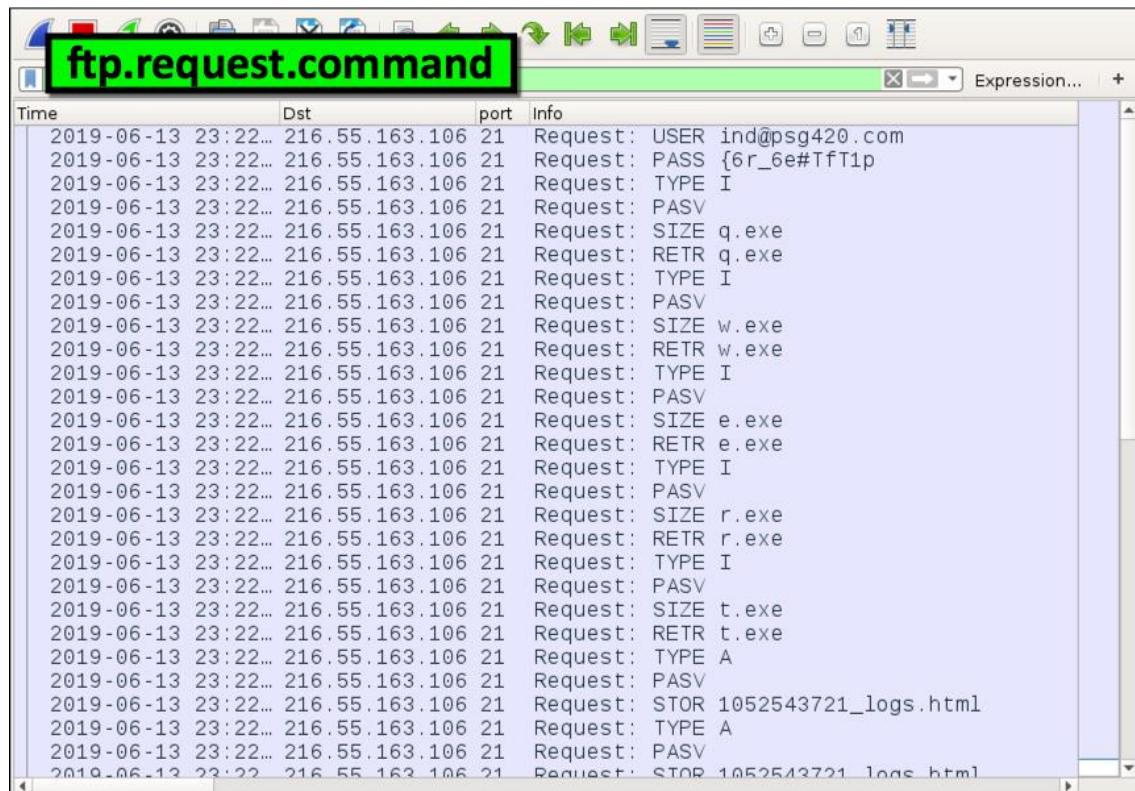


Figure 14. Filtering for FTP requests in Wireshark.

Now that we have an idea of the files that were retrieved and sent, we can review traffic from the FTP data channel using a filter for ***ftp-data*** as shown in Figure 15.

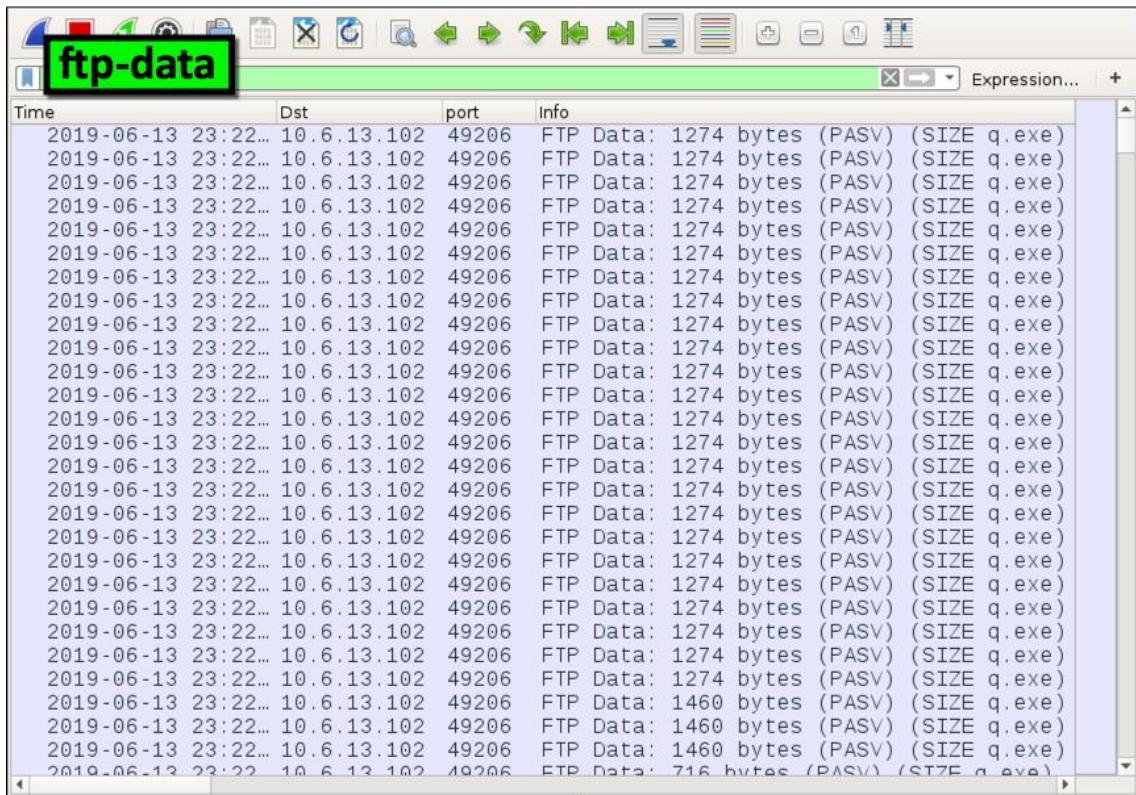


Figure 15. Filtering on FTP data traffic in Wireshark.

We cannot use the ***Export Objects*** function in Wireshark to export these objects. However, we can follow the TCP stream from the data channels for each. Left-click on any of the lines that end with (SIZE q.exe) to select one of the TCP segments. Then right-click to bring up a menu and select the menu path for ***Follow --> TCP stream*** as shown in Figure 16.

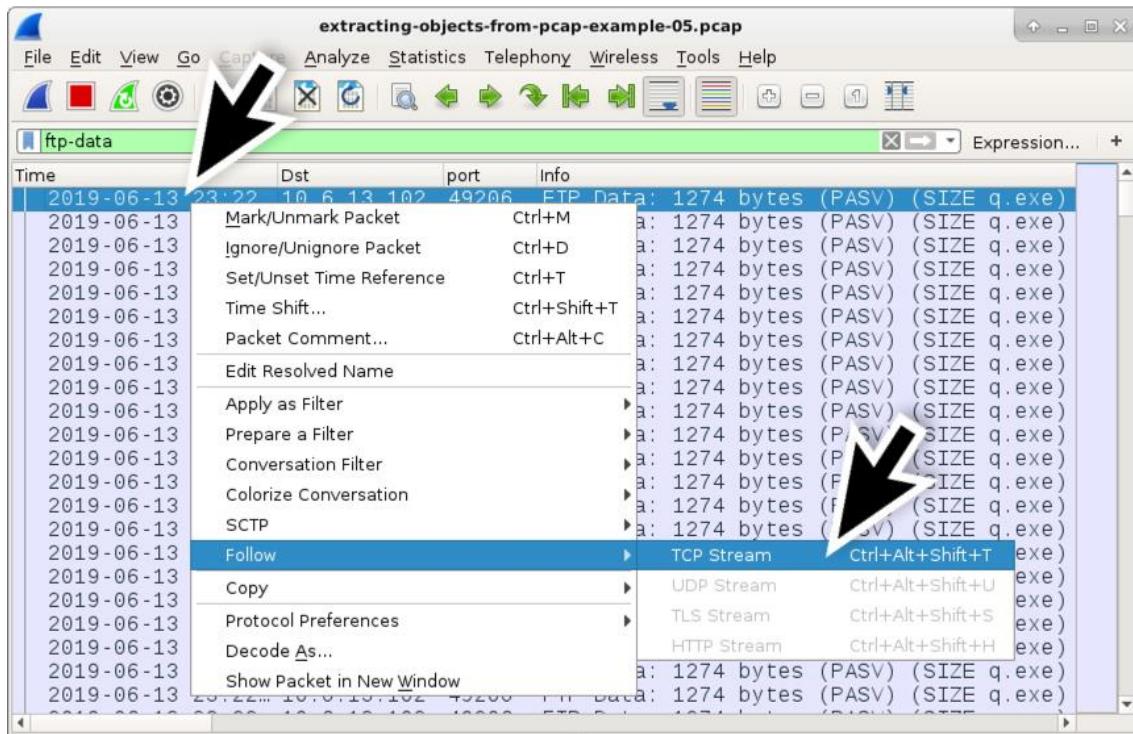


Figure 16. Following the TCP stream of an FTP data channel for **q.exe**.

This will bring up the TCP stream for **q.exe** over the FTP data channel. Near the bottom of the window is a button-style menu labeled "Show and save data as" which defaults to ASCII as shown in Figure 17. Click on the menu and select "Raw" as shown in Figure 18.

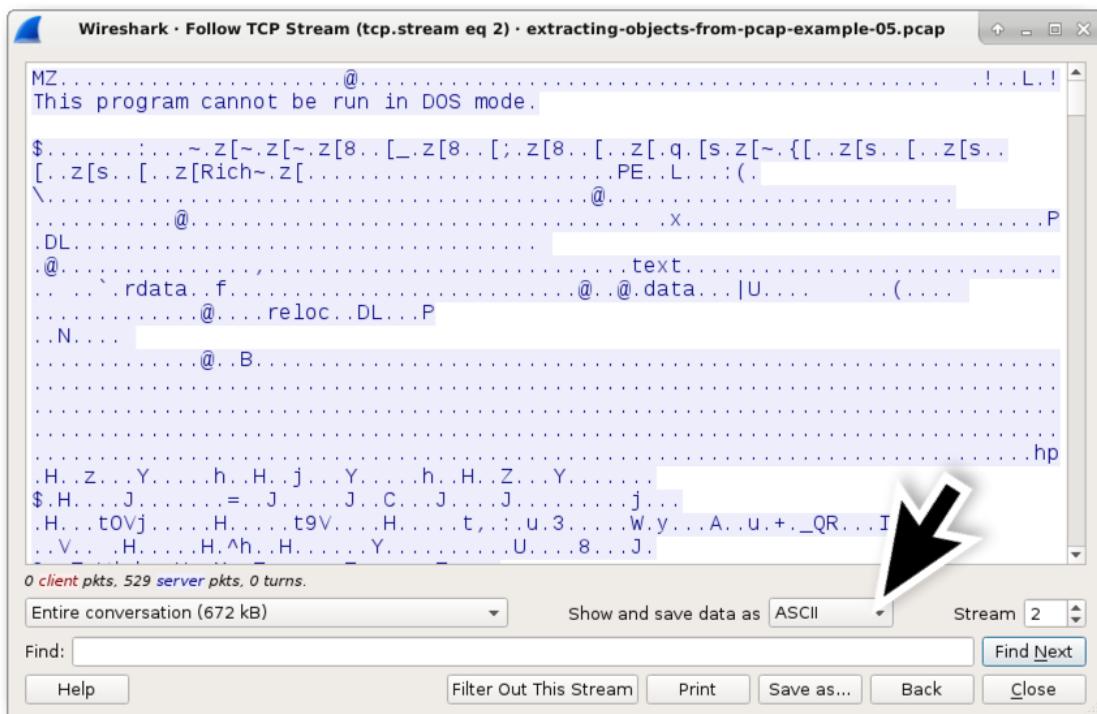


Figure 17. The TCP stream window for **q.exe**. Note the "Show and save data as" button-style menu.

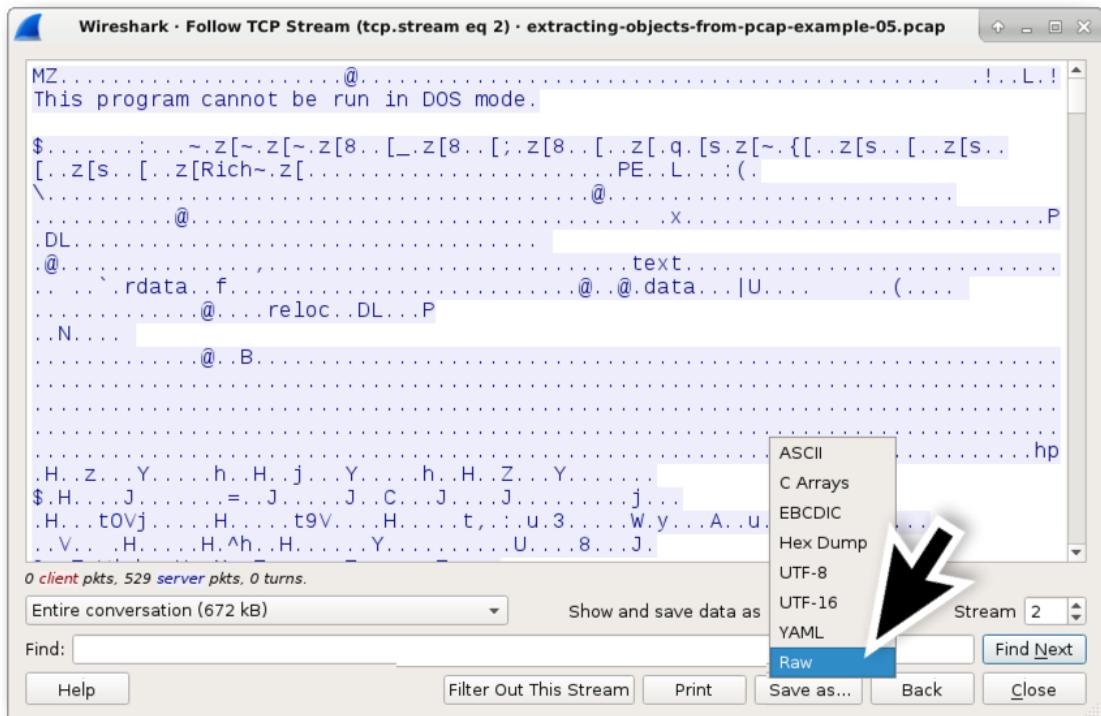


Figure 18. Selecting "Raw" from the "Show and save data as" menu.

The window should now show hexadecimal characters instead of ASCII as shown in Figure 19. Use the **Save as...** button near the bottom of the window to save this as a raw binary, also shown in Figure 19.

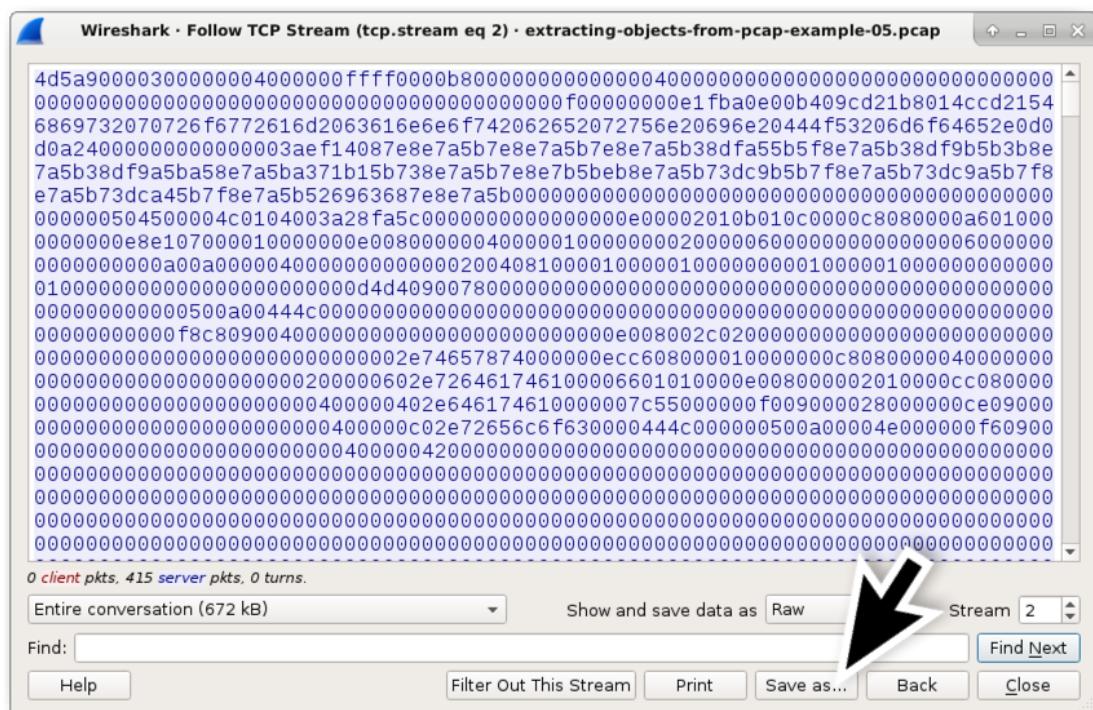


Figure 19. Saving the data from a TCP stream as a raw binary.

Save the file as **q.exe**. In a Linux or similar CLI environment, confirm this is a Windows executable file and get the SHA256 hash as shown below.

```
$ file q.exe  
q.exe: PE32 executable (GUI) Intel 80386, for MS  
Windows  
  
$ shasum -a 256 q.exe  
  
ca34b0926cdc3242bbfad1c4a0b42cc2750d90db9a272d92cfb6  
cb7034d2a3bd q.exe
```

The SHA256 hash shows a high detection rate as malware [on VirusTotal](#). Follow the same process for the other .exe files in the pcap:

- Filter on **ftp-data**
- Follow the TCP stream for a TCP segment with the name of your file in the Info column
- Change "Show and save data as" to "Raw"
- Use the "Save as..." button to save the file
- Check to make sure your saved file is, in fact, a Windows executable file.

This should give you the following files as shown below in Table 3.

| SHA256 hash |
|--|
| 32e1b3732cd779af1bf7730d0ec8a7a87a084319f6a0870dc7362a15ddbd3199 |
| ca34b0926cdc3242bbfad1c4a0b42cc2750d90db9a272d92cfb6cb7034d2a3bd |
| 4ebd58007ee933a0a8348aee2922904a7110b7fb6a316b1c7fb2c6677e613884 |
| 10ce4b79180a2ddd924fdc95951d968191af2ee3b7dfc96dd6a5714dbeae613a |
| 08eb941447078ef2c6ad8d91bb2f52256c09657ecd3d5344023edccf7291e9fc |

Table 3. Executable files from the FTP traffic.

We have to search more precisely when exporting the HTML files sent from the infected Windows host back to the FTP server. Why? Because the same file name is used each time. Filter on ***ftp.request.command***, and scroll to the end. We can see the same file name used to store (STOR) stolen data to the FTP server as an HTML file as shown in Figure 20.

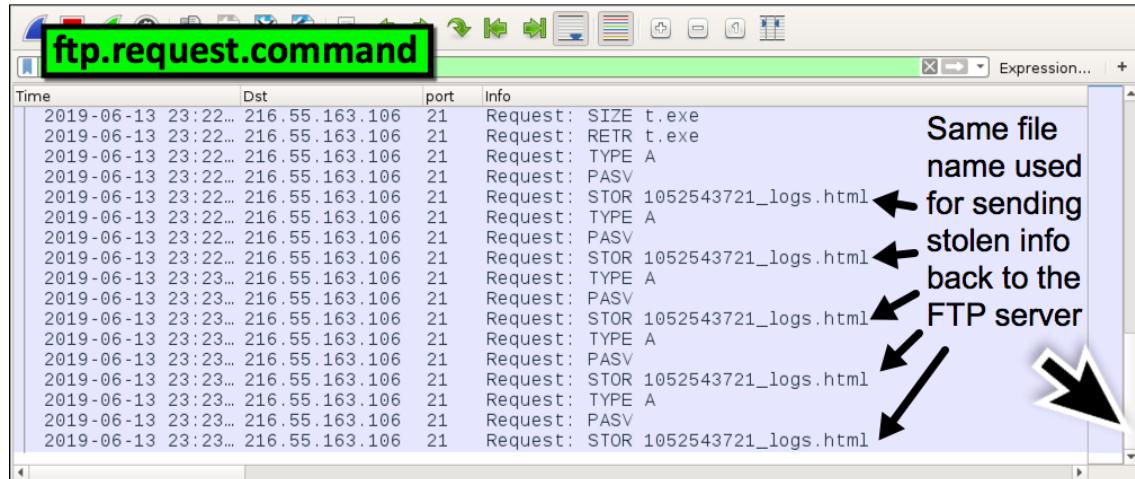


Figure 20. The same file name used for sending stolen info back to the FTP server.

To see the associated files sent over the ftp data channel, use the filter ***ftp-data.command contains .html*** as shown in Figure 21.

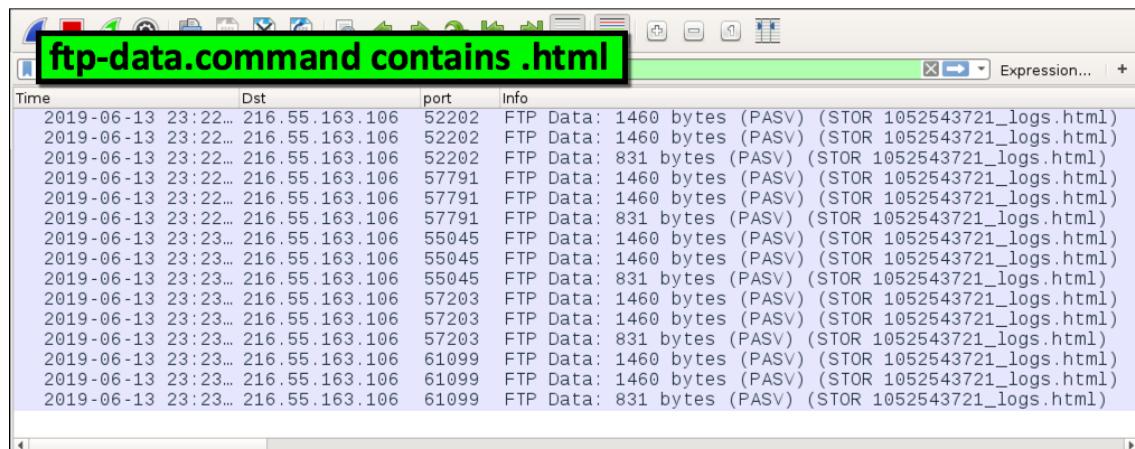


Figure 21. Filtering on files with *.html* in the file name over the FTP data channel.

In Figure 21, the destination port changes each time the file is stored (STOR) to the FTP server. The first time has TCP port 52202. The second time has TCP port 57791. The third time has TCP port 55045. The fourth time has 57203. And the fifth time has 61099.

We use the same process as before. Instead of focusing on the file names, focus on the TCP ports. Follow the TCP stream for any of the TCP segments using port 52202. In the TCP stream window, change

"Show and save data as" to "Raw." Then save the file. Do the same for the HTML file over TCP port 57791.

If you do this for all five HTML files, you'll find they are the same exact file. These text-based HTML files contain data about the infected Windows host, including any passwords found by the malware.

<https://unit42.paloaltonetworks.com/using-wireshark-exporting-objects-from-a-pcap/>

https://www.youtube.com/watch?v=MB8KHnhpkY0&ab_channel=LauraChappell

ElasticSearch, Kibana and Logstash (ELK)

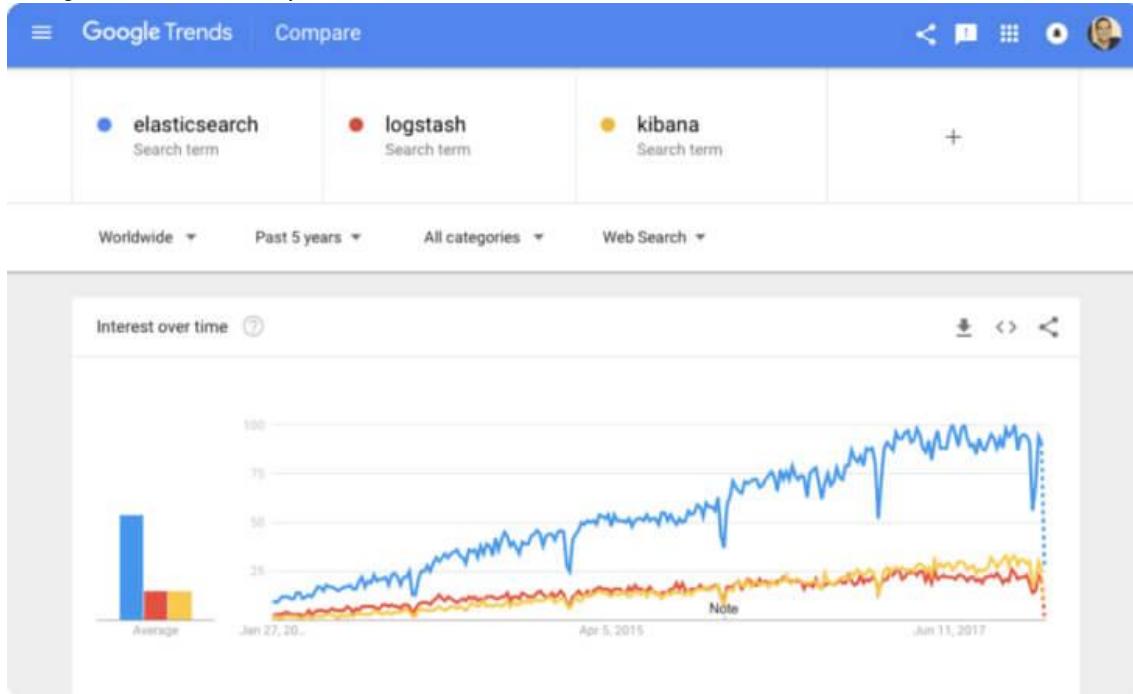
What is the ELK Stack?

Up until a year or two ago, the ELK Stack was a collection of three open-source products — Elasticsearch, Logstash, and Kibana — all developed, managed and maintained by Elastic. The introduction and subsequent addition of Beats turned the stack into a four legged project.

Elasticsearch is an open source, full-text search and analysis engine, based on the Apache Lucene search engine. Logstash is a log aggregator that collects data from various input sources, executes different transformations and enhancements and then ships the data to various supported output destinations. Kibana is a visualization layer that works on top of Elasticsearch, providing users with the ability to analyze and visualize the data. And last but not least — Beats are lightweight agents that are installed on edge hosts to collect different types of data for forwarding into the stack.

Together, these different components are most commonly used for monitoring, troubleshooting and securing IT environments (though there are many more use cases for the ELK Stack such as business intelligence and web analytics). Beats and Logstash take care of data collection and processing, Elasticsearch indexes and stores the data, and Kibana provides a user interface for querying the data and visualizing it.

Why is ELK So Popular?



The ELK Stack is popular because it fulfills a need in the log management and analytics space. Monitoring modern applications and the IT infrastructure they are deployed on requires a log management and analytics solution that enables engineers to overcome the challenge of monitoring what are highly distributed, dynamic and noisy environments.

The ELK Stack helps by providing users with a powerful platform that collects and processes data from multiple data sources, stores that data in one centralized data store that can scale as data grows, and that provides a set of tools to analyze the data.

Of course, the ELK Stack is open source. With IT organizations favoring open source products, this alone could explain the popularity of the stack. Using open source means organizations can avoid vendor lock-in and onboard new talent much more easily. Everyone knows how to use Kibana, right? Open source also means a vibrant community constantly driving new features and innovation and helping out in case of need.

Sure, Splunk has long been a market leader in the space. But its numerous functionalities are increasingly not worth the expensive price — especially

for smaller companies such as SaaS products and tech startups. Splunk has about 15,000 customers while ELK is downloaded more times in a single month than Splunk's total customer count — and many times over at that. ELK might not have all of the features of Splunk, but it does not need those analytical bells and whistles. ELK is a simple but robust log management and analytics platform that costs a fraction of the price.

Why is Log Analysis Becoming More Important?

In today's competitive world, organizations cannot afford one second of downtime or slow performance of their applications. Performance issues can damage a brand and in some cases translate into a direct revenue loss. For the same reason, organizations cannot afford to be compromised as well, and not complying with regulatory standards can result in hefty fines and damage a business just as much as a performance issue.

To ensure apps are available, performant and secure at all times, engineers rely on the different types of data generated by their applications and the infrastructure supporting them. This data, whether event logs or metrics, or both, enables monitoring of these systems and the identification and resolution of issues should they occur.

Logs have always existed and so have the different tools available for analyzing them. What has changed, though, is the underlying architecture of the environments generating these logs. Architecture has evolved into microservices, containers and orchestration infrastructure deployed on the cloud, across clouds or in hybrid environments. Not only that, the sheer volume of data generated by these environments is constantly growing and constitutes a challenge in itself. Long gone are the days when an engineer could simply SSH into a machine and grep a log file. This cannot be done in environments consisting of hundreds of containers generating TBs of log data a day.

This is where centralized log management and analytics solutions such as the ELK Stack come into the picture, allowing engineers, whether DevOps, IT Operations or SREs, to gain the visibility they need and ensure apps are available and performant at all times.

Modern log management and analysis solutions include the following key capabilities:

- Aggregation – the ability to collect and ship logs from multiple data sources.
- Processing – the ability to transform log messages into meaningful data for easier analysis.
- Storage – the ability to store data for extended time periods to allow for monitoring, trend analysis, and security use cases.
- Analysis – the ability to dissect the data by querying it and creating visualizations and dashboards on top of it.

How to Use the ELK Stack for Log Analysis

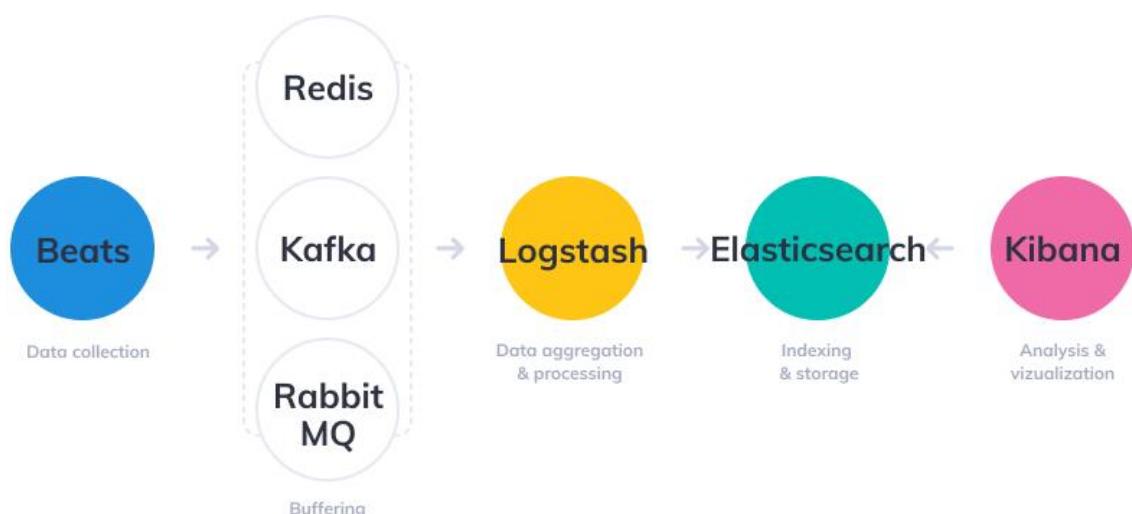
As I mentioned above, taken together, the different components of the ELK Stack provide a simple yet powerful solution for log management and analytics.

The various components in the ELK Stack were designed to interact and play nicely with each other without too much extra configuration. However, how you end up designing the stack greatly differs on your environment and use case.

For a small-sized development environment, the classic architecture will look as follows:



However, for handling more complex pipelines built for handling large amounts of data in production, additional components are likely to be added into your logging architecture, for resiliency (Kafka, RabbitMQ, Redis) and security (nginx):



This is of course a simplified diagram for the sake of illustration. A full production-grade architecture will consist of multiple Elasticsearch nodes, perhaps multiple Logstash instances, an archiving mechanism, an alerting plugin and a full replication across regions or segments of your data center for high availability. You can read a full description of what it takes to deploy ELK as a production-grade log management and analytics solution in the relevant section below.

What's new?

As one might expect from an extremely popular open source project, the ELK Stack is constantly and frequently updated with new features. Keeping abreast of these changes is challenging, so in this section we'll provide a highlight of the new features introduced in major releases.

Elasticsearch

Elasticsearch 7.x is much easier to setup since it now ships with Java bundled. Performance improvements include a real memory circuit breaker, improved search performance and a 1-shard policy. In addition, a new cluster coordination layer makes Elasticsearch more scalable and resilient.

Logstash

Logstash's Java execution engine (announced as experimental in version 6.3) is enabled by default in version 7.x. Replacing the old Ruby execution engine, it boasts better performance, reduced memory usage and overall — an entirely faster experience.

Kibana

Kibana is undergoing some major facelifting with new pages and usability improvements. The latest release includes a dark mode, improved querying and filtering and improvements to Canvas.

Beats

Beats 7.x conform with the new Elastic Common Schema (ECS) — a new standard for field formatting. Metricbeat supports a new AWS module for pulling data from Amazon CloudWatch, Kinesis and SQS. New modules were introduced in Filebeat and Auditbeat as well.

Installing ELK

The ELK Stack can be installed using a variety of methods and on a wide array of different operating systems and environments. ELK can be installed locally, on the cloud, using Docker and configuration management systems like Ansible, Puppet, and Chef. The stack can be installed using a tarball or .zip packages or from repositories.

Many of the installation steps are similar from environment to environment and since we cannot cover all the different scenarios, we will provide an example for installing all the components of the stack — Elasticsearch, Logstash, Kibana, and Beats — on Linux. Links to other installation guides can be found below.

Environment specifications

To perform the steps below, we set up a single AWS Ubuntu 18.04 machine on an m4.large instance using its local storage. We started an EC2 instance in the public subnet of a VPC, and then we set up the security group (firewall) to enable access from anywhere using SSH and TCP 5601 (Kibana). Finally, we added a new elastic IP address and associated it with our running instance in order to connect to the internet.

Please note that the version we installed here is 6.2. Changes have been made in more recent versions to the licensing model, including the inclusion of basic X-Pack features into the default installation packages.

Installing Elasticsearch

First, you need to add Elastic's signing key so that the downloaded package can be verified (skip this step if you've already installed packages from Elastic):

```
 wget -qO - https://artifacts.elastic.co/GPG-KEY-  
 elasticsearch | sudo apt-key add -
```

Copy

For Debian, we need to then install the apt-transport-https package:

```
sudo apt-get update sudo apt-get install apt-transport-https
```

Copy

The next step is to add the repository definition to your system:

```
echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main" |  
sudo tee -a /etc/apt/sources.list.d/elastic-7.x.list
```

Copy

To install a version of Elasticsearch that contains only features licensed under Apache 2.0 (aka OSS Elasticsearch):

```
echo "deb https://artifacts.elastic.co/packages/oss-7.x/apt stable main" |  
sudo tee -a /etc/apt/sources.list.d/elastic-7.x.list
```

Copy

All that's left to do is to update your repositories and install Elasticsearch:

```
sudo apt-get update sudo apt-get install elasticsearch
```

Copy

Elasticsearch configurations are done using a configuration file that allows you to configure general settings (e.g. node name), as well as network settings (e.g. host and port), where data is stored, memory, log files, and more.

For our example, since we are installing Elasticsearch on AWS, it is a good best practice to bind Elasticsearch to either a private IP or localhost:

```
sudo vim /etc/elasticsearch/elasticsearch.yml  
network.host:  
"localhost" http.port:9200 cluster.initial_master_nodes:  
["<PrivateIP>"]
```

Copy

To run Elasticsearch, use:

```
sudo service elasticsearch start
```

Copy

To confirm that everything is working as expected, point curl or your browser to <http://localhost:9200>, and you should see something like the following output:

```
{ "name" : "ip-172-31-10-207", "cluster_name" : "elasticsearch",  
"cluster_uuid" : "bzFHfhcoTAKCH-Niq6_GEA", "version" : { "number" :  
"7.1.1", "build_flavor" : "default", "build_type" : "deb", "build_hash" :  
"7a013de", "build_date" : "2019-05-23T14:04:00.380842Z",  
"build_snapshot" : false, "lucene_version" : "8.0.0",  
"minimum_wire_compatibility_version" : "6.8.0",  
"minimum_index_compatibility_version" : "6.0.0-beta1" }, "tagline" :  
"You Know, for Search" }
```

Copy

Installing an Elasticsearch cluster requires a different type of setup. Read our [Elasticsearch Cluster tutorial](#) for more information on that.

Installing Logstash

Logstash requires Java 8 or Java 11 to run so we will start the process of setting up Logstash with:

```
sudo apt-get install default-jre
```

Copy

Verify java is installed:

```
java -version  
openjdk version "1.8.0_191" OpenJDK Runtime Environment (build 1.8.0_191-8u191-b12-2ubuntu0.16.04.1-b12)  
OpenJDK 64-Bit Server VM (build 25.191-b12, mixed mode)
```

Copy

Since we already defined the repository in the system, all we have to do to install Logstash is run:

```
sudo apt-get install logstash
```

Copy

Before you run Logstash, you will need to configure a data pipeline. We will get back to that once we've installed and started Kibana.

Installing Kibana

As before, we will use a simple apt command to install Kibana:

```
sudo apt-get install kibana
```

Copy

Open up the Kibana configuration file at: `/etc/kibana/kibana.yml`, and make sure you have the following configurations defined:

```
server.port: 5601 elasticsearch.url: "http://localhost:9200"
```

Copy

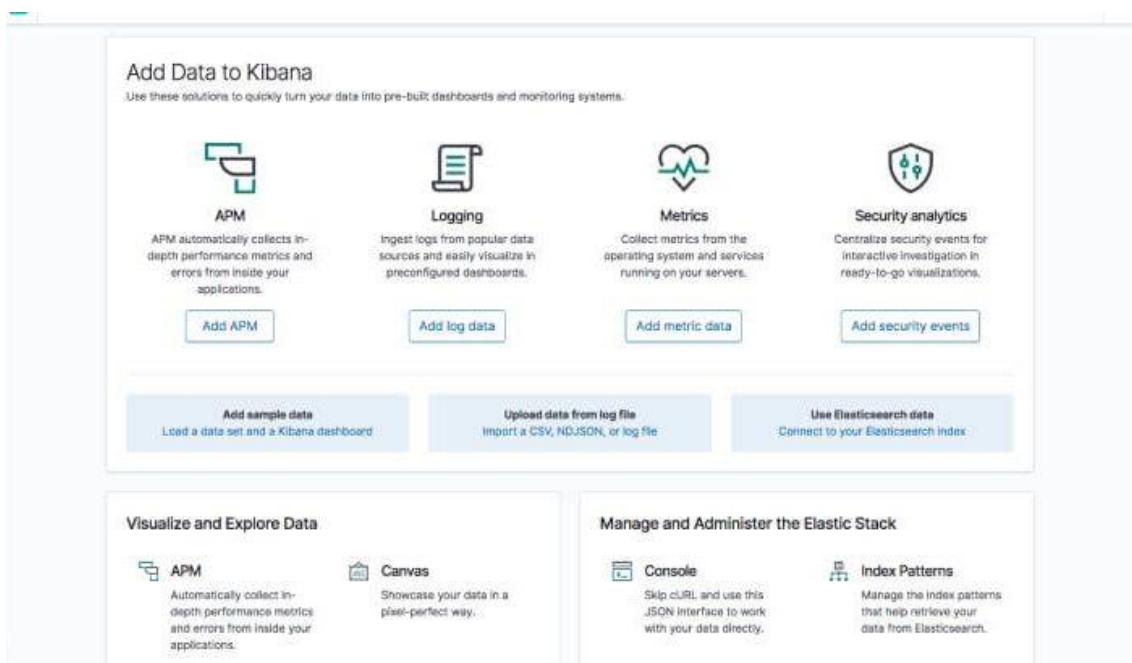
These specific configurations tell Kibana which Elasticsearch to connect to and which port to use.

Now, start Kibana with:

```
sudo service kibana start
```

Copy

Open up Kibana in your browser with: `http://localhost:5601`. You will be presented with the Kibana home page.



Installing Beats

The various shippers belonging to the Beats family can be installed in exactly the same way as we installed the other components.

As an example, let's install Metricbeat:

```
sudo apt-get install metricbeat
```

Copy

To start Metricbeat, enter:

```
sudo service metricbeat start
```

Copy

Metricbeat will begin monitoring your server and create an Elasticsearch index which you can define in Kibana. In the next step, however, we will describe how to set up a data pipeline using Logstash.

More information on using the different beats is available on our blog:

○ [Filebeat](#)

○ [Metricbeat](#)

○ [Winlogbeat](#)

○ [Auditbeat](#)

Shipping some data

For the purpose of this tutorial, we've prepared some sample data containing Apache access logs that is refreshed daily. You can download the data here: [sample-data](#)

Next, create a new Logstash configuration file at:

/etc/logstash/conf.d/apache-01.conf:

```
sudo vim /etc/logstash/conf.d/apache-01.conf
```

Copy

Enter the following Logstash configuration (change the path to the file you downloaded accordingly):

```
input { file { path => "/home/ubuntu/apache-daily-access.log"
  start_position => "beginning" since_db_path => "/dev/null" } }
filter { grok { match => { "message" =>
  "%{COMBINEDAPACHELOG}" } } date { match => [ "timestamp" ,
  "dd/MMM/yyyy:HH:mm:ss Z" ] } geoip { source => "clientip" } }
output { elasticsearch { hosts => ["localhost:9200"] } }
```

Copy

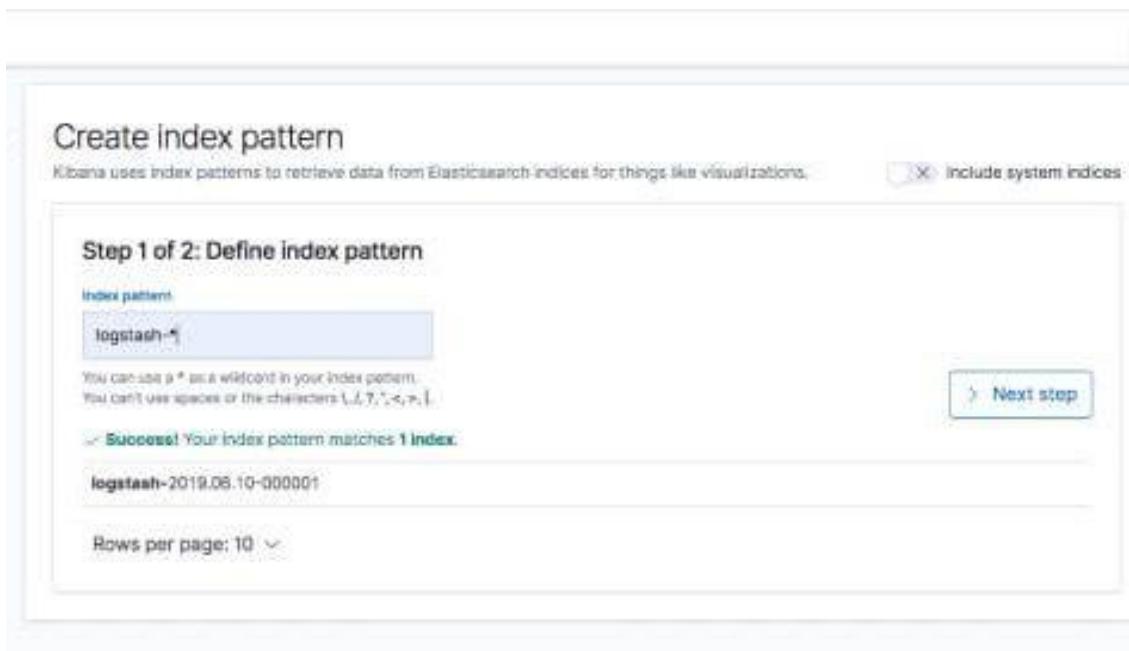
Start Logstash with:

```
sudo service logstash start
```

Copy

If all goes well, a new Logstash index will be created in Elasticsearch, the pattern of which can now be defined in Kibana.

In Kibana, go to **Management → Kibana Index Patterns**. Kibana should display the Logstash index and along with the Metricbeat index if you followed the steps for installing and running Metricbeat).



Enter “logstash-*” as the index pattern, and in the next step select @timestamp as your Time Filter field.

Create index pattern

Kibana uses index patterns to retrieve data from Elasticsearch indices for things like visualizations.

Include system indices

Step 2 of 2: Configure settings

You've defined logstash-* as your index pattern. Now you can specify some settings before we create it.

Time Filter field name

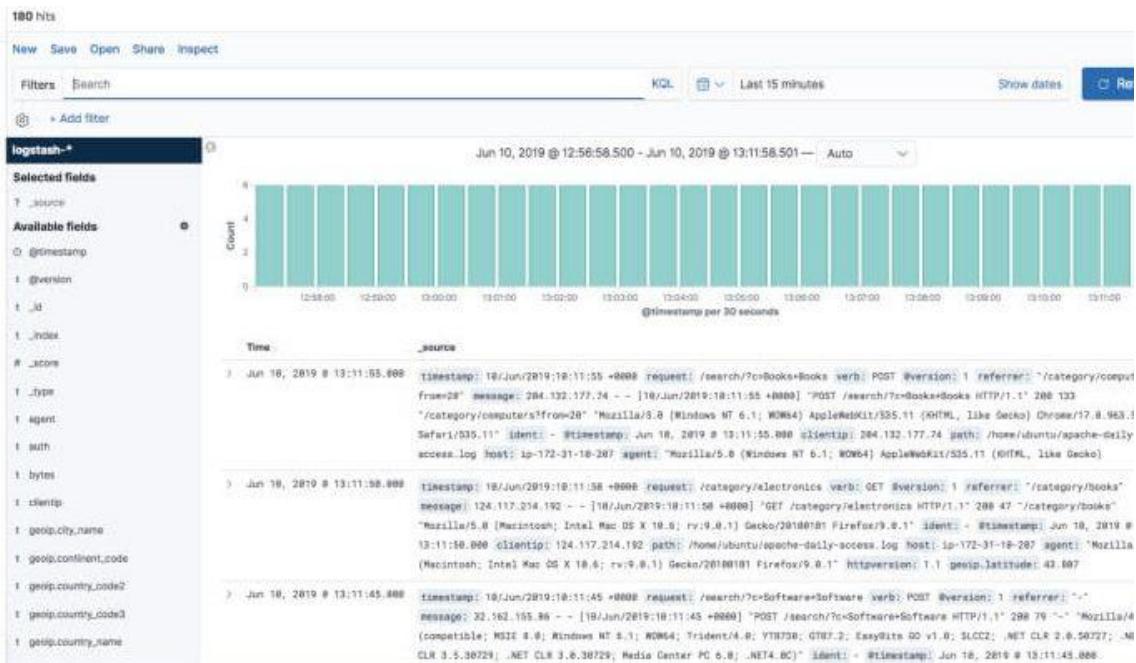
@timestamp

The Time Filter will use this field to filter your data by time. You can choose not to have a time field, but you will not be able to narrow down your data by a time range.

Show advanced options

< Back

Hit **Create index pattern**, and you are ready to analyze the data. Go to the Discover tab in Kibana to take a look at the data (look at today's data instead of the default last 15 mins).



Congratulations! You have set up your first ELK data pipeline using Elasticsearch, Logstash, and Kibana.

Additional installation guides

As mentioned before, this is just one environment example of installing ELK. There are other systems and platforms covered in other articles on our blog that might be relevant for you:

○ [Installing ELK on Google Cloud Platform](#)

○ [Installing ELK on Azure](#)

○ [Installing ELK on Windows](#)

○ [Installing ELK with Docker](#)

○ [Installing ELK on Mac OS X](#)

○ [Installing ELK with Ansible](#)

○ [Installing ELK on RaspberryPi](#)

Check out the other sections of this guide to understand more advanced topics related to working with Elasticsearch, Logstash, Kibana and Beats.

Elasticsearch

What is Elasticsearch?

Elasticsearch is the living heart of what is today the world's most popular log analytics platform — the ELK Stack ([Elasticsearch](#), [Logstash](#), and [Kibana](#)).

The role played by Elasticsearch is so central that it has become synonymous with the name of the stack itself. Used primarily for search and log analysis, Elasticsearch is today one of the [most popular database systems available](#) today.

Initially released in 2010, Elasticsearch is a modern search and analytics engine which is based on Apache Lucene. Completely open source and built with Java, Elasticsearch is categorized as a NoSQL database. Elasticsearch stores data in an unstructured way, and up until recently you could not

query the data using SQL. The new Elasticsearch SQL project will allow using SQL statements to interact with the data. You can read more on that in [this article](#).

Unlike most NoSQL databases, though, Elasticsearch has a strong focus on search capabilities and features — so much so, in fact, that the easiest way to get data from Elasticsearch is to search for it using its extensive [REST API](#).

In the context of data analysis, Elasticsearch is used together with the other components in the ELK Stack, Logstash and Kibana, and plays the role of data indexing and storage.

Read more about installing and using Elasticsearch in our [Elasticsearch tutorial](#).

Basic Elasticsearch Concepts

Elasticsearch is a feature-rich and complex system. Detailing and drilling down into each of its nuts and bolts is impossible. However, there are some basic concepts and terms that all Elasticsearch users should learn and become familiar with. Below are the six “must-know” concepts to start with.

Index

Elasticsearch Indices are logical partitions of documents and can be compared to a database in the world of relational databases.

Continuing our e-commerce app example, you could have one index containing all of the data related to the products and another with all of the data related to the customers.

You can have as many indices defined in Elasticsearch as you want but this can affect performance. These, in turn, will hold documents that are unique to each index.

Indices are identified by lowercase names that are used when performing various actions (such as searching and deleting) against the documents that are inside each index.

Documents

Documents are JSON objects that are stored within an Elasticsearch index and are considered the base unit of storage. In the world of relational databases, documents can be compared to a row in a table.

In the example of our e-commerce app, you could have one document per product or one document per order. There is no limit to how many documents you can store in a particular index.

Data in documents is defined with fields comprised of keys and values. A key is the name of the field, and a value can be an item of many different types such as a string, a number, a boolean expression, another object, or an array of values.

Documents also contain reserved fields that constitute the document metadata such as `_index`, `_type` and `_id`.

Types

Elasticsearch types are used within documents to subdivide similar types of data wherein each type represents a unique class of documents. Types consist of a name and a mapping (see below) and are used by adding the `_type` field. This field can then be used for filtering when querying a specific type.

Types are gradually being removed from Elasticsearch. Starting with Elasticsearch 6, indices can have only one mapping type. Starting in version 7.x, specifying types in requests is deprecated. Starting in version 8.x, specifying types in requests will no longer be supported.

Mapping

Like a schema in the world of relational databases, mapping defines the different types that reside within an index. It defines the fields for documents of a specific type — the data type (such as string and integer) and how the fields should be indexed and stored in Elasticsearch.

A mapping can be defined explicitly or generated automatically when a document is indexed using templates. (Templates include settings and mappings that can be applied automatically to a new index.)

Shards

Index size is a common cause of Elasticsearch crashes. Since there is no limit to how many documents you can store on each index, an index may take up an amount of disk space that exceeds the limits of the hosting server. As soon as an index approaches this limit, indexing will begin to fail.

One way to counter this problem is to split up indices horizontally into pieces called shards. This allows you to distribute operations across shards and nodes to improve performance. You can control the amount of shards per index and host these “index-like” shards on any node in your Elasticsearch cluster.

Replicas

To allow you to easily recover from system failures such as unexpected downtime or network issues, Elasticsearch allows users to make copies of shards called replicas. Because replicas were designed to ensure high availability, they are not allocated on the same node as the shard they are copied from. Similar to shards, the number of replicas can be defined when creating the index but also altered at a later stage.

For more information on these terms and additional Elasticsearch concepts, read the [10 Elasticsearch Concepts You Need To Learn](#) article.

Elasticsearch Queries

Elasticsearch is built on top of Apache Lucene and exposes Lucene's query syntax. Getting acquainted with the syntax and its various operators will go a long way in helping you query Elasticsearch.

Boolean Operators

As with most computer languages, Elasticsearch supports the AND, OR, and NOT operators:

- **jack AND jill** — Will return events that contain both jack and jill
- **ahab NOT moby** — Will return events that contain ahab but not moby
- **tom OR jerry** — Will return events that contain tom or jerry, or both

Fields

You might be looking for events where a specific field contains certain terms. You specify that as follows:

- **name:"Ned Stark"**

Ranges

You can search for fields within a specific range, using square brackets for inclusive range searches and curly braces for exclusive range searches:

- **age:[3 TO 10]** — Will return events with age between 3 and 10
- **price:{100 TO 400}** — Will return events with prices between 101 and 399
- **name:[Adam TO Ziggy]** — Will return names between and including Adam and Ziggy

Wildcards, Regexes and Fuzzy Searching

A search would not be a search without the wildcards. You can use the * character for multiple character wildcards or the ? character for single character wildcards.

URI Search

The easiest way to search your Elasticsearch cluster is through URI search. You can pass a simple query to Elasticsearch using the *q* query parameter. The following query will search your whole cluster for documents with a name field equal to “travis”:

- **curl “localhost:9200/_search?q=name:travis”**

Combined with the Lucene syntax, you can build quite impressive searches. Usually, you’ll have to URL-encode characters such as spaces (it’s been omitted in these examples for clarity):

- **curl “localhost:9200/_search?q=name:john~1 AND (age:[30 TO 40} OR surname:K*) AND -city”**

A number of options are available that allow you to customize the URI search, specifically in terms of which analyzer to use (analyzer), whether the query should be fault-tolerant (lenient), and whether an explanation of the scoring should be provided (explain).

Although the URI search is a simple and efficient way to query your cluster, you’ll quickly find that it doesn’t support all of the features offered to you by Elasticsearch. The full power of Elasticsearch is exposed through Request Body Search. Using Request Body Search allows you to build a complex search request using various elements and query clauses that will match, filter, and order as well as manipulate documents based on multiple criteria.

More information on Request Body Search in Elasticsearch, Query DSL and examples can be found in our: [Elasticsearch Queries: A Thorough Guide](#).

Elasticsearch REST API

One of the great things about Elasticsearch is its extensive REST API which allows you to integrate, manage and query the indexed data in countless different ways. Examples of using this API to integrate with Elasticsearch data are abundant, spanning different companies and use cases.

Interacting with the API is easy — you can use any HTTP client but Kibana comes with a built-in tool called Console which can be used for this purpose.



The screenshot shows the Kibana Console interface. At the top, there are three tabs: 'Console' (which is selected), 'Search Profiler', and 'Grok Debugger'. Below the tabs, a code editor displays a JSON search query. The response from the server is shown in a larger panel below the query. The response is a JSON object with the following structure:

```
1 { "took" : 5,
2   "timed_out" : false,
3   "_shards" : {
4     "total" : 3,
5     "successful" : 3,
6     "skipped" : 0,
7     "failed" : 0
8   },
9   "hits" : {
10    "total" : {
11      "value" : 10000,
12      "relation" : "gte"
13    },
14    "max_score" : 1.0,
15    "hits" : [
16      {
17        "_index" : ".kibana_1",
18        "_type" : "_doc",
19        "_id" : "space:default",
20        "_score" : 1.0,
21        "_source" : {
22          "space" : {
23            "name" : "Default",
24            "description" : "This is your default space!",
25            "color" : "#00bfb3",
26            "_reserved" : true
27          },
28          "type" : "space",
29          "references" : [ ],
30          "updated_at" : "2019-06-10T09:06:42.669Z"
31        }
32      },
33    ]
34 }
```

As extensive as Elasticsearch REST APIs are, there is a learning curve. To get started, read the API conventions, learn about the different options that can be applied to the calls, how to construct the APIs and how to filter responses. A good thing to remember is that some APIs change and get deprecated from version to version, and it's a good best practice to keep tabs on breaking changes.

Below are some of the most common Elasticsearch API categories worth researching. Usage examples are available in the [Elasticsearch API 101](#) article. Of course, [Elasticsearch official documentation](#) is an important resource as well.

Elasticsearch Document API

This category of APIs is used for handling documents in Elasticsearch. Using these APIs, for example, you can create documents in an index, update them, move them to another index, or remove them.

Elasticsearch Search API

As its name implies, these API calls can be used to query indexed data for specific information. Search APIs can be applied globally, across all available indices and types, or more specifically within an index. Responses will contain matches to the specific query.

Elasticsearch Indices API

This type of Elasticsearch API allows users to manage indices, mappings, and templates. For example, you can use this API to create or delete a new index, check if a specific index exists or not, and define a new mapping for an index.

Elasticsearch Cluster API

These are cluster-specific API calls that allow you to manage and monitor your Elasticsearch cluster. Most of the APIs allow you to define which Elasticsearch node to call using either the internal node ID, its name or its address.

Elasticsearch Plugins

Elasticsearch plugins are used to extend the basic Elasticsearch functionality in various, specific ways. There are plugins, for example, that add security functionality, discovery mechanisms, and analysis capabilities to Elasticsearch.

Regardless of what functionalities they add, Elasticsearch plugins belong to either of the following two categories: [core plugins](#) or community plugins.

The former is supplied as part of the Elasticsearch package and are maintained by the Elastic team while the latter is developed by the community and are thus separate entities with their own versioning and development cycles.

Plugin Categories

- API Extension
- Alerting
- Analysis
- Discovery
- Ingest
- Management
- Mapper
- Security
- Snapshot/Restore
- Store

Installing Elasticsearch Plugins

Installing core plugins is simple and is done using a plugin manager. In the example below, I'm going to install the EC2 Discovery plugin. This plugin queries the AWS API for a list of EC2 instances based on parameters that you define in the plugin settings :

```
cd /usr/share/elasticsearch sudo bin/elasticsearch-plugin  
install discovery-ec2
```

Copy

Plugins must be installed on every node in the cluster, and each node must be restarted after installation.

To remove a plugin, use:

```
sudo bin/elasticsearch-plugin remove discovery-ec2
```

Copy

Community plugins are a bit different as each of them has different installation instructions.

Some community plugins are installed the same way as core plugins but require additional Elasticsearch configuration steps.

What's next?

We described Elasticsearch, detailed some of its core concepts and explained the REST API. To continue learning about Elasticsearch, here are some resources you may find useful:

○ [An Elasticsearch Tutorial: Getting Started](#)

○ [Elasticsearch Cheatsheet](#)

○ [Elasticsearch Queries: A Thorough Guide](#)

○ [How to Avoid and Fix the Top 5 Elasticsearch Mistakes](#)

Logstash

Efficient log analysis is based on well-structured logs. The structure is what enables you to more easily search, analyze and visualize the data in whatever logging tool you are using. Structure is also what gives your data context. If possible, this structure needs to be tailored to the logs on the application level. In other cases, infrastructure and system logs, for example, it is up to you to give logs their structure by parsing them.

What is Logstash?

In the ELK Stack ([Elasticsearch](#), [Logstash](#) and [Kibana](#)), the crucial task of parsing data is given to the “L” in the stack – Logstash.

Logstash started out as an open source tool developed to handle the streaming of a large amount of log data from multiple sources. After being incorporated into the ELK Stack, it developed into the stack's workhorse, in charge of also processing the log messages, enhancing them and massaging them and then dispatching them to a defined destination for storage (stashing).

Thanks to a large ecosystem of plugins, Logstash can be used to collect, enrich and transform a wide array of different data types. There are over 200 different plugins for Logstash, with a vast community making use of its extensible features.

It has not always been smooth sailing for Logstash. Due to some inherent performance issues and design flaws, Logstash has received a decent amount of complaints from users over the years. Side projects were developed to alleviate some of these issues (e.g. Lumberjack, Logstash-Forwarder, Beats), and alternative log aggregators began competing with Logstash.

Yet despite these flaws, Logstash still remains a crucial component of the stack. Big steps have been made to try and alleviate these pains by introducing improvements to Logstash itself, such as a brand new execution engine made available in version 7.0, all ultimately helping to make logging with ELK much more reliable than what it used to be.

Read more about installing and using Logstash in our [Logstash tutorial](#).

Logstash Configuration

Events aggregated and processed by Logstash go through three stages: collection, processing, and dispatching. Which data is collected, how it is processed and where it is sent to, is defined in a Logstash configuration file that defines the pipeline.

Each of these stages is defined in the Logstash configuration file with what are called plugins — “Input” plugins for the data collection stage, “Filter” plugins for the processing stage, and “Output” plugins for the dispatching stage. Both the input and output plugins support [codecs](#) that allow you to encode or decode your data (e.g. json, multiline, plain).

Input plugins

One of the things that makes Logstash so powerful is its ability to aggregate logs and events from various sources. Using more than 50 input plugins for different platforms, databases and applications, Logstash can be defined to collect and process data from these sources and send them to other systems for storage and analysis.

The most common inputs used are: file, beats, syslog, http, tcp, udp, stdin, but you can ingest data from plenty of other sources.

Filter plugins

Logstash supports a number of extremely powerful filter plugins that enable you to enrich, manipulate, and process logs. It’s the power of these filters that makes Logstash a very versatile and valuable tool for parsing log data.

Filters can be combined with conditional statements to perform an action if a specific criterion is met.

The most common inputs used are: grok, date, mutate, drop. You can read more about these and other in [5 Logstash Filter Plugins](#).

Output plugins

As with the inputs, Logstash supports a number of output plugins that enable you to push your data to various locations, services, and technologies. You can store events using outputs such as File, CSV, and S3, convert them into

messages with RabbitMQ and SQS, or send them to various services like HipChat, PagerDuty, or IRC. The number of combinations of inputs and outputs in Logstash makes it a really versatile event transformer.

Logstash events can come from multiple sources, so it's important to check whether or not an event should be processed by a particular output. If you do not define an output, Logstash will automatically create a stdout output. An event can pass through multiple output plugins.

Logstash Codecs

Codecs can be used in both inputs and outputs. Input codecs provide a convenient way to decode your data before it enters the input. Output codecs provide a convenient way to encode your data before it leaves the output.

Some common codecs:

- The default “plain” codec is for plain text with no delimitation between events
- The “json” codec is for encoding JSON events in inputs and decoding json messages in outputs — note that it will revert to plain text if the received payloads are not in a valid JSON format
- The “json_lines” codec allows you either to receive and encode json events delimited by \n or to decode JSON messages delimited by \n in outputs
- The “rubydebug,” which is very useful in debugging, allows you to output Logstash events as data Ruby objects

Configuration example

Logstash has a simple configuration DSL that enables you to specify the inputs, outputs, and filters described above, along with their specific options. Order matters, specifically around filters and outputs, as the configuration is

basically converted into code and then executed. Keep this in mind when you're writing your configs, and try to debug them.

Input

The input section in the configuration file defines the input plugin to use. Each plugin has its own configuration options, which you should research before using.

Example:

```
input { file { path => "/var/log/apache/access.log" start_position => "beginning" } }
```

Copy

Here we are using the file input plugin. We entered the path to the file we want to collect, and defined the start position as beginning to process the logs from the beginning of the file.

Filter

The filter section in the configuration file defines what filter plugins we want to use, or in other words, what processing we want to apply to the logs. Each plugin has its own configuration options, which you should research before using.

Example:

```
filter { grok { match => { "message" => "%{COMBINEDAPACHELOG}" } } date { match => [ "timestamp" , "dd/MMM/yyyy:HH:mm:ss Z" ] } geoip { source => "clientip" } }
```

Copy

In this example we are processing Apache access logs applying:

- A *grok* filter that parses the log string and populates the event with the relevant information.

- A *date* filter to parse a date field which is a string as a *timestamp* field (each Logstash pipeline requires a timestamp so this is a required filter).
- A *geoip* filter to enrich the *clientip* field with geographical data. Using this filter will add new fields to the event (e.g. *countryname*) based on the *clientip* field.

Output

The output section in the configuration file defines the destination to which we want to send the logs to. As before, each plugin has its own configuration options, which you should research before using.

Example:

```
output { elasticsearch { hosts => ["localhost:9200"] } }
```

[Copy](#)

In this example, we are defining a locally installed instance of Elasticsearch.

Complete example

Putting it all together, the Logstash configuration file should look as follows:

```
input { file { path => "/var/log/apache/access.log" start_position => "beginning" } } filter { grok { match => { "message" => "%{COMBINEDAPACHELOG}" } } date { match => [ "timestamp" , "dd/MMM/yyyy:HH:mm:ss Z" ] } geoip { source => "clientip" } } output { elasticsearch { hosts => ["localhost:9200"] } }
```

[Copy](#)

Logstash pitfalls

As implied above, Logstash suffers from some inherent issues that are related to its design. Logstash requires JVM to run, and this dependency can be the root cause of significant memory consumption, especially when multiple pipelines and advanced filtering are involved.

Resource shortage, bad configuration, unnecessary use of plugins, changes in incoming logs — all of these can result in performance issues which can in turn result in data loss, especially if you have not put in place a safety net.

There are various ways to employ this safety net, both built into Logstash as well as some that involve adding middleware components to your stack. Here is a list of some best practices that will help you avoid some of the common Logstash pitfalls:

- Add a buffer – a recommended method involves adding a queuing layer between Logstash and the destination. The most popular methods use [Kafka](#), Redis and RabbitMQ.
- Persistent Queues – a built-in data resiliency feature in Logstash that allows you to store data in an internal queue on disk. Disabled by default — you need to enable the feature in the Logstash settings file.
- Dead Letter Queues – a mechanism for storing events that could not be processed on disk. Disabled by default — you need to enable the feature in the Logstash settings file.
- Keep it simple – try and keep your Logstash configuration as simple as possible. Don't use plugins if there is no need to do so.
- [Test your configs](#) – do not run your Logstash configuration in production until you've tested it in a sandbox environment. Use online tools to make sure it doesn't break your pipeline.

For additional pitfalls to look out for, refer to the [5 Logstash Pitfalls](#) article.

Monitoring Logstash

Logstash automatically records some information and metrics on the node running Logstash, JVM and running pipelines that can be used to monitor performance. To tap into this information, you can use [monitoring API](#).

For example, you can use the Hot Threads API to view Java threads with high CPU and extended execution times:

```
curl -XGET 'localhost:9600/_node/hot_threads?human=true' Hot
threads at 2019-05-27T08:43:05+00:00, busiestThreads=10:
=====
===== 3.16 % of cpu usage, state:
timed_waiting, thread name: 'Logstash::Runner', thread id: 1
java.base@11.0.3/java.lang.Object.wait(Native Method)
java.base@11.0.3/java.lang.Thread.join(Thread.java:1313)
app//org.jruby.internal.runtime.NativeThread.join(NativeThr
ead.java:75) -----
----- 0.61 % of cpu usage,
state: timed_waiting, thread name: '[main]>worker5', thread
id: 29
java.base@11.0.3/jdk.internal.misc.Unsafe.park(Native
Method)
java.base@11.0.3/java.util.concurrent.locks.LockSupport.par
kNanos(LockSupport.java:234)
java.base@11.0.3/java.util.concurrent.locks.AbstractQueuedS
ynchronizer$ConditionObject.awaitNanos(AbstractQueuedSynchroni
zer.java:2123) -----
----- 0.47 % of cpu usage,
state: timed_waiting, thread name: '[main]<file', thread id:
32 java.base@11.0.3/jdk.internal.misc.Unsafe.park(Native
Method)
java.base@11.0.3/java.util.concurrent.locks.LockSupport.par
kNanos(LockSupport.java:234)
java.base@11.0.3/java.util.concurrent.locks.AbstractQueuedS
ynchronizer.doAcquireSharedNanos(AbstractQueuedSynchronizer
.java:1079)
```

[Copy](#)

Alternatively, you can use monitoring UI within Kibana, available under Elastic's Basic license.

What next?

Logstash is a critical element in your ELK Stack, but you need to know how to use it both as an individual tool and together with the other components in the stack. Below is a list of other resources that will help you use Logstash.

○ [Logstash tutorial](#)

○ [How to debug Logstash configurations](#)

○ [A guide to Logstash plugins](#)

○ [Logstash filter plugins](#)

○ [Filebeat vs. Logstash](#)

○ [Kibana tutorial](#)

Did we miss something? Did you find a mistake? We're relying on your feedback to keep this guide up-to-date. Please add your comments at the bottom of the page, or send them to: elk-guide@logz.io

Kibana

No centralized logging solution is complete without an analysis and visualization tool. Without being able to efficiently query and monitor data, there is little use to only aggregating and storing it. Kibana plays that role in the ELK Stack — a powerful analysis and visualization layer on top of [Elasticsearch](#) and [Logstash](#).

What is Kibana?

Completely open source, Kibana is a browser-based user interface that can be used to search, analyze and visualize the data stored in Elasticsearch indices (Kibana cannot be used in conjunction with other databases). Kibana is especially renowned and popular due to its rich graphical and visualization capabilities that allow users to explore large volumes of data.

Kibana can be installed on Linux, Windows and Mac using .zip or tar.gz, repositories or on Docker. Kibana runs on node.js, and the installation packages come built-in with the required binaries. Read more about setting up Kibana in our [Kibana tutorial](#).

Please note that changes have been made in more recent versions to the licensing model, including the inclusion of basic X-Pack features into the default installation packages.

Kibana searches

Searching Elasticsearch for specific log message or strings within these messages is the bread and butter of Kibana. In recent versions of Kibana, improvements and changes to the way searching is done have been applied.

By default, users now use a new querying language called KQL (Kibana Querying Language) to search their data. Users accustomed to the previous method — using Lucene — can opt to do so as well.

Kibana querying is an art unto itself, and there are various methods you can use to perform searches on your data. Here are some of the most common search types:

- Free text searches – used for quickly searching for a specific string.
- Field-level searches – used for searching for a string within a specific field.
- Logical statements – used to combine searches into a logical statement.
- Proximity searches – used for searching terms within a specific character proximity.

For a more detailed explanation of the different search types, check out the [Kibana Tutorial](#).

Kibana searches cheat sheet

Below is a list of some tips and best practices for using the above-mentioned search types:

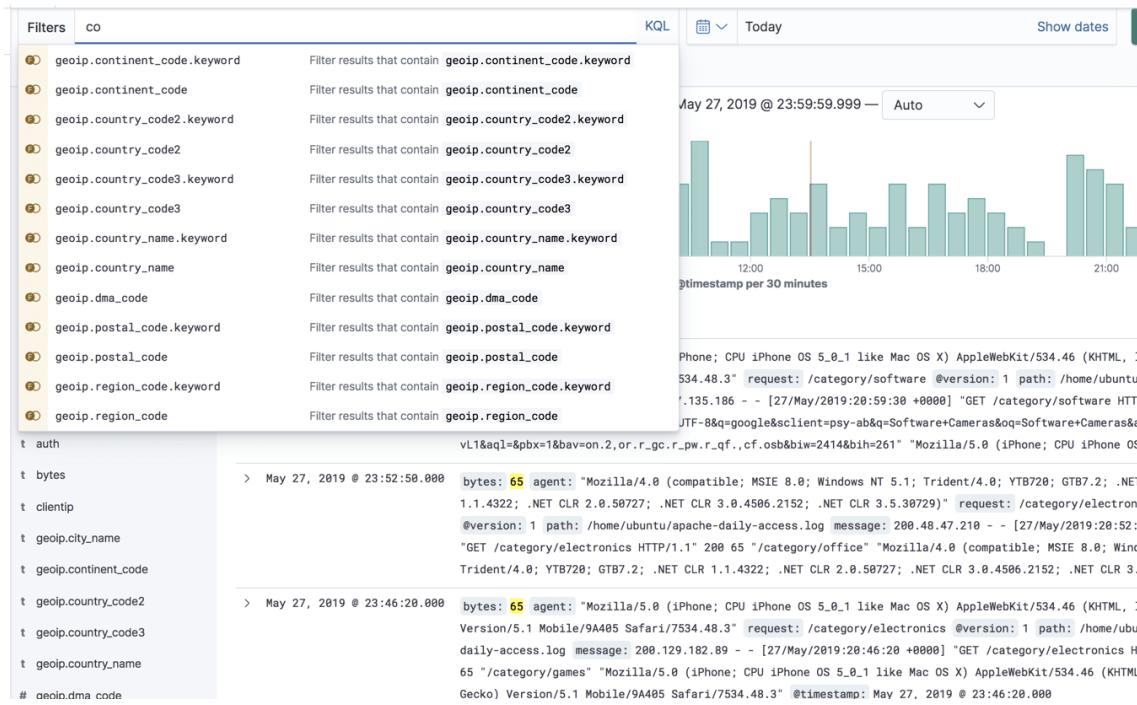
- Use free-text searches for quickly searching for a specific string. Use double quotes ("string") to look for an exact match.
Example: "USA"
- Use the * wildcard symbol to replace any number of characters and the ? wildcard symbol to replace only one character.
- Use the _exists_ prefix for a field to search for logs that have that field.
Example: _exists_:response
- You can search a range within a field.
Examples: If you use brackets [], this means that the results are inclusive. If you use {}, this means that the results are exclusive.
- When using logical statements (e.g. AND, OR, TO) within a search, use capital letters. Example: response:[400 TO 500]
- Use -,! and NOT to define negative terms.
Example: response:[400 TO 500] AND NOT response:404
- Proximity searches are useful for searching terms within a specific character proximity. Example: [categovi~2] will a search for all the terms that are within two changes from [categovi]. Proximity searches use a lot of resources – use wisely!
- Field level search for non analyzed fields work differently than free text search.
Example: If the field value is Error – searching for field:*rror will not return the right answer.
- If you don't specify a logical operator, the default one is OR.
Example: searching for Error Exception will run a search for Error OR Exception

- Using leading wildcards is a very expensive query and should be avoided when possible.

In Kibana 6.3, a new feature simplifies the search experience and includes auto-complete capabilities. This feature needs to be enabled for use, and is currently experimental.

Kibana autocomplete

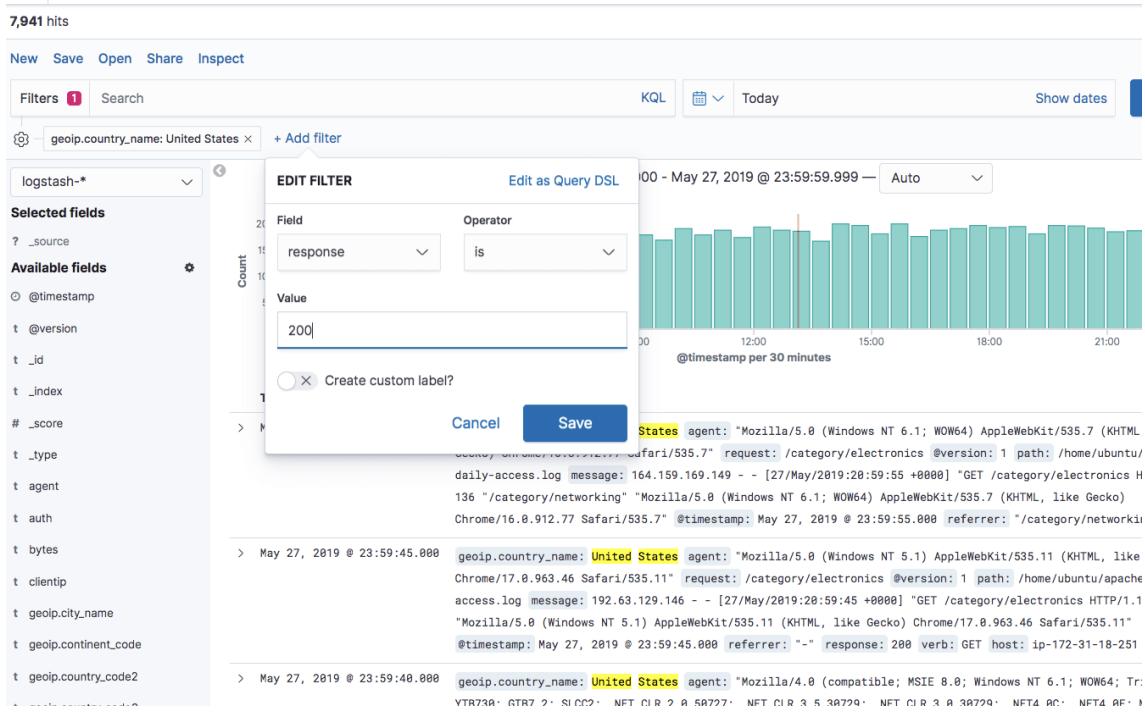
To help improve the search experience in Kibana, the autocomplete feature suggests search syntax as you enter your query. As you type, relevant fields are displayed and you can complete the query with just a few clicks. This speeds up the whole process and makes Kibana querying a whole lot simpler.



Kibana filtering

To assist users in searches, Kibana includes a filtering dialog that allows easier filtering of the data displayed in the main view.

To use the dialog, simply click the **Add a filter +** button under the search box and begin experimenting with the conditionals. Filters can be pinned to the Discover page, named using custom labels, enabled/disabled and inverted.



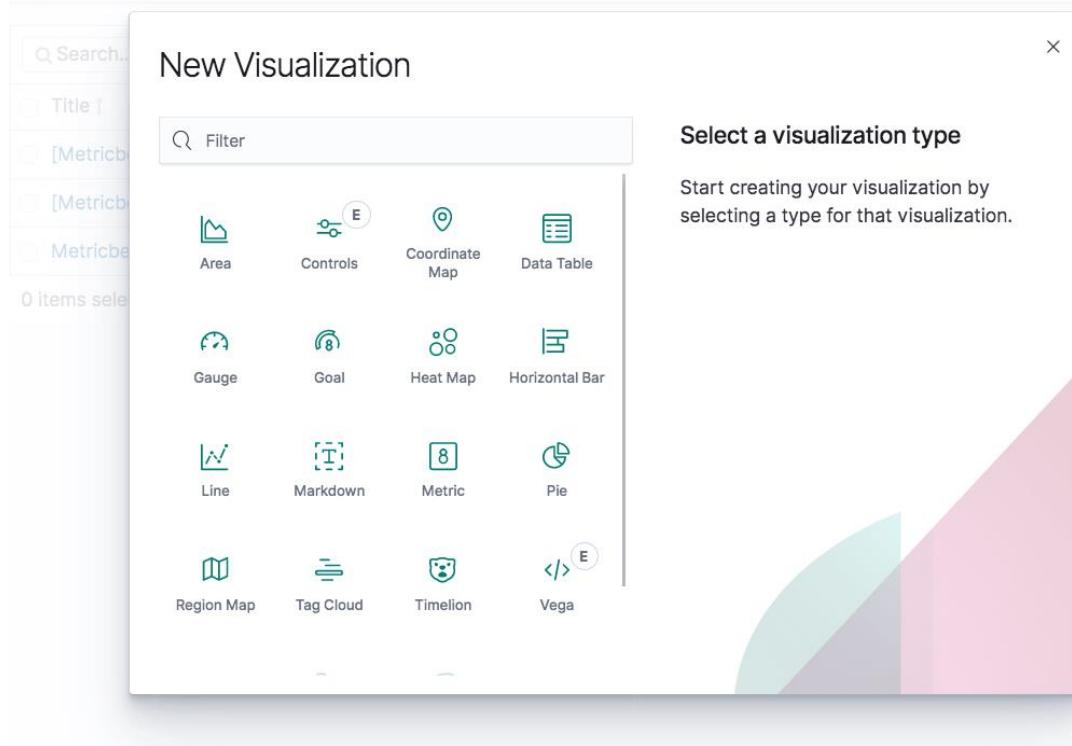
Kibana visualizations

As mentioned above, Kibana is renowned for visualization capabilities. Using a wide variety of different charts and graphs, you can slice and dice your data any way you want. You can create your own [custom visualizations](#) with the help of vega and vega-lite. You will find that you can do almost whatever you want with your data.

Creating visualizations, however, is now always straightforward and can take time. Key to making this process painless is knowing your data. The more you are acquainted with the different nooks and crannies in your data, the easier it is.

Kibana visualizations are built on top of Elasticsearch queries. Using Elasticsearch aggregations (e.g. sum, average, min, max, etc.), you can

perform various processing actions to make your visualizations depict trends in the data.



Visualization types

Visualizations in Kibana are categorized into five different types of visualizations:

- Basic Charts (Area, Heat Map, Horizontal Bar, Line, Pie, Vertical bar)
- Data (Date Table, Gauge, Goal, Metric)
- Maps (Coordinate Map, Region Map)
- Time series (Timelion, Visual Builder)
- Other (Controls, Markdown, Tag Cloud)

In the table below, we describe the main function of each visualization and a usage example:

Vertical Bar Chart: Great For Time Series Data And For Splitting Lines Across Fields

URLs over time

Pie Chart: Useful For Displaying Parts Of A Whole

Top 5 memory consuming system

Area Chart: For Visualizing Time Series Data And For Splitting Lines On Fields

Users over time

Heat Map: For Showing Statistical Outliers And Are Often Used For Latency Values

Latency and outliers

Horizontal Bar Chart: Good For Showing Relationships Between Two Fields

URL and referrer

Line Chart: Are A Simple Way To Show Time Series And Are Good For Splitting Lines To Show Anomalies

Average CPU over time by host

Data Table: Best Way To Split Across Multiple Fields In A Custom Way

Top user, host, pod, container by us

Gauge: A Way To Show The Status Of A Specific Metric Using Thresholds You Define

Memory consumption limits

Metric: Useful Visualization For Displaying A Calculation As A Single Number

No. of Docker containers run.

Coordinate Map & Region Map: Help Add A Geographical Dimension To IP-Based Logs

Geographic origin of web server requests.

Timelion And Visual Query Builder: Allows You To Create More Advanced Queries Based On Time Series Data

Percentage of 500 errors over time

Markdown: A Great Way To Add A Customized Text Or Image-Based

Company logo or a description of a dashboard

Visualization To Your Dashboard Based On Markdown Syntax

Tag Cloud: Helps Display Groups Of Words Sized By Their Importance

Countries sending requests to a web server

Kibana dashboards

Once you have a collection of visualizations ready, you can add them all into one comprehensive visualization called a dashboard. Dashboards give you the ability to monitor a system or environment from a high vantage point for easier event correlation and trend analysis.

Dashboards are highly dynamic — they can be edited, shared, played around with, opened in different display modes, and more. Clicking on one field in a specific visualization within a dashboard, filters the entire dashboard accordingly (you will notice a filter added at the top of the page).

For more information and tips on creating a Kibana dashboard, see [Creating the Perfect Kibana Dashboard](#).

Kibana pages

Recent versions of Kibana include dedicated pages for various monitoring features such as APM and infrastructure monitoring. Some of these features were formerly part of the X-Pack, others, such as Canvas and Maps, are brand new:

- Canvas – the “photoshop” of machine-generated data, Canvas is an advanced visualization tool that allows you to design and visualize your logs and metrics in creative new ways.
- Maps – meant for geospatial analysis, this page supports multiple layers and data sources, the mapping of individual geo points and shapes, global searching for ad-hoc analysis, customization of elements, and more.
- Infrastructure – helps you gain visibility into the different components constructing your infrastructure, such as hosts and containers.
- Logs – meant for live tracking of incoming logs being shipped into the stack with Logstash.
- APM – designed to help you monitor the performance of your applications and identify bottlenecks.
- Uptime – allows you to monitor and gauge the status of your applications using a dedicated UI, based on data shipped into the stack with Heartbeat.
- Stack Monitoring – provides you with built-in dashboards for monitoring Elasticsearch, Kibana, Logstash and Beats. Requires manual configuration.

Note: These pages are not licensed under Apache 2.0 but under Elastic's Basic license.

Kibana Elasticsearch index

The searches, visualizations, and dashboards saved in Kibana are called objects. These objects are stored in a dedicated Elasticsearch index (.kibana) for debugging, sharing, repeated usage and backup.

The index is created as soon as Kibana starts. You can change its name in the Kibana configuration file. The index contains the following documents, each containing their own set of fields:

- Saved index patterns
- Saved searches
- Saved visualizations
- Saved dashboards

What's next?

This article covered the functions you will most likely be using Kibana for, but there are plenty more tools to learn about and play around with. There are development tools such as Console, and if you're using X-Pack, additional monitoring and alerting features.

It's important to note that for production, you will most likely need to add some elements to Kibana to make it more secure and robust. For example, placing a proxy such as Nginx in front of Kibana or plugging in an alerting layer. This requires additional configuration or costs.

If you're just getting started with Kibana, read this [Kibana Tutorial](#).

Beats

The ELK Stack, which traditionally consisted of three main components — Elasticsearch, Logstash, and Kibana, is now also used together with what is called “Beats” — a family of log shippers for different use cases containing Filebeat, Metricbeat, Packetbeat, Auditbeat, Heartbeat and Winlogbeat.

What are Beats?

Beats are a collection of open source log shippers that act as agents installed on the different servers in your environment for collecting logs or metrics. Written in Go, these shippers were designed to be lightweight in nature — they leave a small installation footprint, are resource efficient, and function with no dependencies.

The data collected by the different beats varies — log files in the case of Filebeat, network data in the case of Packetbeat, system and service metrics in the case of Metricbeat, Windows event logs in the case of Winlogbeat, and so forth. In addition to the beats developed and supported by Elastic, there is also a growing list of beats developed and contributed by the community.

Once collected, you can configure your beat to ship the data either directly into Elasticsearch or to Logstash for additional processing. Some of the beats also support processing which helps offload some of the heavy lifting Logstash is responsible for.

Since version 7.0, Beats comply with the Elastic Common Schema (ECS) introduced at the beginning of 2019. ECS aims at making it easier for users to correlate between data sources by sticking to a uniform field format.

Read about how to install, use and run beats in our [Beats Tutorial](#).

Filebeat

Filebeat is used for collecting and shipping log files. Filebeat can be installed on almost any operating system, including as a Docker container, and also comes with internal modules for specific platforms such as Apache, MySQL, Docker and more, containing default configurations and Kibana objects for these platforms.

Packetbeat

A network packet analyzer, Packetbeat was the first beat introduced. Packetbeat captures network traffic between servers, and as such can be used for application and performance monitoring. Packetbeat can be installed on the server being monitored or on its own dedicated server.

Read more about how to use Packetbeat [here](#).

Metricbeat

Metricbeat collects ships various system-level metrics for various systems and platforms. Like Filebeat, Metricbeat also supports internal modules for collecting statistics from specific platforms. You can configure the frequency by which Metricbeat collects the metrics and what specific metrics to collect using these modules and sub-settings called metricsets.

Winlogbeat

Winlogbeat will only interest Windows sysadmins or engineers as it is a beat designed specifically for collecting Windows Event logs. It can be used to analyze security events, updates installed, and so forth.

Read more about how to use Winlogbeat [here](#).

Auditbeat

Auditbeat can be used for auditing user and process activity on your Linux servers. Similar to other traditional system auditing tools (systemd, auditd), Auditbeat can be used to identify security breaches — file changes, configuration changes, malicious behavior, etc.

Read more about how to use Auditbeat [here](#).

Functionbeat

Functionbeat is defined as a “serverless” shipper that can be deployed as a function to collect and ship data into the ELK Stack. Designed for monitoring cloud environments, Functionbeat is currently tailored for Amazon setups and can be deployed as an Amazon Lambda function to collect data from Amazon CloudWatch, Kinesis and SQS.

Configuring beats

Being based on the same underlying architecture, Beats follow the same structure and configuration rules.

Generally speaking, the configuration file for your beat will include two main sections: one defines what data to collect and how to handle it, the other where to send the data to.

Configuration files are usually located in the same directory — for Linux, this location is the `/etc/<beatname>` directory. For Filebeat, this would be `/etc/filebeat/filebeat.yml`, for Metricbeat, `/etc/metricbeat/metricbeat.yml`. And so forth.

Beats configuration files are based on the YAML format with a dictionary containing a group of key-value pairs, but they can contain lists and strings, and various other data types. Most of the beats also include files with complete configuration examples, useful for learning the different configuration settings that can be used. Use it as a reference.

Beats modules

Filebeat and Metricbeat support modules — built-in configurations and Kibana objects for specific platforms and systems. Instead of configuring these two beats, these modules will help you start out with pre-configured settings which work just fine in most cases but that you can also adjust and fine tune as you see fit.

Filebeat modules: Apache, Auditd, Cisco, Coredns, Elasticsearch, Envoyproxy, HAProxy, Icinga, IIS, Iptables, Kafka, Kibana, Logstash, MongoDB, MySQL, Nats, NetFlow, Nginx, Osquery, Palo Alto Networks, PostgreSQL, RabbitMQ, Redis, Santa, Suricata, System, Traefik, Zeek (Bro).

Metricbeat modules: Aerospike, Apache, AWS, Ceph, Couchbase, Docker, Dropwizard, Elasticsearch, Envoyproxy, Etcd, Golang, Graphite, HAProxy,

HTTP, Jolokia, Kafka, Kibana, Kubernetes, kvm, Logstash, Memcached, MongoDB, mssql, Munin, MySQL, Nats, Nginx, PHP_FPM, PostgreSQL, Prometheus, RabbitMQ, Redis, System, traefik, uwsgi, vSphere, Windows, Zookeeper.

Configuration example

So, what does a configuration example look like? Obviously, this differs according to the beat in question. Below, however, is an example of a Filebeat configuration that is using a single prospector for tracking Puppet server logs, a JSON directive for parsing, and a local Elasticsearch instance as the output destination.

```
filebeat.prospectors: - type: log enabled: true paths: - /var/log/puppetlabs/puppetserver/puppetserver.log.json - /var/log/puppetlabs/puppetserver/puppetserver-access.log.json json.keys_under_root: true output.elasticsearch: # Array of hosts to connect to. hosts: ["localhost:9200"]
```

Copy

Configuration best practices

Each beat contains its own unique configuration file and configuration settings, and therefore requires its own set of instructions. Still, there are some common configuration best practices that can be outlined here to provide a solid general understanding.

- Some beats, such as Filebeat, include full example configuration files (e.g., /etc/filebeat/filebeat.full.yml). These files include long lists all the available configuration options.
- YAML files are extremely sensitive. DO NOT use tabs when indenting your lines — only spaces. YAML configuration files for Beats are mostly built the same way, using two spaces for indentation.

- Use a text editor (I use Sublime) to edit the file.
- The '-' (dash) character is used for defining new elements — be sure to preserve their indentations and the hierarchies between sub-constructs.

Additional information and tips are available in the [Musings in YAML](#) article.

What next?

Beats are a great and welcome addition to the ELK Stack, taking some of the load off Logstash and making data pipelines much more reliable as a result. Logstash is still a critical component for most pipelines that involve aggregating log files since it is much more capable of advanced processing and data enrichment.

Beats also have some glitches that you need to take into consideration. YAML configurations are always sensitive, and Filebeat, in particular, should be handled with care so as not to create resource-related issues. I cover some of the issues to be aware of in the [5 Filebeat Pitfalls](#) article.

Read more about how to install, use and run beats in our [Beats Tutorial](#).

Did we miss something? Did you find a mistake? We're relying on your feedback to keep this guide up-to-date. Please add your comments at the bottom of the page, or send them to: elk-guide@logz.io

ELK in Production

Log management has become a must-do action for any organization to resolve problems and ensure that applications are running in a healthy manner. As such, log management has become in essence, a mission-critical system.

When you're troubleshooting a production issue or trying to identify a security hazard, the system must be up and running around the clock.

Otherwise, you won't be able to troubleshoot or resolve issues that arise — potentially resulting in performance degradation, downtime or security breach. A log analytics system that runs continuously can equip your organization with the means to track and locate the specific issues that are wreaking havoc on your system.

In this section, we will share some of our experiences from building Logz.io. We will detail some of the challenges involved in building an ELK Stack at scale as well as offer some related guidelines.

Generally speaking, there are some basic requirements a production-grade ELK implementation needs to answer:

1. Save and index all of the log files that it receives (sounds obvious, right?)
2. Operate when the production system is overloaded or even failing (because that's when most issues occur)
3. Keep the log data protected from unauthorized access
4. Have maintainable approaches to data retention policies, upgrades, and more

How can this be achieved?

Don't Lose Log Data

If you're troubleshooting an issue and go over a set of events, it only takes one missing logline to get incorrect results. Every log event must be captured. For example, you're viewing a set of events in MySQL that ends with a database exception. If you lose one of these events, it might be impossible to pinpoint the cause of the problem.

The recommended method to ensure a resilient data pipeline is to place a buffer in front of Logstash to act as the entry point for all log events that are

shipped to your system. It will then buffer the data until the downstream components have enough resources to index.

The most common buffer used in this context is Kafka, though also Redis and RabbitMQ are used.

Elasticsearch is the engine at the heart of ELK. It is very susceptible to load, which means you need to be extremely careful when indexing and increasing your amount of documents. When Elasticsearch is busy, Logstash works slower than normal — which is where your buffer comes into the picture, accumulating more documents that can then be pushed to Elasticsearch. This is critical not to lose log events.

Monitor Logstash/Elasticsearch Exceptions

Logstash may fail when trying to index logs in Elasticsearch that cannot fit into the automatically-generated mapping.

For example, let's say you have a log entry that looks like this:

```
timestamp=time, type=my_app, error=3,....
```

Copy

But later, your system generates a similar log that looks as follows:

```
timestamp=time, type=my_app, error="Error",....
```

Copy

In the first case, a number is used for the *error* field. In the second case, a string is used. As a result, Elasticsearch will NOT index the document — it will just return a failure message and the log will be dropped.

To make sure that such logs are still indexed, you need to:

1. 32. Work with developers to make sure they're keeping log formats consistent. If a log schema change is required, just change the index according to the type of log.

2. Ensure that Logstash is consistently fed with information and monitor Elasticsearch exceptions to ensure that logs are not shipped in the wrong formats. Using mapping that is fixed and less dynamic is probably the only solid solution here (that doesn't require you to start coding).

At Logz.io, we solve this problem by building a pipeline to handle mapping exceptions that eventually index these documents in manners that don't collide with existing mapping.

Keep up with growth and bursts

As your company succeeds and grows, so does your data. Machines pile up, environments diversify, and log files follow suit. As you scale out with more products, applications, features, developers, and operations, you also accumulate more logs. This requires a certain amount of compute resource and storage capacity so that your system can process all of them.

In general, log management solutions consume large amounts of CPU, memory, and storage. Log systems are bursty by nature, and sporadic bursts are typical. If a file is purged from your database, the frequency of logs that you receive may range from 100 to 200 to 100,000 logs per second.

As a result, you need to allocate up to 10 times more capacity than normal. When there is a real production issue, many systems generally report failures or disconnections, which cause them to generate many more logs. This is actually when log management systems are needed more than ever.

ELK Elasticity

One of the biggest challenges of building an ELK deployment is making it scalable.

Let's say you have an e-commerce site and experience an increasing number of incoming log files during a particular time of year. To ensure that this influx of log data does not become a bottleneck, you need to make sure that your environment can scale with ease. This requires that you scale on all fronts — from Redis (or Kafka), to Logstash and Elasticsearch — which is challenging in multiple ways.

Regardless of where you're deploying your ELK stack — be it on AWS, GCP, or in your own datacenter — we recommend having a cluster of Elasticsearch nodes that run in different availability zones, or in different segments of a data center, to ensure high availability.

Let's take a look at some of the components required for a scalable ELK deployment.

Kafka

As mentioned above, placing a buffer in front of your indexing mechanism is critical to handle unexpected events. It could be mapping conflicts, upgrade issues, hardware issues or sudden increases in the volume of logs. Whatever the cause you need an overflow mechanism, and this is where Kafka comes into the picture.

Acting as a buffer for logs that are to be indexed, Kafka must persist your logs in at least 2 replicas, and it must retain your data (even if it was consumed already by Logstash) for at least 1-2 days.

This goes against planning for the local storage available to Kafka, as well as the network bandwidth provided to the Kafka brokers. Remember to take into account huge spikes in incoming log traffic (tens of times more than "normal"), as these are the cases where you will need your logs the most.

Consider how much manpower you will have to dedicate to fixing issues in your infrastructure when planning the retention capacity in Kafka.

Another important consideration is the ZooKeeper management cluster – it has its own requirements. Do not overlook the disk performance requirements for ZooKeeper, as well as the availability of that cluster. Use a three or five node cluster, spread across racks/availability zones (but not regions).

One of the most important things about Kafka is the monitoring implemented on it. You should always be looking at your log consumption (aka “Lag”) in terms of the time it takes from when a log message is published to Kafka until after it has been indexed in Elasticsearch and is available for search.

Kafka also exposes a plethora of operational metrics, some of which are extremely critical to monitor: network bandwidth, thread idle percent, under-replicated partitions and more. When considering consumption from Kafka and indexing you should consider what level of parallelism you need to implement (after all, Logstash is not very fast). This is important to understand the consumption paradigm and plan the number of partitions you are using in your Kafka topics accordingly.

Logstash

Knowing how many Logstash instances to run is an art unto itself and the answer depends on a great many of factors: volume of data, number of pipelines, size of your Elasticsearch cluster, buffer size, accepted latency — to name just a few.

Deploy a scalable queuing mechanism with different scalable workers. When a queue is too busy, scale additional workers to read into Elasticsearch. Once you’ve determined the number of Logstash instances required, run each one of them in a different AZ (on AWS). This comes at a cost due to data transfer but will guarantee a more resilient data pipeline.

You should also separate Logstash and Elasticsearch by using different machines for them. This is critical because they both run as JVMs and

consume large amounts of memory, which makes them unable to run on the same machine effectively.

Hardware specs vary, but it is recommended allocating a maximum of 30 GB or half of the memory on each machine for Logstash. In some scenarios, however, making room for caches and buffers is also a good best practice.

Elasticsearch cluster

Elasticsearch is composed of a number of different node types, two of which are the most important: the master nodes and the data nodes. The master nodes are responsible for cluster management while the data nodes, as the name suggests, are in charge of the data (read more about setting up an Elasticsearch cluster [here](#)).

We recommend building an Elasticsearch cluster consisting of at least three master nodes because of the common occurrence of split brain, which is essentially a dispute between two nodes regarding which one is actually the master.

As far as the data nodes go, we recommend having at least two data nodes so that your data is replicated at least once. This results in a minimum of five nodes: the three master nodes can be small machines, and the two data nodes need to be scaled on solid machines with very fast storage and a large capacity for memory.

Run in Different AZs (But Not in Different Regions)

We recommend having your Elasticsearch nodes run in different availability zones or in different segments of a data center to ensure high availability. This can be done through an [Elasticsearch setting](#) that allows you to configure every document to be replicated between different AZs. As with Logstash, the resulting costs resulting from this kind of deployment can be quite steep due to data transfer.

Security

Due to the fact that logs may contain sensitive data, it is crucial to protect who can see what. How can you limit access to specific dashboards, visualizations, or data inside your log analytics platform? There is no simple way to do this in the ELK Stack.

One option is to use nginx reverse proxy to access your Kibana dashboard, which entails a simple nginx configuration that requires those who want to access the dashboard to have a username and password. This quickly blocks access to your Kibana console and allows you to configure authentication as well as add SSL/TLS encryption Elastic

Elastic recently announced making some security features free, incl. encryption, role-based access, and authentication. More advanced security configurations and integrations, however, e.g. LDAP/AD support, SSO, encryption at rest, are not available out of the box. Keep in mind that while these features are indeed free of charge, they are not completely open source.

Another option is SearchGuard which provides a free security plugin for Elasticsearch including role-based access control and SSL/TLS encrypted node-to-node communication. It's also worth mentioning Amazon's OpenDistro for Elasticsearch that comes built in with an open source security plugin with similar capabilities.

Last but not least, be careful when exposing Elasticsearch because it is very susceptible to attacks. There are some basic steps to take that will help you secure your Elasticsearch instances.

Maintainability

Log Data Consistency

Logstash processes and parses logs in accordance with a set of rules defined by filter plugins. Therefore, if you have an access log from nginx, you want the ability to view each field and have visualizations and dashboards built based on specific fields. You need to apply the relevant parsing abilities to Logstash — which has proven to be quite a challenge, particularly when it comes to building groks, debugging them, and actually parsing logs to have the relevant fields for Elasticsearch and Kibana.

At the end of the day, it is very easy to make mistakes using Logstash, which is why you should carefully test and maintain all of your log configurations by means of version control. That way, while you may get started using nginx and MySQL, you may incorporate custom applications as you grow that result in large and hard-to-manage log files. The community has generated a lot of solutions around this topic, but trial and error are extremely important with open source tools before using them in production.

Data Retention

Another aspect of maintainability comes into play with excess indices. Depending on how long you want to retain data, you need to have a process set up that will automatically delete old indices — otherwise, you will be left with too much data and your Elasticsearch will crash, resulting in data loss.

To prevent this from happening, you can use Elasticsearch Curator to delete indices. We recommend having a cron job that automatically spawns Curator with the relevant parameters to delete any old indices, ensuring you don't end up holding too much data. It is commonly required to save logs to S3 in a bucket for compliance, so you want to be sure to have a copy of the logs in their original format.

Upgrades

Major versions of the stack are released quite frequently, with great new features but also breaking changes. It is always wise to read and do research on what these changes mean for your environment before you begin upgrading. Latest is not always the greatest!

Performing Elasticsearch upgrades can be quite an endeavor but has also become safer due to some recent changes. First and foremost, you need to make sure that you will not lose any data as a result of the process. Run tests in a non-production environment first. Depending on what version you are upgrading from and to, be sure you understand the process and what it entails.

Logstash upgrades are generally easier, but pay close attention to the compatibility between Logstash and Elasticsearch and breaking changes.

Kibana upgrades can be problematic, especially if you're running on an older version. Importing objects is “generally” supported, but you should backup your objects and test the upgrade process before upgrading in production. As always — study breaking changes!

Summary

Getting started with ELK to process logs from a server or two is easy and fun. Like any other production system, it takes much more work to reach a solid production deployment. We know this because we've been working with many users who struggle with making ELK operational in production. Read more about [the real cost of doing ELK on your own](#).

Did we miss something? Did you find a mistake? We're relying on your feedback to keep this guide up-to-date. Please add your comments at the bottom of the page, or send them to: elk-guide@logz.io

Common Pitfalls

Like any piece of software, the ELK Stack is not without its pitfalls. While relatively easy to set up, the different components in the stack can become difficult to handle as soon as you move on to complex setups and a larger scale of operations necessary for handling multiple data pipelines.

There's nothing like trial and error. At the end of the day, the more you do, the more you err and learn along the way. At Logz.io, we have accumulated a decent amount of Elasticsearch, Logstash and Kibana time, and are happy to share our hard-earned lessons with our readers.

There are several common, and yet sometimes critical, mistakes that users tend to make while using the different components in the stack. Some are extremely simple and involve basic configurations, others are related to best practices. In this section of the guide, we will outline some of these mistakes and how you can avoid making them.

Elasticsearch

Not defining Elasticsearch mapping

Say that you start Elasticsearch, create an index, and feed it with JSON documents without incorporating schemas. Elasticsearch will then iterate over each indexed field of the JSON document, estimate its field, and create a respective mapping. While this may seem ideal, Elasticsearch mappings are not always accurate. If, for example, the wrong field type is chosen, then indexing errors will pop up.

To fix this issue, you should define mappings, especially in production-line environments. It's a best practice to index a few documents, let Elasticsearch guess the field, and then grab the mapping it creates with GET /index_name/doc_type/_mapping. You can then take matters into your own

hands and make any appropriate changes that you see fit without leaving anything up to chance.

For example, if you index your first document like this:

```
{ "action": "Some action", "payload": "2016-01-20" }
```

Copy

Elasticsearch will automatically map the “payload” field as a date field

Now, suppose that your next document looks like this:

```
{ "action": "Some action 1", "payload": "USER_LOCKED" }
```

Copy

In this case, “payload” of course is not a date, and an error message may pop up and the new index will not be saved because Elasticsearch has already marked it as “date.”

Capacity Provisioning

Provisioning can help to equip and optimize Elasticsearch for operational performance. It requires that Elasticsearch is designed in such a way that will keep nodes up, stop memory from growing out of control, and prevent unexpected actions from shutting down nodes.

“How much space do I need?” is a question that users often ask themselves. Unfortunately, there is no set formula, but certain steps can be taken to assist with the planning of resources.

First, simulate your actual use-case. Boot up your nodes, fill them with real documents, and push them until the shard breaks.

Still, be sure to keep in mind that the concept of “start big and scale down” can save you time and money when compared to the alternative of adding and configuring new nodes when your current amount is no longer enough.

Once you define a shard's capacity, you can easily apply it throughout your entire index. It is very important to understand resource utilization during the testing process because it allows you to reserve the proper amount of RAM for nodes, configure your JVM heap space, and optimize your overall testing process.

Oversized Template

Large templates are directly related to large mappings. In other words, if you create a large mapping for Elasticsearch, you will have issues with syncing it across your nodes, even if you apply them as an index template.

The issues with big index templates are mainly practical — you might need to do a lot of manual work with the developer as the single point of failure — but they can also relate to Elasticsearch itself. Remember: You will always need to update your template when you make changes to your data model.

Production Fine-tuning

By default, the first cluster that Elasticsearch starts is called `elasticsearch`. If you are unsure about how to change a configuration, it's best to stick to the default configuration. However, it is a good practice to rename your production cluster to prevent unwanted nodes from joining your cluster.

Below is an example of how you might want to rename your cluster and nodes:

```
cluster.name: elasticsearch_production node.name:  
elasticsearch_node_001
```

Copy

Logstash

Logstash configuration file

This is one of the main pain points not only for working with Logstash but for the entire stack. Having your entire ELK-based pipelines stalled because of a bad Logstash configuration error is not an uncommon occurrence.

Hundreds of different plugins with their own options and syntax instructions, differently located configuration files, files that tend to become complex and difficult to understand over time — these are just some of the reasons why Logstash configuration files are the cemetery of many a pipeline.

As a rule of the thumb, try and keep your Logstash configuration file as simple as possible. This also affects performance. Use only the plugins you are sure you need. This is especially true of the various filter plugins which tend to add up necessarily.

If possible — test and verify your configurations before starting Logstash in production. If you're running Logstash from the command line, use the `-config.test_and_exit` parameter. Use the grok debugger to test your grok filter.

Memory consumption

Logstash runs on JVM and consumes a hefty amount of resources to do so. Many discussions have been floating around regarding Logstash's significant memory consumption. Obviously, this can be a great challenge when you want to send logs from a small machine (such as AWS micro instances) without harming application performance.

Recent versions of Logstash and the ELK Stack have improved this inherent weakness. The new execution engine was introduced in version 7.x promises to speed up performance and the resource footprint Logstash has.

Also, Filebeat and/or Elasticsearch Ingest Node, can help with outsourcing some of the processing heavy lifting to the other components in the stack. You can also make use of monitoring APIs to identify bottlenecks and problematic processing.

Slow processing

Limited system resources, a complex or faulty configuration file, or logs not suiting the configuration can result in extremely slow processing by Logstash that might result in data loss.

You need to closely monitor key system metrics to make sure you're keeping tabs on Logstash processing — monitor the host's CPU, I/O, memory and JVM heap. Be ready to fine-tune your system configurations accordingly (e.g. raising the JVM heap size or raising the number of pipeline workers). There is a nice [performance checklist here](#).

Key-Value Filter Plugin

Key-values is a filter plug-in that extracts keys and values from a single log using them to create new fields in the structured data format. For example, let's say a logline contains "x=5". If you pass that through a key-value filter, it will create a new field in the output JSON format where the key would be "x" and the value would be "5".

By default, the key-value filter will extract every key=value pattern in the source field. However, the downside is that you don't have control over the keys and values that are created when you let it work automatically, out-of-the-box with the default configuration. It may create many keys and values with an undesired structure, and even malformed keys that make the output unpredictable. If this happens, Elasticsearch may fail to index the resulting document and parse irrelevant information.

Kibana

Elasticsearch connectivity

Kibana is a UI for analyzing the data indexed in Elasticsearch– A super-useful UI at that, but still, only a UI. As such, how Kibana and Elasticsearch talk to each other directly influences your analysis and visualization workflow. It's easy to miss some basic steps needed to make sure the two behave nicely together.

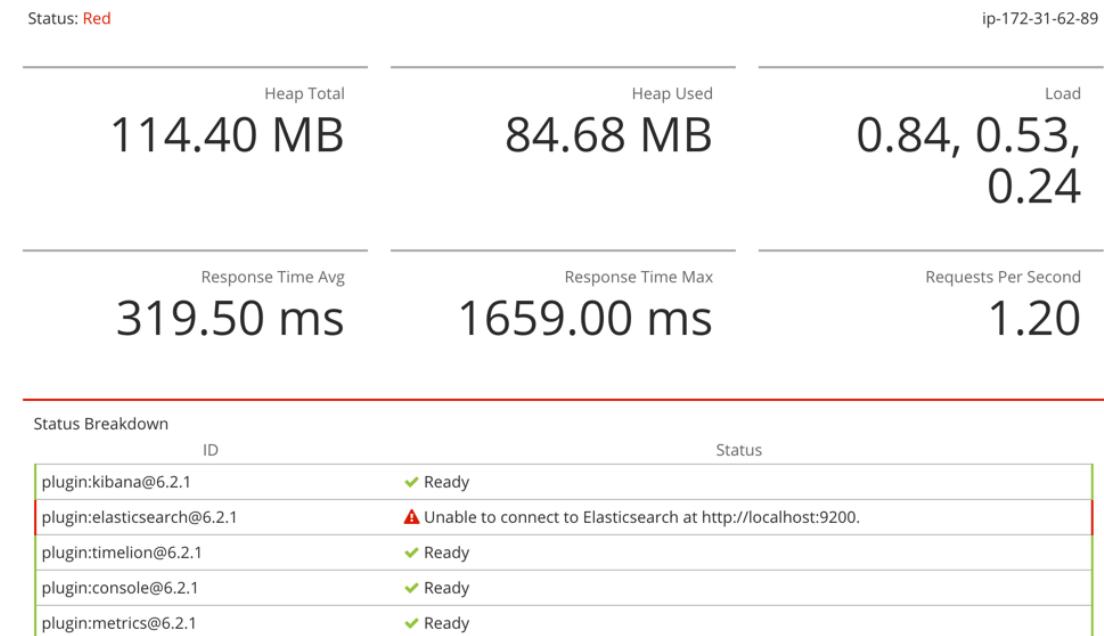
Defining an index pattern

There's little use for of an analysis tool if there is no data for it to analyze. If you have no data indexed in Elasticsearch or have not defined the correct index pattern for Kibana to read from, your analysis work cannot start.

So, verify that a) your data pipeline is working as expected and indexing data in Elasticsearch (you can do this by querying Elasticsearch indices), and b) you have defined the correct index pattern in Kibana (Management → Index Patterns in Kibana).

Can not connect to Elasticsearch

A common glitch when setting up Kibana is to misconfigure the connection with Elasticsearch, resulting in the following message when you open Kibana:



As the message reads, Kibana simply cannot connect to an Elasticsearch instance. There are some simple reasons for this — Elasticsearch may not be running, or Kibana might be configured to look for an Elasticsearch instance on a wrong host and port.

The latter is the more common reason for seeing the above message, so open the Kibana configuration file and be sure to define the IP and port of the Elasticsearch instance you want Kibana to connect to.

Bad Kibana searches

Querying Elasticsearch from Kibana is an art because many different types of searches are available. From free-text searches to field-level and regex searches, there are many options, and this variety is one of the reasons that people opt for the ELK Stack in the first place. As implied in the opening statement above, some Kibana searches are going to crash Elasticsearch in certain circumstances.

For example, using a leading wildcard search on a large dataset has the potential of stalling the system and should, therefore, be avoided.

Try and avoid using wildcard queries if possible, especially when performed against very large data sets.

Advanced settings

Some Kibana-specific configurations can cause your browser to crash. For example, depending on your browser and system settings, changing the value of the discover:sampleSize setting to a high number can easily cause Kibana to freeze.

That is why the good folks at Elastic have placed a warning at the top of the page that is supposed to convince us to be extra careful. Anyone with a guess on how successful this warning is?

Management / Kibana

Index Patterns Saved Objects Advanced Settings

Caution: You can break stuff here

Be careful in here, these settings are for very advanced users only. Tweaks you make here can break large portions of Kibana. Some of these settings may be undocumented, unsupported or experimental. If a field has a default value, blanking the field will reset it to its default which may be unacceptable given other configuration directives. Deleting a custom setting will permanently remove it from Kibana's config.

| Name | Value | Edit |
|---|--|------|
| query:queryString:options Options for the lucene query string parser | { "analyze_wildcard": true, "default_field": "*" } | Edit |
| search:queryLanguage Query language used by the query bar. Kuery is an experimental new language built specifically for Kibana. | lucene | Edit |
| search:queryLanguage:switcher:enable Show or hide the query language switcher in the query bar | false | Edit |
| sort:options Options for the Elasticsearch sort parameter | { "unmapped_type": "boolean" } | Edit |

Beats

The log shippers belonging to the Beats family are pretty resilient and fault-tolerant. They were designed to be lightweight in nature and with a low resource footprint.

YAML configuration files

The various beats are configured with YAML configuration files. YAML being YAML, these configurations are extremely syntax sensitive. You can find a list of tips for writing these files [in this article](#), but generally speaking, it's best to handle these files carefully — validate your files using an online YAML validator, makes use of the example files provided in the different packages, and use spaces instead of tabs.

Filebeat – CPU Usage

Filebeat is an extremely lightweight shipper with a small footprint, and while it is extremely rare to find complaints about Filebeat, there are some cases where you might run into high CPU usage.

One factor that affects the amount of computation power used is the scanning frequency — the frequency at which Filebeat is configured to scan for files. This frequency can be defined for each prospector using the `scan_frequency` setting in your Filebeat configuration file, so if you have a large number of prospectors running with a tight scan frequency, this may result in excessive CPU usage.

Filebeat – Registry File

Filebeat is designed to remember the previous reading for each log file being harvested by saving its state. This helps Filebeat ensure that logs are not lost if, for example, Elasticsearch or Logstash suddenly go offline (that never happens, right?).

This position is saved to your local disk in a dedicated registry file, and under certain circumstances, when creating a large number of new log files, for example, this registry file can become quite large and begin to consume too much memory.

It's important to note that there are some good options for making sure you don't fall into this caveat — you can use the *clean_removed* option, for example, to tell Filebeat to clean non-existing files from the registry file.

Filebeat – Removed or Renamed Log Files

File handlers for removed or renamed log files might exhaust disk space. As long as a harvester is open, the file handler is kept running. Meaning that if a file is removed or renamed, Filebeat continues to read the file, the handler consuming resources. If you have multiple harvesters working, this comes at a cost.

Again, there are workarounds for this. You can use the *close_inactive* configuration setting to tell Filebeat to close a file handler after identifying inactivity for a defined duration and the *closed_removed* setting can be enabled to tell Filebeat to shut down a harvester when a file is removed (as soon as the harvester is shut down, the file handler is closed and this resource consumption ends.)

Summing it up

The ELK Stack is a fantastic piece of software with some known and some less-known weak spots.

The good news is that all of the issues listed above can be easily mitigated and avoided as described. The bad news is that there are additional pitfalls that have not been detailed here.

Here are some articles with more tips and best practices to help avoid them:

○ [Top 5 Elasticsearch Mistakes](#)

○ [5 Logstash Pitfalls You Need to Avoid](#)

○ [5 Filebeat Pitfalls To Be Aware Of](#)

○ 5 Easy Ways to Crash Elasticsearch

Be diligent. Do your research.

Did we miss something? Did you find a mistake? We're relying on your feedback to keep this guide up-to-date. Please add your comments at the bottom of the page, or send them to: elk-guide@logz.io

Use Cases

The ELK Stack is most commonly used as a log analytics tool. Its popularity lies in the fact that it provides a reliable and relatively scalable way to aggregate data from multiple sources, store it and analyze it. As such, the stack is used for a variety of different use cases and purposes, ranging from development to monitoring, to security and compliance, to SEO and BI.

Before you decide to set up the stack, understand your specific use case first. This directly affects almost all the steps implemented along the way — where and how to install the stack, how to configure your Elasticsearch cluster and which resources to allocate to it, how to build data pipelines, how to secure the installation — the list is endless.

So, what are you going to be using ELK for?

Development and troubleshooting

Logs are notorious for being in handy during a crisis. The first place one looks at when an issue takes place are your error logs and exceptions. Yet, logs come in handy much earlier in an application's lifecycle.

We are strong believers in log-driven development, where logging starts from the very first function written and then subsequently instrumented throughout the entire application. Implementing logging into your code adds a measure of observability into your applications that come in handy when troubleshooting issues.

Whether you are developing a monolith or microservices, the ELK Stack comes into the picture early on as a means for developers to correlate, identify and troubleshoot errors and exceptions taking place, preferably in testing or staging, and before the code goes into production. Using a variety of different appenders, frameworks, libraries and shippers, log messages are pushed into the ELK Stack for centralized management and analysis.

Once in production, Kibana dashboards are used for monitoring the general health of applications and specific services. Should an issue take place, and if logging was instrumented in a structured way, having all the log data in one centralized location helps make analysis and troubleshooting a more efficient and speedy process.

Cloud operations

Modern IT environments are multilayered and distributed in nature, posing a huge challenge for the teams in charge of operating and monitoring them. Monitoring across all the different systems and components comprising an application's architecture is extremely time and resource consuming.

To be able to accurately gauge and monitor the status and general health of an environment, DevOps and IT Operations teams need to take into account the following key considerations: how to access each machine, how to collect the data, how to add context to the data and process it, where to store the data and how long to store it for, how to analyze the data, how to secure the data and how to back it up.

The ELK Stack helps by providing organizations with the means to tackle these questions by providing an almost all-in-one solution. Beats can be deployed on machines to act as agents forwarding log data to Logstash instances. Logstash can be configured to aggregate the data and process it before indexing the data in Elasticsearch. Kibana is then used to analyze the

data, detect anomalies, perform root cause analysis, and build beautiful monitoring dashboards.

And it's not just logs. While Elasticsearch was initially designed for full-text search and analysis, it is increasingly being used for metrics analysis as well. Monitoring performance metrics for each component in your architecture is key for gaining visibility into operations. Collecting these metrics can be done using 3rd party auditing or monitoring agents or even using some of the available beats (e.g. Metricbeat, Packetbeat) and Kibana now ships with new visualization types to help analyze time series (Timelion, Visual Builder).

Application performance monitoring (APM)

Application Performance Monitoring, aka APM, is one of the most common methods used by engineers today to measure the availability, response times and behavior of applications and services.

Elastic APM is an application performance monitoring system which is built on top of the ELK Stack. Similar to other APM solutions in the market, Elastic APM allows you to track key performance-related information such as requests, responses, database transactions, errors, etc.

Likewise, open source distributed tracing tools such as Zipkin and Jaeger can be integrated with ELK for diving deep into application performance.

Security and compliance

Security has always been crucial for organizations. Yet over the past few years, because of both an increase in the frequency of attacks and compliance requirements (HIPAA, PCI, SOC, FISMA, etc.), employing security mechanisms and standards has become a top priority.

Because log data contains a wealth of valuable information on what is actually happening in real time within running processes, it should come as

little surprise that security is fast becoming a strong use case for the ELK Stack.

Despite the fact that as a standalone stack, ELK does not come with security features built-in, the fact that you can use it to centralize logging from your environment and create monitoring and security-orientated dashboards has led to the integration of the stack with some prominent security standards.

Here are two examples of how the ELK Stack can be implemented as part of a security-first deployment.

1.Anti-DDoS

Once a DDoS attack is mounted, time is of the essence. Quick identification is key to minimizing the damage, and that's where log monitoring comes into the picture. Logs contain the raw footprint generated by running processes and thus offer a wealth of information on what is happening in real time.

Using the ELK Stack, organizations can build a system that aggregates data from the different layers in an IT environment (web server, databases, firewalls, etc.), process the data for easier analysis and visualizes the data in powerful monitoring dashboards.

2.SIEM

SIEM is an approach to enterprise security management that seeks to provide a holistic view of an organization's IT security. The main purpose of SIEM is to provide a simultaneous and comprehensive view of your IT security. The SIEM approach includes a consolidated dashboard that allows you to identify activity, trends, and patterns easily. If implemented correctly, SIEM can prevent legitimate threats by identifying them early, monitoring online activity, providing compliance reports, and supporting incident-response teams.

The ELK Stack can be instrumental in achieving SIEM. Take an AWS-based environment as an example. Organizations using AWS services have a large amount of auditing and logging tools that generate log data, auditing information and details on changes made to the configuration of the service. These distributed data sources can be tapped and used together to give a good and centralized security overview of the stack.

Read more about SIEM and ELK [here](#).

Business Intelligence (BI)

Business Intelligence (BI) is the use of software, tools, and applications to analyze an organization's raw data with the goal of optimizing decisions, improving collaboration, and increasing overall performance.

The process involves collecting and analyzing large sets of data from varied data sources: databases, supply chains, personnel records, manufacturing data, sales and marketing campaigns, and more. The data itself might be stored in internal data warehouses, private clouds or public clouds, and the engineering involved in extracting and processing the data (ETL) has given rise to a number of technologies, both proprietary and open source.

As with the previous use cases outlined here, the ELK Stack comes in handy for pulling data from these varied data sources into one centralized location for analysis. For example, we might pull web server access logs to learn how our users are accessing our website, We might tap into our CRM system to learn more about our leads and users, or we might check out the data our marketing automation tool provides.

There are a whole bunch of proprietary tools used for precisely this purpose. But the ELK Stack is a cheaper and open source option to perform almost all of the actions these tools provide.

SEO

Technical SEO is another edge use case for the ELK Stack but a relevant one nonetheless. What has SEO to do with ELK? Well, the common denominator is of course logs.

Web server access logs (Apache, nginx, IIS) reflect an accurate picture of who is sending requests to your website, including requests made by bots belonging to search engines crawling the site. SEO experts will be using this data to monitor the number of requests made by Baidu, BingBot, GoogleBot, Yahoo, Yandex and others.

Technical SEO experts use log data to monitor when bots last crawled the site but also to optimize crawl budget, website errors and faulty redirects, crawl priority, duplicate crawling, and plenty more. Check out our guide on [how to use log data for technical SEO](#).

Integrations

Almost any data source can be tapped into to ship log data into the ELK Stack. What method you choose will depend on your requirements, specific environment, preferred toolkit, and many more.

Over the last few years, we have written a large number of articles describing different ways to integrate the ELK Stack with different systems, applications and platforms. The method varies from a data source to data source — it could be a Docker container, Filebeat or another beat, Logstash and so forth. Just take your pick.

Below, is a list of these integrations just in case you're looking into implementing it. We've tried to categorize them into separate categories for easier navigation.

Please note that most include Logz.io-specific instructions as well, including ready-made dashboards that are part of our ELK Apps library. Integrations with instructions for integrating with the Logz.io ELK are marked.

<https://logz.io/learn/complete-guide-elk-stack/>

<https://logz.io/blog/elk-siem/>

<https://mohomedarfath.medium.com/siem-implementation-with-elk-stack-for-windows-and-linux-791395df470b>

Winlogbeat configuration options

Winlogbeat has a number of configuration options that are worth mentioning before configuring our specific logging pipeline. These configurations are made in the Winlogbeat configuration file at *C:\Program Files\Winlogbeat\winlogbeat.yml*.

First, in the ‘Winlogbeat specific options’ section, you can define which specific event logs you want to monitor. By default, Winlogbeat is set to monitor application, security, and system logs.

A tip: If you’re not sure which event logs are available, just run *Get-EventLog ** in PowerShell. You should get an output like this:

| Max(K) | Retain | OverflowAction | Entries | Log |
|--------|--------|-------------------|---------|------------------|
| ----- | ----- | ----- | ----- | --- |
| 20,480 | 0 | OverwriteAsNeeded | 386 | Application |
| 10,240 | 0 | OverwriteAsNeeded | 31 | EC2ConfigService |
| 20,480 | 0 | OverwriteAsNeeded | 0 | HardwareEvents |
| 512 | 7 | OverwriteOlder | 0 | Internet |
| | | Explorer | | |

| | | |
|-----------------|---------------------|----------------|
| 20,480 | 0 OverwriteAsNeeded | 0 Key |
| Management Serv | | |
| 20,480 | 0 OverwriteAsNeeded | 1,724 Security |
| 20,480 | 0 OverwriteAsNeeded | 929 System |
| 15,360 | 0 OverwriteAsNeeded | 39 Windows |
| PowerShell | | |
| Copy | | |

Next, in the General section of the configuration file, you can add the name of the log shipper that is publishing the network data (the name is used for grouping data with a single name identifier). You can also add additional fields to the log messages such as *application_name* and *environment*.

The Outputs sections are where we configure the location to which we want to forward the logs. By default, a local Elasticsearch installation is defined as the output:

```
output.elasticsearch:  
  hosts:  
    - localhost:9200  
Copy
```

Of course, you can decide to output the event logs to Logstash instead — as we will do in the next step.

Last but not least, we can set the logging level in the Logging section to critical, error, warning, info, or debug.

Configuring our logging pipeline

Now that we have understood our configuration options, it's time to configure Winlogbeat to ship event logs to the Logz.io ELK Stack.

Open the Winlogbeat configuration file at: *C:\Program Files\Winlogbeat\winlogbeat.yml* and paste the following configuration:

```
winlogbeat.event_logs:
  - name: Application
    ignore_older: 72h
  - name: Security
  - name: System

  fields:
    logzio_codec: json
    token: UfKqCazQjUYnBNcJqSryIRyDIjExjwIZ
  fields_under_root: true

output.logstash:
  # The Logstash hosts
  hosts: ["listener.logz.io:5015"]
  # Optional TLS. By default is off.
  # List of root certificates for HTTPS server verifications
  tls.certificateAuthorities: ['c:\Program Files\Winlogbeat\COMODORSADomainValidationSecureServerCA.crt']
  Copy
```

A few notes on this configuration.

In this case, we are sending the event logs to the Logz.io ELK, so we commented out the Elasticsearch as an output section. Also, there are additional fields here that are specific to shipping to the Logz.io ELK Stack: logz.io_codec, token, and the TLS certificate

path (you will need to download your own certificate to ship to Logz.io).

If you're shipping to your own Elasticsearch or Logstash instance, you can use the default settings in the file and omit the addition of these additional fields.

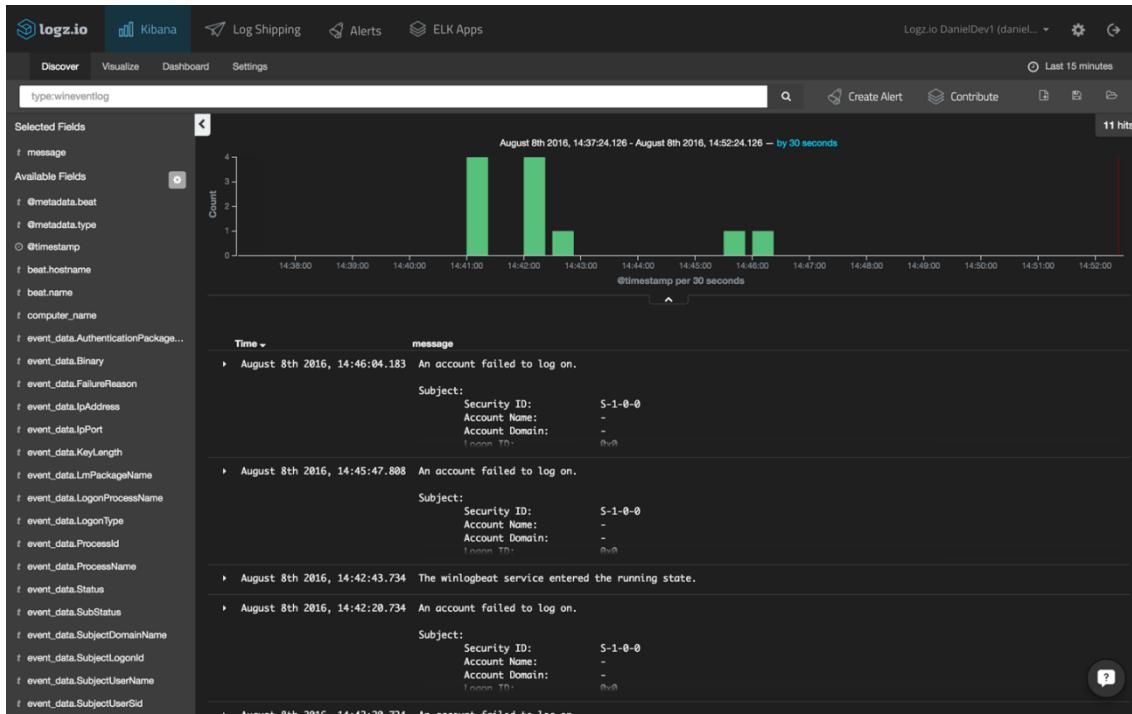
Save the file, and start the service. You can use the Windows service manager or PowerShell to do this:

```
PS C:\Program Files\Winlogbeat> Start-Service winlogbeat  
Copy
```

Analyzing and visualizing event logs

Open Kibana, and you should see your event logs displayed. If you're already shipping logs from a different data source, you can differentiate the two streams of data using the following query in Kibana:

```
type: wineventlog  
Copy
```



Select one of the entries to view all of the fields that are available for analysis. Select the JSON tab to view the event logs as they are indexed by Elasticsearch. For example, here is the event log report that the Winlogbeat service has begun running:

```
{
  "_index": "logz-dkdhmyttiijymjdammbltqlivlylpzwqb-160808_v1",
  "_type": "wineventlog",
  "_id": "AVZp9m4P-SDvkjEsrQTt",
  "_score": null,
  "_source": {
    "computer_name": "WIN-E9CD0MMN9GJ",
    "process_id": 676,
    "keywords": [
      "Classic"
    ],
    "logzio_codec": "json",
    "log_name": "System",
    "level": "Information",
    "@metadata": {
      "beat": "winlogbeat",
      "type": "wineventlog"
    }
}
```

```
"record_number": "24082",
"event_data": {
    "Binary": "770069006E006C006F00670062006500610074002F0034000000",
    "param1": "winlogbeat",
    "param2": "running"
},
"type": "wineventlog",
"message": "The winlogbeat service entered the running state.",
"tags": [
    "beats-5015",
    "_logzio_codec_json",
    "_jsonparsefailure"
],
"thread_id": 3876,
"@timestamp": "2016-08-08T11:42:43.734+00:00",
"event_id": 7036,
"provider_guid": "{555908d1-a6d7-4695-8e1e-26931d2012f4}",
"beat": {
    "hostname": "WIN-E9CD0MMN9GJ",
    "name": "WIN-E9CD0MMN9GJ"
},
"source_name": "Service Control Manager"
},
"fields": {
    "@timestamp": [
        1470656563734
    ]
},
"highlight": {
    "type": [
        "@kibana-highlighted-field@wineventlog@/kibana-highlighted-field@"
    ]
},
"sort": [
    1470656563734
]
}
}
```

Copy

You can now use Kibana to query the data stored in Elasticsearch. Querying is an art unto itself, and we cover some of the common methods in [Elasticsearch queries](#) guide.

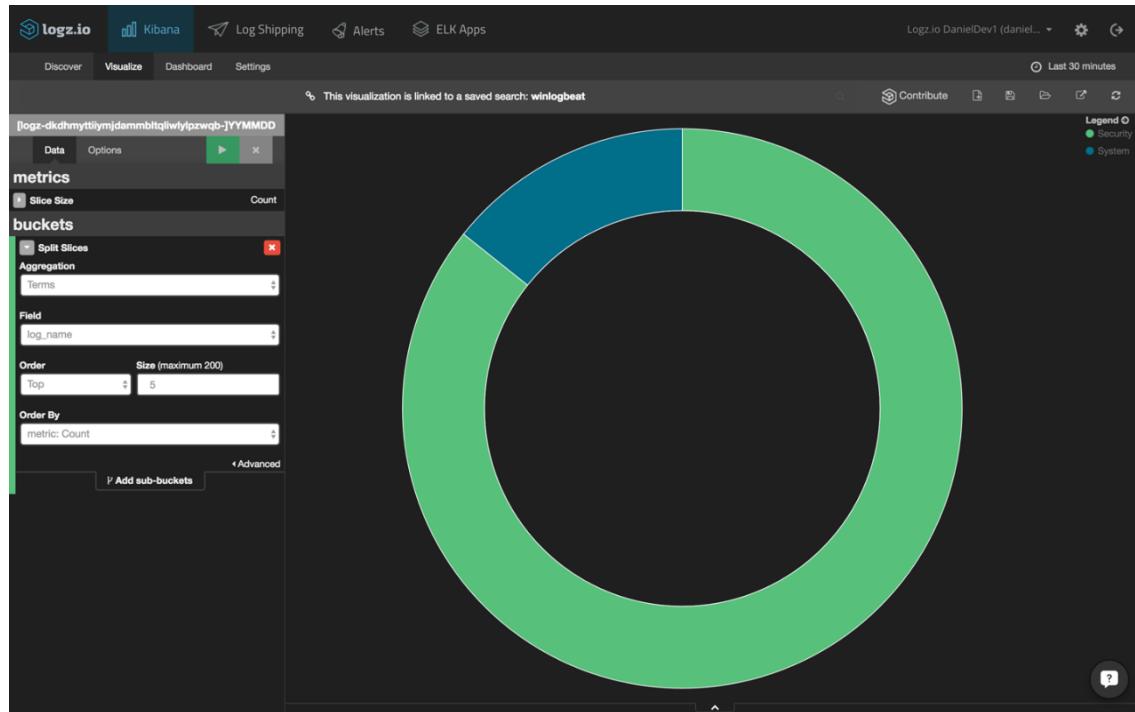
As an example, say you would like to see a breakdown of the different event types. Using the query above as the basis for a pie-chart visualization, we are going to use the following configuration:

The screenshot shows the Kibana interface with the 'Visualize' tab selected. The main area displays a configuration for a bucket, specifically a 'Terms' aggregation on the 'log_name' field. The configuration includes:

- Aggregation:** Terms
- Field:** log_name
- Order:** Top, Size (maximum 200)
- Order By:** metric: Count

At the bottom right of the configuration area, there is a link labeled "Advanced". At the very bottom of the interface, there is a button labeled "Add sub-buckets".

We're using a split-slice aggregation using the "log_name" field. Here is the result of this configuration (hit the green play button to preview the visualization):



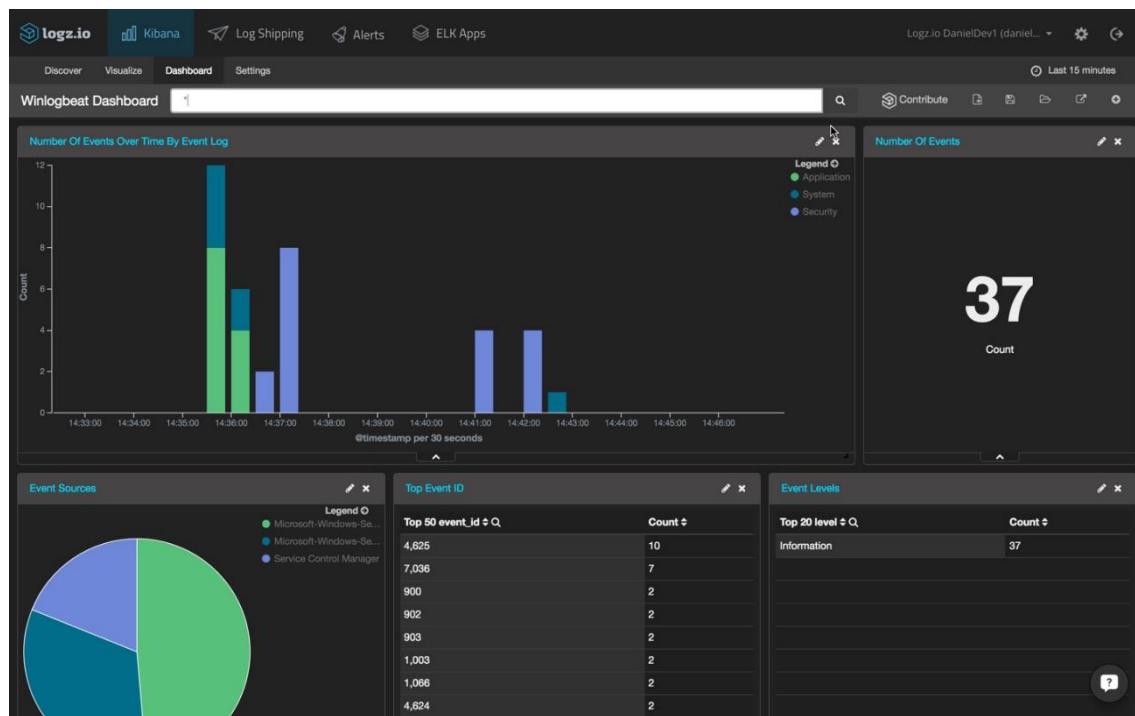
Now, Logz.io ships with [a library of pre-made Kibana searches, alerts, visualizations and dashboards](#) tailored for specific log types — including, Windows event logs. Saving you the time for building different visualizations, you can hit the ground running with a ready-made dashboard.

To install this dashboard, simply [open ELK Apps](#), search for Winlogbeat in the search box, and install the dashboard.

ELK APPS

The screenshot shows the ELK Apps interface with the "Winlogbeat" dashboard selected. The top navigation bar includes links for Dashboards, Visualizations, Alerts, and Searches, along with a dropdown for "Most popular". The main area displays the "wineventlog Dashboard by Anonymous" which contains several visualizations: "Event Log Count", "Event Log Sources", "Event Log Levels", "Event Log IDs", and "Event Log Searches". Below the dashboard, a summary states "Winlogbeat Dashboard" and "Dashboard for Winlogbeat metrics". At the bottom, there are like and share buttons.

When you open the dashboard, you will see a series of visualizations: number of events over time, number of events, event sources, top event IDs, event levels, and Windows event log searches.



Of course, you can customize these visualizations any way that you like or create various other visualizations based on the indexed fields — this functionality is why Kibana is so powerful.

<https://www.elastic.co/guide/en/security/current/mimikatz-memssp-log-file-detected.html>

<https://blog.netwrix.com/2021/11/30/how-to-detect-pass-the-hash-attacks/>

<https://www.linkedin.com/pulse/elastic-stack-automated-threat-response-sure-why-hung-nguyen/>

https://www.youtube.com/watch?v=gC3g8bLkhYg&ab_channel=I.TSecurityLabs

https://www.youtube.com/watch?v=0OzMgcPVMY&ab_channel=Elastic

https://www.youtube.com/watch?v=h_915T2-n90&ab_channel=FaculdadeVincit

https://www.youtube.com/watch?v=lwlV3wVX4xs&ab_channel=I.TSecurityLabs

Splunk monitors Windows event log data with Splunk

Windows generates log data during the course of its operations. The Windows Event Log service handles nearly all of this communication. It gathers log data that installed applications, services, and system processes publish and places the log data into event log channels. Programs such as Microsoft Event Viewer subscribe to these log channels to display events that have occurred on the system.

You can monitor event log channels and files that are on the local machine or you can collect logs from remote machines. The event log monitor runs once for every event log input that you define.

To monitor Windows Event Log channels in Splunk Cloud Platform, use a Splunk universal or heavy forwarder to collect the data and forward it to your Splunk Cloud Platform deployment. As a best practice, use the Splunk Add-on for Windows to simplify the process of getting data into Splunk Cloud Platform. For instructions on using the Splunk Add-on for Windows to get data into Splunk Cloud Platform, see [Get Windows Data Into Splunk Cloud](#) in the *Splunk Cloud Admin Manual*.

Why monitor event logs?

Windows event logs are the core metric of Windows machine operations. If there is a problem with your Windows system, the Event Log service has logged it. The Splunk platform indexing, searching, and reporting capabilities make your logs accessible.

Requirements for monitoring event logs

| Activity | Requirements |
|----------|--------------|
| | |

| | |
|---------------------------|---|
| Monitor local event logs | <ul style="list-style-type: none"> The Splunk universal forwarder or Splunk Enterprise instance must run on Windows in the <i>Installation Manual</i>. The Splunk universal forwarder or Splunk Enterprise instance must run as the user to read all local event logs. |
| Monitor remote event logs | <ul style="list-style-type: none"> The universal forwarder or heavy forwarder must run on the Windows machine to collect event logs. The Splunk universal forwarder or heavy forwarder must run as a domain or restricted user to have read access to Windows Management Instrumentation (WMI) on the remote machine. See Windows user Splunk Enterprise should run as in the <i>Installation Manual</i>. The user that the forwarder runs as must have read access to the event logs you want to collect. |

Security and other considerations for collecting event log data from remote machines

You collect event log data from remote machines using a universal forwarder, a heavy forwarder, or WMI. As a best practice, use a universal forwarder to send event log data from remote machines to an indexer. See [The universal forwarder](#) in the *Universal Forwarder* manual for information about how to install, configure and use the forwarder to collect event log data. If you can't install a forwarder on the machine where you want to get data, you can use a WMI.

To install forwarders on your remote machines to collect event log data, install the forwarder as the Local System user on these machines. The Local System user has access to all data on the local machine, but not on remote machines.

To use WMI to get event log data from remote machines, you must ensure that your network and Splunk Enterprise instances are properly configured. Do not install Splunk software as the Local System user. The user you use to install the software determines the event logs that Splunk software has access to. See [Security and remote access considerations](#) for additional information on the requirements you must satisfy to collect remote data properly using WMI.

By default, Windows restricts access to some event logs depending on the version of Windows you run. For example, only members of the local Administrators or global Domain Admins groups can read the Security event logs by default.

How the Windows Event Log monitor interacts with Active Directory

When you set up an Event Log monitoring input for WMI, the input connects to an Active Directory (AD) domain controller to authenticate and, if necessary, performs any security ID (SID) translations before it begins to monitor the data.

The Event Log monitor uses the following logic to interact with AD after you set it up:

1. If you specify a domain controller when you define the input with the `evt_dc_name` setting in the `inputs.conf` file, then the input uses that domain controller for AD operations.
2. If you do not specify a domain controller, then the input does the following:
 1. The input attempts to use the local system cache to authenticate or resolve SIDs.
 2. If the monitor cannot authenticate or resolve SIDs that way, it attempts a connection to the domain controller that the machine that runs the input used to log in.
 3. If that does not work, then the input attempts to use the closest AD domain controller that has a copy of the Global Catalog.
3. If the domain controller that you specify is not valid, or a domain controller cannot be found, then the input generates an error message.

Collect event logs from a remote Windows machine

You have two choices to collect data from a remote Windows machine:

- Use a universal forwarder
- Use WMI

Use a universal or heavy forwarder

You can install a universal forwarder or a heavy forwarder on the Windows machine and instruct it to collect event logs. You can do this manually or use a deployment server to manage the forwarder configuration.

1. On the Windows machine for which you want to collect Windows Event Logs, download Splunk Enterprise or the universal forwarder software.
2. Run the universal forwarder installation package to begin the installation process.
3. When the installer prompts you, configure a receiving indexer.
4. When the installer prompts you to specify inputs, enable the event log inputs by checking the **Event logs** checkbox.
5. Complete the installation procedure.
6. On the receiving indexer, use Splunk Web to search for the event log data as in the following example:

```
host=<name of remote Windows machine> sourcetype=Wineventlog
```

For specific instructions to install the universal forwarder, see [Install a Windows universal forwarder](#) in the *Forwarder Manual*.

Use WMI

If you want to collect event logs remotely using WMI, you must install the universal or heavy forwarder to run as an Active Directory domain user. If the selected domain user is

not a member of the Administrators or Domain Admins groups, then you must configure event log security to give the domain user access to the event logs.

To change event log security to get access to the event logs from remote machines, you must meet the following requirements:

- Have administrator access to the machine from which you are collecting event logs.
- Understand how the Security Description Definition Language (SDDL) works and how to assign permissions with it. See [http://msdn.microsoft.com/en-us/library/aa379567\(v=VS.85\).aspx](http://msdn.microsoft.com/en-us/library/aa379567(v=VS.85).aspx) on the Microsoft website for more information.
- Decide how to monitor your data. See [Considerations for deciding how to monitor remote Windows data](#) for information on collecting data from remote Windows machines.

You can use the wevtutil utility to set event log security.

1. Download a Splunk Enterprise instance onto a Windows machine.
2. Double-click the installer file to begin the installation.
3. When the installer prompts you to specify a user, select **Domain user**.
4. On the next installer pane, enter the domain user name and password that you want Splunk Enterprise to use when it runs.
5. Follow the prompts to complete the installation of the software.
6. Once the software has installed, log in to the instance.
7. Use Splunk Web to add the remote event log input. See [Configure remote event log monitoring](#) later in this topic.

Anomalous machine names are visible in event logs on some systems

On some Windows systems, you might see some event logs with randomly-generated machine names. This is the result of those systems logging events before the user has named the system during the OS installation process.

This anomaly occurs only when you collect logs from versions of Windows remotely over WMI.

Configure local event log monitoring with Splunk Web

To get local Windows event log data, point your Splunk Enterprise instance at the Event Log service.

Go to the Add Data page

You can get there in two ways:

- Splunk Settings
- Splunk Home

From Splunk Settings:

1. Click **Settings > Data Inputs**.
2. Click **Local event log collection**.
3. Click **New** to add an input.

From Splunk Home:

1. Click the **Add Data** link in Splunk Home.
2. Click **Monitor** to monitor Event Log data on the local Windows machine, or **Forward** to forward Event Log data from another Windows machine. Splunk Enterprise loads the Add Data - Select Source page.
3. If you selected **Forward**, select or create the group of forwarders you want this input to apply to. See [Forward data](#) in this manual.
4. Click **Next**.

Select the input source

1. Select **Local Event Logs**
2. In the **Select Event Logs** list, select the Event Log channels you want this input to monitor.
3. Click each Event Log channel you want to monitor once. Splunk Enterprise moves the channel from the **Available items** window to the **Selected items** window.
4. To deselect a channel, click its name in the **Available Items** window. Splunk Enterprise moves the channel from the **Selected items** window to the **Available items** window.
5. To select or deselect all of the event logs, click the **add all** or **remove all** links.

Selecting all of the channels can result in the indexing of a lot of data.

6. Click **Next**.

Specify input settings

The **Input Settings** page lets you specify the application context, default host value, and index. All of these parameters are optional.

The **Host** field sets only the [**host**](#) field in the resulting events. It doesn't direct Splunk Enterprise to look on a specific machine on your network.

1. Select the appropriate **Application context** for this input.
2. Set the **Host** value. You have several choices for this setting. For more information about setting the host value, see [About hosts](#).
3. Set the **Index** that you want Splunk Enterprise to send data to. Leave the value as **default**, unless you defined multiple indexes to handle different types of events. In addition to indexes for user data, Splunk Enterprise has a number of utility indexes, which also appear in this dropdown box.
4. Click **Review**.

Review your choices

After you specify all your input settings, you can review your selections. Splunk Enterprise lists all options you selected, including the type of monitor, the source, the source type, the application context, and the index.

1. Review the settings.
2. If they do not match what you want, click the left angle bracket (<) to go back to the previous step in the wizard. Otherwise, click **Submit**.

Splunk Enterprise then displays the "Success" page and begins indexing the specified Event Log channels.

Configure remote event log monitoring with Splunk Web

The process for configuring remote event log monitoring is nearly identical to the process for monitoring local event logs.

1. Follow the instructions to get to the Add Data page. See [Go to the Add Data page](#).
2. Locate and select **Remote Event Logs**.
3. In the **Event Log collection name** field, enter a unique, memorable name for this input.
4. In the **Choose logs from this host** field, enter the host name or IP address of the machine that contains the Event Log channels you want to monitor.

Selecting all of the Event Log channels can result in the indexing of a lot of data.
5. Click the **Find logs** button to refresh the page with a list of available Event Log channels on the machine you entered.
6. Click once on each Event Log channel you want to monitor. Splunk Enterprise moves the channel from the **Available items** window to the **Selected items** window.
7. To deselect a channel, click its name in the **Available Items** window. Splunk Enterprise moves the channel from the **Selected items** window to the **Available items** window.
8. To select or deselect all of the event logs, click the **add all** or **remove all** links.
9. In the **Collect the same set of logs from additional hosts** field, enter the host names or IP addresses of additional machines that contain the Event Logs you selected previously. Separate multiple machines with commas.
10. Click the green **Next** button.
11. Follow the instructions to specify input settings. See [Specify input settings](#).
12. Follow the instructions to review your choices. See [Review your choices](#).

Use the inputs.conf configuration file to configure event log monitoring

On either a universal or heavy forwarder, you can edit the inputs.conf configuration file to configure Windows event log monitoring.

1. Using Notepad or a similar editor, open %SPLUNK_HOME%\etc\system\local\inputs.conf for editing. You might need to create this file if it doesn't exist.
2. Enable Windows event log inputs by adding input stanzas that reference Event Log channels.
3. Save the file and close it.
4. Restart the Splunk platform.

For more information on configuring data inputs with the inputs.conf file, see [Edit inputs.conf](#).

Specify global settings for Windows Event Log inputs

When you define Windows Event Log inputs in inputs.conf, make sure you explicitly specify global settings in the correct place.

If you specify global settings for Windows Event Log inputs, such as `host`, `sourcetype`, and so on, you can place those settings in one of the following areas:

- Under the `[WinEventLog]` global stanza. This stanza is equal to the `[default]` stanza for other monitoring inputs. For example:
 - `[default]`
 - `_meta = hf_proxy::meta_test`
 -
 - `[WinEventLog]`
 - `_meta = hf_proxy::meta_test`
 - `host = WIN2K16_DC`
 - `index = wineventlog`
 -
 - `[WinEventLog://Application]`
 - `disabled = 0`
- Under the Windows Event Log input stanza for the Event Log channel that you want to monitor. For example:
 - `[default]`
 - `_meta = hf_proxy::meta_test`
 -
 - `[WinEventLog]`
 - `host = WIN2K16_DC`
 - `index = wineventlog`
 -
 - `[WinEventLog://Application]`
 - `disabled = 0`
 - `_meta = hf_proxy::meta_test`

You can review the defaults for a configuration file by looking at the examples in %SPLUNK_HOME%\etc\system\default or at the spec file in the *Admin Manual*.

Event log monitor configuration values

Windows event log (*.evt) files are in binary format. You can't monitor them like you do a normal text file. The `splunkd` service monitors these binary files by using the appropriate APIs to read and index the data within the files.

Splunk Enterprise uses the following stanzas in inputs.conf to monitor the default Windows event logs:

```
# Windows platform specific input processor.  
[WinEventLog://Application]  
disabled = 0  
[WinEventLog://Security]  
disabled = 0  
[WinEventLog://System]  
disabled = 0
```

Monitor non-default Windows event logs

You can also configure Splunk Enterprise to monitor non-default Windows event logs. Before you can do this, you must import them to the Windows Event Viewer. After you import the logs, you can add them to your local copy of inputs.conf, as in the following example:

```
[WinEventLog://DNS Server]  
disabled = 0  
[WinEventLog://Directory Service]  
disabled = 0  
[WinEventLog://File Replication Service]  
disabled = 0
```

Use the Full Name log property in Event Viewer to specify complex Event Log channel names properly

You can use the `Full Name` Event Log property in Event Viewer to ensure that you specify the correct Event Log channel in an inputs.conf stanza.

For example, to monitor the Task Scheduler application log, `Microsoft-Windows-TaskScheduler-Operational`, do the following steps:

1. Open Event Viewer.
2. Expand **Applications and Services Logs > Microsoft > Windows > TaskScheduler**.
3. Right-click **Operational** and select **Properties**.
4. In the dialog that appears, copy the text in the **Full Name** field.
5. Append this text into the `WinEventLog://` stanza of inputs.conf as in the following example:
6. `[WinEventLog://Microsoft-Windows-TaskScheduler/Operational]`
7. `disabled = 0`

Disable an event log stanza

To disable indexing for an event log, add `disabled = 1` after its listing in the stanza in `%SPLUNK_HOME%\etc\system\local\inputs.conf`.

Configuration settings for monitoring Windows Event Logs

Splunk software uses the following settings in `inputs.conf` to monitor Event Log files:

| Attribute | Description |
|---------------------------------|---|
| <code>start_from</code> | <p>How to read events.</p> <p>Acceptable values are <code>oldest</code>, meaning read logs from the oldest to the newest, and <code>newest</code>, meaning read logs from the newest to the oldest.</p> <p>You can't set this attribute to <code>newest</code> while also setting the <code>current_only</code> attribute to <code>1</code>.</p> |
| <code>current_only</code> | <p>How to index events.</p> <p>Acceptable values are <code>1</code>, where the input acquires events that arrive after the input starts for the first time, like <code>tail -f</code> on *nix systems, or <code>0</code>, where the input gets existing events in the log and then continues to monitor incoming events in real time.</p> <p>You can't set this attribute to <code>newest</code> while also setting the <code>current_only</code> attribute to <code>1</code>.</p> |
| <code>checkpointInterval</code> | <p>How frequently, in seconds, the Windows Event Log input saves a checkpoint.</p> <p>Checkpoints store the eventID of acquired events to enable Splunk software to resume monitoring at the correct event after a shutdown or outage.</p> |
| <code>evt_resolve_ad_ds</code> | <p>The domain controller Splunk software uses to interact with Active Directory while indexing Windows Event Log channels. Valid only when you set the <code>evt_resolve_ad_obj</code> attribute to <code>1</code> and omit the <code>evt_dc_name</code> attribute.</p> <p>Valid values are <code>auto</code>, meaning to use the nearest domain controller to bind to objects for object resolution, or <code>PDC</code>, meaning to bind to the primary domain controller for the site that the host is in. If you also set the <code>evt_dc_name</code> attribute, Splunk software ignores this attribute.</p> |
| <code>evt_resolve_ad_obj</code> | <p>How Splunk software interacts with Active Directory while indexing Windows Event Log channels. Valid values are <code>1</code>, meaning resolve Active Directory objects like GUID and SID Unique Identifier (GUID) and Security Identifier (SID) objects to their canonical names for a specific Windows event log channel, and <code>0</code>, meaning not to attempt any resolution.</p> |

When you set this value to `1`, you can optionally specify the Domain Controller name or DNS name of the domain to bind to, which Splunk software uses to resolve the Active Directory objects. If you don't set this value, or if you set it to `0`, Splunk software does not attempt to resolve the AD objects.

`evt_dc_name`

Which Active Directory domain controller to bind to resolve AD objects. This name can be the NetBIOS name of the domain controller, the fully-qualified DNS name of the domain controller, or an environment variable name specified as `$Environment_variable`.

If you set this attribute, then Splunk software ignores the `evt_resolve_ad_ds` attribute, which controls how the software determines the best domain controller to bind to for AD object resolution.

If you specify an environment variable, you must prepend a dollar sign (`$`) to the environment variable name. Splunk software uses the specified environment variable as the domain controller to connect to for AD object resolution. For example, to use the `%LOGONSERVER%` variable, specify `evt_dc_name = $logonserver`.

You can precede either format with two backslash characters. This attribute does not have a default.

`evt_dns_name`

The fully-qualified DNS name of the domain to bind to resolve AD objects.

`evt_exclude_fields`

A list of Windows Event Log fields that the Windows Event Log input is to exclude when it ingests Windows Event Log data. When you specify this setting, the input removes both the key and value data for the fields you exclude. This setting works similarly to the `suppress_*` settings, but unlike those settings, this setting is valid for Windows Event Log fields, and excludes fields that you might have included in a collision list. When this collision happens, the instance logs an error. See "Create advanced filters with 'whitelist' and 'blacklist'" later in this topic for the list of Windows Event Log fields that you can exclude.

`suppress_text`

Whether to include the message text that comes with a security event. A value of `1` suppresses the message text, and a value of `0` preserves the text.

`use_old_eventlog_api`

Whether or not to read Event Log events with the Event Logging API.

This is an advanced setting. Contact Splunk Support before you change it.

If set to true, the input uses the Event Logging API instead of the Windows Event Log API to read from the Event Log on Windows Server 2008, Windows Vista, and higher installations.

`use_threads`

Specifies the number of threads, in addition to the default writer thread, that can be created to filter events with the allow list/deny list regular expression.

This is an advanced setting. Contact Splunk Support before you change it.

The maximum number of threads is 15.

`thread_wait_time_msec`

The interval, in milliseconds, between attempts to re-read Event Log files when an error occurs.

This is an advanced setting. Contact Splunk Support before you change it.

`suppress_checkpoint`

Whether or not the Event Log strictly follows the `checkpointInterval` setting when it saves a checkpoint.

This is an advanced setting. Contact Splunk Support before you change it.

By default, the Event Log input saves a checkpoint from between zero and `checkpointInterval` seconds, depending on incoming event volume.

`suppress_sourcename`

Whether or not to exclude the `sourcename` field from events.

This is an advanced setting. Contact Splunk Support before you change it.

When set to true, the input excludes the `sourcename` field from events and throughput performance (the number of events processed per second) improves.

`suppress_keywords`

Whether or not to exclude the `keywords` field from events.

This is an advanced setting. Contact Splunk Support before you change it.

When set to true, the input excludes the `keywords` field from events and throughput performance (the number of events processed per second) improves.

`suppress_type`

Whether or not to exclude the `type` field from events.

This is an advanced setting. Contact Splunk Support before you change it.

When set to true, the input excludes the `type` field from events and throughput performance (the number of events processed per second) improves.

`suppress_task`

Whether or not to exclude the `task` field from events.

This is an advanced setting. Contact Splunk Support before you change it.

When set to true, the input excludes the `task` field from events and throughput performance (the number of events processed per second) improves.

`suppress_opcode`

Whether or not to exclude the `opcode` field from events.

This is an advanced setting. Contact Splunk Support before you change it.

When set to true, the input excludes the `opcode` field from events and throughput performance (the number of events processed per second) improves.

`whitelist`

Whether to index events that match the specified text string. This attribute is optional.

You can specify one of two formats:

- One or more Event Log event codes or event IDs (Event Code/ID format.)
- One or more sets of keys and regular expressions (Advanced filtering format.)

You cannot mix formats in a single entry. You also cannot mix formats in the same stanza.

Allow lists are processed first, then deny lists. If no allow list is present, the Splunk platform indexes all events. If a file matches the regexes in both the deny list and allow list settings, the file is NOT monitored. Deny lists take precedence over allow lists.

When you use the Event Code/ID format:

- For multiple codes/IDs, separate the list with commas.
- For ranges, use hyphens (for example "0-1000,5000-1000").

When using the advanced filtering format:

- Use `=` between the key and the regular expression that represents your filter (for example `whitelist = EventCode=%^1([8-9])$%`)
- You can have multiple key/regular expression sets in a single advanced filtering entry. The Splunk platform conjuncts the sets logically. This means that the entry is valid only if all of the sets in the entry are true.
- You can specify up to 10 whitelists per stanza by adding a number to the end of the `whitelist` attribute, for example `whitelist1...whitelist10`.

`blacklist`

Do not index events that match the text string specified. This attribute is optional.

You can specify one of two formats:

- One or more Event Log event codes or event IDs (Event Log code/ID format.)
- One or more sets of keys and regular expressions. (Advanced filtering format.)

You cannot mix formats in a single entry. You also cannot mix formats in the same stanza.

Allow lists are processed first, then deny lists. If no deny list is present, the Splunk platform indexes all events.

When using the Event Log code/ID format:

- For multiple codes/IDs, separate the list with commas.
- For ranges, use hyphens (for example `0-1000,5000-1000`).

When using the advanced filtering format:

- Use `=` between the key and the regular expression that represents your filter (for example `blacklist = EventCode=%^1([8-9])$%`)

- You can have multiple key/regular expression sets in a single advanced filtering entry. The Splunk platform conjuncts the sets logically. This means that the entry is valid only if all of the sets in the entry are true.
- You can specify up to 10 deny lists per stanza by adding a number to the end of the `blacklist` attribute, for example `blacklist1...blacklist10`.

`renderXml`

Render event data as extensible markup language (XML) supplied by the Windows Event Log subsystem. This setting is optional.

A value of `1` or `true` means to render the events as XML. A value of `0` or `false` means to render the events as plain text.

If you set `renderXml` to `true`, and if you want to also create allow lists or deny lists, filter event data, you must use the `$XmlRegex` special key in your allow lists or deny lists.

`index`

The index that this input is to send the data to.

`disabled`

Whether or not the input is to run.

Valid values are `0`, meaning that the input is to run, and `1`, meaning that the input is not run.

Use the Security event log to monitor changes to files

You can monitor changes to files on your system by enabling security auditing on a set of files or directories and then monitoring the Security event log channel for change events. The event log monitoring input includes three attributes which you can use in inputs.conf. For example:

```
[WinEventLog://Security]
disabled = 0
start_from = oldest
current_only = 0
evt_resolve_ad_obj = 1
checkpointInterval = 5
# only index events with these event IDs.
whitelist = 0-2000,3001-10000
# exclude these event IDs from being indexed.
blacklist = 2001-3000
```

To enable security auditing for a set of files or directories, read Auditing Security Events How To on MS Technet at <http://technet.microsoft.com/en-us/library/cc727935%28v=ws.10%29.aspx>.

You can also use the `suppress_text` attribute to include or exclude the message text that comes with a security event.

When you set `suppress_text` to 1 in a Windows Event Log Security stanza, the entire message text does not get indexed, including any contextual information about the security event. If you need this contextual information, do not set `suppress_text` in the stanza.

See the following example to include or exclude message text:

```
[WinEventLog://Security]
disabled = 0
start_from = oldest
current_only = 0
evt_resolve_ad_obj = 1
checkpointInterval = 5
# suppress message text, we only want the event number.
suppress_text = 1
# only index events with these event IDs.
whitelist = 0-2000,2001-10000
# exclude these event IDs from being indexed.
blacklist = 2001-3000
```

To use a specific domain controller, set the `evt_dc_name` attribute:

```
[WinEventLog://Security]
disabled = 0
start_from = oldest
current_only = 0
evt_resolve_ad_obj = 1
evt_dc_name = boston-dc1.contoso.com
checkpointInterval = 5
# suppress message text, we only want the event number.
suppress_text = 1
# only index events with these event IDs.
whitelist = 0-2000,2001-10000
# exclude these event IDs from being indexed.
blacklist = 2001-3000
```

To use the primary domain controller to resolve AD objects, set the `evt_resolve_ad_ds` attribute to `PDC`. Otherwise, it locates the nearest domain controller.

```
[WinEventLog://Security]
disabled = 0
start_from = oldest
current_only = 0
evt_resolve_ad_obj = 1
evt_resolve_ad_ds = PDC
checkpointInterval = 5
# suppress message text, we only want the event number.
suppress_text = 1
# only index events with these event IDs.
whitelist = 0-2000,2001-10000
# exclude these event IDs from being indexed.
blacklist = 2001-3000
```

<https://docs.splunk.com/Documentation/Splunk/9.0.4/Data/MonitorWindowsevenlogdata>

Attackers can make users run malicious code or persist on an endpoint by targeting file extensions that users are familiar with. For example, if users see that a file ends in .doc or .docx, they will assume that it is a Microsoft Word document and expect that double-clicking will open it using winword.exe. The user will typically also assume that the .docx file is safe. Attackers take advantage of this expectation by obfuscating the true file extension.

These searches help you detect this type of abuse of file extensions and Windows file associations by searching for the execution of files with multiple extensions in the file name, a common technique used by attackers to obscure the true file extension.

Required data

- Normalized [endpoint data](#) that records process activity from your hosts. For information on installing and using the CIM, see the [Common Information Model documentation](#).

How to use Splunk software for this use case

This search looks for processes launched from files that have double extensions in the file name. This is typically done to obscure the true file extension and make it appear as though the file being accessed is a data file, as opposed to executable content.

```
| tstats summariesonly=false allow_old_summaries=true  
count, min(_time) AS firstTime, max(_time) AS lastTime FROM  
datamodel=Endpoint.Processes WHERE  
("Processes.process"="*.doc.exe OR  
"Processes.process"="*.htm.exe OR  
"Processes.process"="*.html.exe OR  
"Processes.process"="*.txt.exe OR  
"Processes.process"="*.pdf.exe OR  
"Processes.process"="*.doc.exe) BY "Processes.dest",  
"Processes.user", "Processes.process",  
"Processes.parent_process"  
  
| convert timeformat="%Y-%m-%dT%H:%M:%S" ctime(firstTime)  
| convert timeformat="%Y-%m-%dT%H:%M:%S" ctime(lastTime)  
| rename "Processes.*" AS "*"
```

https://lantern.splunk.com/Splunk_Platform/Use_Cases/Use_Cases_Security/Threat_Hunting/Detecting_Windows_file_extension_abuse

Monitor Windows host information

You can monitor detailed statistics about your local Windows machine with the Splunk platform.

If you use Splunk Cloud Platform, you must collect Windows host information with a forwarder and send it to your Splunk Cloud Platform deployment. Follow these high-level steps:

1. Install the universal forwarder on the Windows machine that you want to collect the host information.
2. Install the app to connect the universal forwarder to the Splunk Cloud Platform instance.
3. Configure the forwarder to collect the Windows host information.

Both full instances of Splunk Enterprise and universal forwarders support direct, local collection of host information. On these instance types, the Windows host monitor input runs as a process called `splunk-winhostmon.exe`. This process runs once for every Windows host monitoring input that you define at the interval that you specify in the input. On Splunk Enterprise, you can configure host monitoring using Splunk Web, and on the universal forwarder you can configure the inputs using the `inputs.conf` configuration file.

Why monitor host information?

You can monitor hosts to get detailed information about your Windows machines. You can monitor changes to the system, such as installation and removal of software, the starting and stopping of services, and uptime. When a system failure occurs, you can use Windows host monitoring information as a first step into the forensic process. With the Splunk Search Processing Language, you can give your team statistics on all machines in your Windows network.

The Splunk platform can collect the following information about a Windows machine:

General computer

The make and model of the computer, its host name, and the Active Directory domain it is in.

Operating system

The version and build number of the operating system and service packs installed on the computer, the computer name, the last time it started, the amount of installed and free memory, and the system drive.

Processor

The make and model of the CPUs installed in the system, their speed and version, the number of processors and cores, and the processor ID.

Disk

A list of all drives available to the system and, if available, their file system type and total and available space.

Network adapter

Information about the installed network adapters in the system, including manufacturer, product name, and MAC address.

Service

Information about the installed services on the system, including name, display name, description, path, service type, start mode, state, and status.

Process

Information on the running processes on the system, including the name, the command line with arguments), when they were started, and the executable path.

Requirements

To monitor host information, you must fulfill the following requirements:

- Splunk Cloud Platform must receive Windows host information from a forwarder.
- The forwarder must run on Windows. See [Install on Windows](#) in the *Installation Manual*.
- To read all Windows host information locally, the forwarder must run as the Local System Windows user or a local administrator user.

Security and remote access considerations

The universal forwarder must run as the Local System user to collect Windows host information by default.

Where possible, use a universal forwarder to send Windows host information from remote machines to Splunk Cloud Platform or a Splunk Enterprise indexer. You must use a universal forwarder to send Windows host information to Splunk Cloud Platform. Review the [Forwarder Manual](#) for information about how to install, configure, and use the universal forwarder to collect Windows host data.

If you choose to install forwarders on your remote machines to collect Windows host data, then you can install the forwarder as the Local System user on these machines. The Local System user has access to all data on the local machine, but not on remote machines.

If you run Splunk Enterprise or the universal forwarder as a user other than the Local System user, then that user must have local administrator rights and other permissions on the machine that you want to collect host data. See [Choose the Windows user Splunk Enterprise should run as](#) in the *Installation Manual*.

Use the inputs.conf configuration file to configure host monitoring

To collect Windows host information on your Splunk Cloud Platform instance, you must configure a universal forwarder on the Windows machine that you want to collect host information. Then, you can send the data to Splunk Cloud Platform.

You can edit inputs.conf to configure host monitoring. For more information on how to edit configuration files, see [About configuration files](#) in the *Admin Manual*.

You can also configure this file directly on a Splunk Enterprise instance.

To configure host monitoring on inputs.conf, follow these steps:

1. On the machine that you want to collect Windows host information, install a universal forwarder.
2. Download and install the Splunk Cloud Platform universal forwarder credentials package on the forwarder.

3. On the forwarder, use a text editor to create an inputs.conf configuration file in %SPLUNK_HOME%\etc\system\local and open it for editing.
4. In the same text editor, open %SPLUNK_HOME%\etc\system\default\inputs.conf and review it for the Windows event log inputs you want to enable.
5. Copy the Windows event log input stanzas you want to enable from %SPLUNK_HOME%\etc\system\default\inputs.conf.
6. Paste the stanzas you copied into the %SPLUNK_HOME%\etc\system\local\inputs.conf file.
7. Make edits to the stanzas that you copied to the %SPLUNK_HOME%\etc\system\local\inputs.conf file to collect the Windows event log data you want.
8. Save the %SPLUNK_HOME%\etc\system\local\inputs.conf file and close it.
9. Restart the universal forwarder.

When the Splunk platform indexes data from Windows host monitoring inputs, it sets the [source](#) for received events to `windows`. It sets the [source type](#) of the incoming events to `WinHostMon`.

Windows host monitor configuration values

Splunk Enterprise and the universal forwarder use the following settings in the inputs.conf configuration file to monitor Windows host information.

| Setting | Required? | Description |
|-----------------------|-----------|--|
| <code>interval</code> | Yes | How often, in seconds, to poll for new data. If you set the interval to a negative number, the input does not run. If you set the interval to zero, the input runs one time. If you do not define this setting, the input does not run. |
| <code>type</code> | Yes | The type of host information to monitor. Can be one of <code>Computer</code> , <code>operatingSystem</code> , <code>processor</code> , <code>disk</code> , <code>networkAdapter</code> , <code>service</code> , <code>process</code> , or <code>file</code> . The input does not run if this setting is not present. |
| <code>disabled</code> | No | Whether or not to run the input. If you set this setting to <code>1</code> , then the platform ignores the <code>interval</code> setting. |

For examples, see Examples of Windows host monitoring configurations later in this topic.

Use Splunk Web to configure host monitoring

You can configure Windows host information on Splunk Web in Splunk Enterprise only. Follow these high-level steps to configure host monitoring through Splunk Web:

1. [Go to the Add Data page](#)
2. [Select the input source](#)
3. (Optional) [Specify input settings](#)
4. [Review your choices](#)

Go to the Add Data page

Follow these steps to get to the Add Data page from Settings:

1. Click **Settings > Data Inputs**.
2. Click **Files & Directories**.
3. Click **New Local File & Directory** to add an input.

Follow these steps to get to the Add Data page from your Splunk Enterprise home page:

1. Click **Add Data** on the page.
2. Click **Monitor** to monitor host information from the local Windows machine.

Select the input source

1. In the left pane, locate and select **Local Windows host monitoring**.
2. In the **Collection Name** field, enter a unique, memorable name for this input.
3. In the **Event Types** box, locate the host monitoring event types you want this input to monitor.
4. Click once on each type you want to monitor.
Splunk Enterprise moves the type from the **Available type(s)** window to the **Selected type(s)** window.
5. To deselect a type, click its name in the **Selected type(s)** window.
Splunk Enterprise moves the counter from the **Selected type(s)** window to the **Available type(s)** window.
6. (Optional) To select or deselect all of the types, click the **add all** or **remove all** links.

Selecting all of the types can index a lot of data and might exceed the data limits of your license.

7. In the **Interval** field, enter the time, in seconds, between polling attempts for the input.
8. Click **Next**.

Specify input settings

Go to the **Input Settings** page to specify the application context, default host value, and index. All of these parameters are optional.

1. Select the appropriate **Application context** for this input.

- Set the **Host** name. You have several choices for this setting. For more about setting the host value, see [About hosts](#).

Host sets the host field only in the resulting events. It does not configure Splunk Enterprise to look on a specific host on your network.

- Set the **Index** to send data to. Leave the value as **default**, unless you defined multiple indexes to handle different types of events. In addition to indexes for user data, the Splunk platform has multiple utility indexes, which also appear in this dropdown list.
- Click **Review**.

Review your choices

After specifying all your input settings, review your selections. Splunk Enterprise lists all options you selected, including the type of monitor, the source, the source type, the application context, and the index.

- Review the settings.
- If they do not match what you want, click the left-angle bracket (<) to go back to the previous step in the wizard. Otherwise, click **Submit**.

Splunk Enterprise then loads the Success page and begins indexing the specified host information.

When Splunk Enterprise indexes data from Windows host monitoring inputs, it sets the [source](#) for received events to `windows`. It sets the [source type](#) of the incoming events to `WinHostMon`.

Examples of Windows host monitoring configurations

The following examples of how to configure Windows host monitoring in inputs.conf.

```
# Queries computer information.  
[WinHostMon://computer]  
type = Computer  
interval = 300  
  
# Queries OS information.  
# 'interval' set to a negative number tells Splunk Enterprise to  
# run the input once only.  
[WinHostMon://os]  
type = operatingSystem  
interval = -1  
  
# Queries processor information.  
[WinHostMon://processor]  
type = processor
```

```

interval = -1

# Queries hard disk information.
[WinHostMon://disk]
type = disk
interval = -1

# Queries network adapter information.
[WinHostMon://network]
type = networkAdapter
interval = -1

# Queries service information.
# This example runs the input every 5 minutes.
[WinHostMon://service]
type = service
interval = 300

# Queries information on running processes.
# This example runs the input every 5 minutes.
[WinHostMon://process]
type = process
interval = 300

```

PowerShell Detections

The Splunk Threat Research Team has developed a set of detections to assist with getting started in detecting suspicious 4104 script block events.

| Analytic | Technique | Tactic | Notes |
|--|------------------|---|---|
| Detect Empire with PowerShell Script Block Logging | T1059.001 | Execution | Identifies two values that are always found in the default PowerShell-Empire payloads. |
| Detect Mimikatz With PowerShell Script Block Logging | T1059.001 | Execution ¹ | Identifies strings typically found in PowerShell script block code related to mimikatz. |
| Powershell Fileless Process Injection via GetProcAddress | T1059.001, T1055 | Execution , Defense Evasion , Evasion | Identifies the use of GetProcAddress within the script block. |

| | | Privilege Escalation | |
|--|------------------|--------------------------------------|---|
| Powershell Fileless Script Contains Base64 Encoded Content | T1059.001, T1027 | Execution | Identifies the use of Base64 within the script block. |
| Unloading AMSI via Reflection | T1562 | Defense Evasion | Identifies system.management.automation.amsi within the script block, typically found in encoded commands disabling AMSI. |
| PowerShell Domain Enumeration | T1059.001 | Execution | Identifies commands typically found with domain and trust enumeration. |
| PowerShell Loading .NET into Memory via System.Reflection.Assembly | T1059.001 | Execution | Identifies system.reflection.assembly within the script block being used, typically found in malicious PowerShell script execution. |
| Powershell Creating Thread Mutex | T1027.005 | Defense Evasion | Identifies the `mutex` function typically found and used in malicious PowerShell script execution. |
| Powershell Processing Stream Of Data | T1059.001 | Execution | Identifies suspicious PowerShell script execution via EventCode 4104 that is processing compressed stream data. |
| Powershell Using memory As Backing Store | T1140 | Defense Evasion | Identifies within the script block the use of memory stream as new object backstore. |

| | | | |
|---|-----------|---|---|
| <u>Recon AVProduct Through Pwh or WMI</u> | T1592 | <u>Reconnaissance</u> | Identifies suspicious PowerShell script execution performing checks to identify anti-virus products installed on the endpoint. |
| <u>Recon Using WMI Class</u> | T1592 | <u>Reconnaissance</u> | Identifies suspicious PowerShell where WMI is performing an event query looking for running processes or running services. |
| <u>WMI Recon Running Process or Services</u> | T1592 | <u>Reconnaissance</u> | Identifies suspicious PowerShell script execution where WMI is performing an event query looking for running processes or running services. |
| <u>Allow Inbound Traffic In Firewall Rule</u> | T1021.001 | <u>Lateral Movement</u> | Identifies suspicious PowerShell commands to allow inbound traffic inbound to a specific local port within the public profile. |
| <u>Mailsniper Invoke functions</u> | T1114.001 | <u>Collection</u> | Identifies known mailsniper.ps1 functions executed on an endpoint. |
| <u>Delete ShadowCopy With PowerShell</u> | T1490 | <u>Impact</u> | Identifies PowerShell commands to delete shadow copy using the WMIC PowerShell module. |
| <u>Powershell Enable SMB1Protocol Feature</u> | T1027.005 | <u>Defense Evasion</u> | Identifies enabling of smb1protocol through PowerShell Script Block logging. |

| | | | |
|---|-----------|---|---|
| Detect WMI Event Subscription Persistence | T1546.003 | Privilege Escalation, Persistence | Identifies WMI Event Subscription to establish persistence or perform privilege escalation. |
|---|-----------|---|---|

Responding to PowerShell with Automated Playbooks

The following community [Splunk SOAR](#) playbooks mentioned below can be used in conjunction with some of the previously described Powershell analytics.

| Name | Technique ID | Tactic | Description |
|--|-----------------------|-----------|--|
| Malware Hunt And Contain | T1204 | Execution | This playbook hunts for malware across managed endpoints, disables affected users, shuts down their devices, and blocks files by their hash from further execution via Carbon Black. |
| Email Notification for Malware | T1204 | Execution | This playbook tries to determine if a file is malware and whether or not the file is present on any managed machines. VirusTotal "file reputation" and PANW WildFire "detonate file" are used to determine if a file is malware, and CarbonBlack Response "hunt file" is used to search managed machines for the file. The results of these investigations are summarized in an email to the incident response team. |
| Block Indicators | T1204 | Execution | This playbook retrieves IP addresses, domains, and file hashes, blocks them on various |

| | | | |
|--|--|--|--|
| | | | services, and adds them to specific blocklists as custom lists |
|--|--|--|--|

For more information about how Splunk SOAR can accelerate investigations and remediations for your SOC, check out the upcoming [Splunk4Ninjas Splunk SOAR Hands On Workshop](#).
https://www.splunk.com/en_us/blog/security/powershell-detections-threat-research-release-august-2021.html

Summary of Logging Types

- **Transcript Logging:** As described by Microsoft, Transcript Logging provides a summary of what's happening in a PowerShell session as if you were looking over the shoulder of the person typing. It will provide the commands and the output of those commands. This is fairly verbose in most organizations and would require filtering (via the Splunk UF, in Splunk) or logging only on critical assets. What does it look like?

```

Start time: 20210810204526
Username: ATTACKRANGE\administrator
RunAs User: ATTACKRANGE\administrator
Machine: WIN-DC-26 (Microsoft Windows NT 10.0.14393.0)
Host Application: C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -nop -Command Write-Host 54797c2f-4d5b-44f6-8e0f-0048c8896ec2; Start-Sleep -Seconds 2; exit
Process ID: 6312
PSVersion: 5.1.14393.4530
PSEdition: Desktop
PSCOMPATIBLEVERSIONS: 1.0, 2.0, 3.0, 4.0, 5.0, 5.1.14393.4530
BuildVersion: 10.0.14393.4530
CLRVersion: 4.0.30319.42000
WSManStackVersion: 3.0
PSRemotingProtocolVersion: 2.3
SerializationVersion: 1.1.0.1
*****
*****
```

- **Module Logging:** module logging is a bit different in that it includes the command invocations and portions of the script. It's possible it will not have the entire details of the execution and the results. EventCode = 4103. What does it look like?

```

08/10/2021 08:04:35 PM
LogName=Microsoft-Windows-PowerShell/Operational
SourceName=Microsoft-Windows-PowerShell
EventCode=4103
EventType=4
Type=Information
ComputerName=win-dc-26.attackrange.local
User=NOSTRANSLATED
Sid=5-1-5-21-4085333548-3215320843-1534856649-500
SidType=0
TaskCategory=Executing Pipeline
OpCode=To be used when operation is just executing a method
RecordNumber=81465
Keywords=None
Message=CommandInvocation(Write-Host): "Write-Host"
ParameterBinding(Write-Host): name="Object"; value="65540fb3-05d7-49a4-8565-7cfdd103bea7"

Context:
Severity = Informational
Host Name = ConsoleHost
Host Version = 5.1.14393.4530
Host ID = cf76ef36-4808-41be-b03f-41f7c39a0ba7
Host Application = C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -nop -Command Write-Host 65540fb3-05d7-49a4-8565-7cfdd103bea7; Start-Sleep -Seconds 2; exit
Engine Version = 5.1.14393.4530
Runspace ID = f7e926f9-726d-40a2-a82b-1ef736caa911
Pipeline ID = 1
Command Name = Write-Host
Command Type = Cmdlet
Script Name =
Command Path =
Sequence Number = 16
User = ATTACKRANGE\administrator
Connected User =
Shell ID = Microsoft.PowerShell

```

- **Script Block Logging:** This is the raw, deobfuscated script supplied through the command line or wrapped in a function, script, workflow or similar. Think of everytime an adversary executes an encoded PowerShell script or command, script block logging provides that data in its raw form. EventCode = 4104. What does it look like?

```

Event
08/10/2021 08:04:34 PM
LogName=Microsoft-Windows-PowerShell/Operational
SourceName=Microsoft-Windows-PowerShell
EventCode=4104
EventType=5
Type=Verbose
ComputerName=win-dc-26.attackrange.local
User=NOT_TRANSLATED
Sid=S-1-5-21-4085333548-3215320843-1534856649-500
SidType=0
TaskCategory=Execute a Remote Command
OpCode=On create calls
RecordNumber=81443
Keywords=None
Message=Creating Scriptblock text (1 of 1):
{
    # Only signal success if the parent process is mshta or rundll32
    $ParentProcessID = $EventArgs.NewEvent.TargetInstance.ParentProcessId

    $ParentProcess = Get-CimInstance -ClassName 'Win32_Process' -Filter "ProcessId = $ParentProcessID"
    $ExecutableFileInfo = Get-Item -Path $ParentProcess.ExecutablePath
    $ParentProcessCommandLine = $ParentProcess.CommandLine
    $ParentProcessPath = $ParentProcess.Path

    $SpawnedProcInfo = [PSCustomObject] @{
        ProcessId = $EventArgs.NewEvent.TargetInstance.ProcessId
        ProcessCommandLine = $EventArgs.NewEvent.TargetInstance.CommandLine
        ParentProcessId = $ParentProcessID
        ParentProcessCommandLine = $ParentProcessCommandLine
        ParentPath = $ParentProcessPath
    }
}

if (@('MSHTA.EXE', 'rundll') -contains $ExecutableFileInfo.VersionInfo.InternalName) {
    # Signal that the child proc was spawned and surface the relevant info to Wait-Event
    New-Event -SourceIdentifier 'ChildProcSpawned' -MessageData $SpawnedProcInfo
}
}

ScriptBlock ID: 1e8344af-ec29-4edd-b109-6f1cd1d2bc0a
Path: C:\Users\Administrator\Documents\WindowsPowerShell\Modules\AtomicTestHarnesses\1.7.0.0\TestHarnesses\T1218.005_Mshta\InvokeHTMLApplication.ps1

```

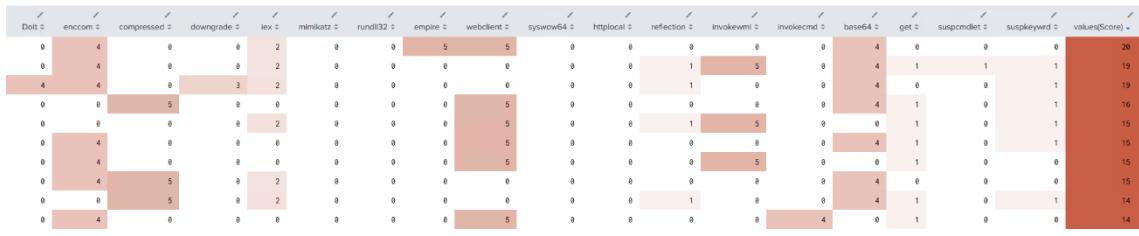
Hunting Analytic

As we began generating content, we wanted a way to evaluate the dataset created to identify keywords that would ultimately convert to new analytics. We utilized all the standard frameworks in use; Empire, Cobalt Strike, Metasploit, Atomic Red Team and AtomicTestHarnesses. Script block provides a voluminous amount of data and we didn't want to be too selective on our keywords for the analytics we wanted to produce. With this release, we're publishing a hunting analytic that will assist with combing through 4104 event data. This detection is powered by the merging of two queries (Thank you [Alex Teixeira](#)) to assist with maximizing the identification of suspicious PowerShell usage. The detection may be found on our Security Content repository [here](#).

```

`powershell` EventCode=4104
| eval DoIt = if(match(Message,"(?i)(\$doit)"), "4", 0)
| eval enccom;if(match(Message,"[A-Za-z0-9+\\/]{44}|([A-Za-z0-9+\\/]{4}|[A-Za-z0-9+\\/]{3}=|[A-Za-z0-9+\\/]{2}==)") OR
match(Message, "(?i)[-]e[nc*o*d*e*d*c*x*m*m*x*a*n*d*)*s+[^\"]"),4,0)
| eval suspcmdlet;if(match(Message, "(?i)Add-Exfiltration|Add-Persistence|Add-RegBackdoor|Add-
ScrnlSaveBackdoor|Check-VM|Do-Exfiltration|Enabled-DuplicateToken|Exploit-Jboss|Find-Fruit|Find-GPOLocation|Find-
TrustedDocuments|Get-ApplicationHost|Get-ChromeDump|Get-ClipboardContents|Get-FoxDump|Get-GPPassword|Get-
IndexedItem|Get-Keystrokes|LSASecret|Get-PassHash|Get-RegAlwaysInstallElevated|Get-RegAutoLogon|Get-RickAstley|Get-
Screenshot|Get-SecurityPackages|Get-ServiceFilePermission|Get-ServicePermission|Get-ServiceUnquoted|Get-
SiteListPassword|Get-System|Get-TimedScreenshot|Get-UnattendedInstallFile|Get-Unconstrained|Get-VaultCredential|Get-
VulnAutoRun|Get-VulnSchTask|Gupt-Backdoor|HTTP-Login|Install-SSP|Install-ServiceBinary|Invoke-ACLScanner|Invoke-
ADSBBackdoor|Invoke-ARPScan|Invoke-AllChecks|Invoke-BackdoorLNK|Invoke-BypassUAC|Invoke-CredentialInjection|Invoke-
DCSync|Invoke-DllInjection|Invoke-DowngradeAccount|Invoke-EgressCheck|Invoke-Inveigh|Invoke-InveighRelay|Invoke-
Mimikittenz|Invoke-NetRipper|Invoke-NinjaCopy|Invoke-PSInject|Invoke-Paranoia|Invoke-PortScan|Invoke-PoshRat|Invoke-
PostExfil|Invoke-PowerDump|Invoke-PowerShellTCP|Invoke-PsExec|Invoke-PsUaCme|Invoke-ReflectivePEInjection|Invoke-
ReverseDNSLookup|Invoke-RunAs|Invoke-SMBScanner|Invoke-SSHCommand|Invoke-Service|Invoke-Shellcode|Invoke-
Tater|Invoke-ThunderStruck|Invoke-Token|Invoke-UserHunter|Invoke-VoiceTroll|Invoke-WScriptBypassUAC|Invoke-
WinEnum|MailRaider|New-HoneyHash|Out-Minidump|Port-Scan|PowerBreach|PowerUp|PowerView|Remove-Update|Set-
MacAttribute|Set-Wallpaper|Show-TargetScreen|Start-CaptureServer|VolumeShadowCopyTools|NEEEEWWW|
(Computer|User)Property|CachedRDPConnection|get-netS+|invoke-\$+hunter|Install-Service|get-\$+
(credential|password)|remoteps|Kerberos.*|(policy|ticket)|netfirewall|Uninstall-
Windows|Verb\+s+Runas|AmsiBypass|nishang|Invoke-
Interceptor|EXEonRemote|NetworkRelay|PowerShellUpd|PowerShellIcmp|CreateShortcut|copy-vss|invoke-dll|invoke-
mass|out-shortcut|Invoke-ShellCommand"),1,0)
| eval base64 = if(match(lower(Message),"frombase64"), "4", 0)
| eval empire;if(match(lower(Message),"system.net.webclient") AND match(lower(Message), "frombase64string") ,5,0)
| eval mimikatz;if(match(lower(Message),"mimikatz") OR match(lower(Message), "-dumpcr") OR match(lower(Message),
"SEKURLSA::Pth") OR match(lower(Message), "kerberos::ptt") OR match(lower(Message), "kerberos::golden") ,5,0)
| eval iex = if(match(lower(Message),"iex"), "2", 0)
| eval webclient;if(match(lower(Message),"http") OR match(lower(Message),"web(client|request)") OR
match(lower(Message),"socket") OR match(lower(Message),"download(file|string)") OR
match(lower(Message),"bitransfer") OR match(lower(Message),"internetexplorer.application") OR
match(lower(Message),"xmlhttp"),5,0)
| eval get = if(match(lower(Message),"get-"), "1", 0)
| eval rundll32 = if(match(lower(Message),"rundll32"), "4", 0)
| eval suspcmdlet;if(match(Message, "(?i")
(bitransfer|mimik|metasp|AssemblyBuilderAccess|Reflection\Assembly|shellcode|injection|cnvert|shell\.application|
start-
process|Rc4ByteStream\System\.Security\.Cryptography\lsass\.exe|localAdmin|LastLoggedOn|hijack|BackupPrivilege|ngrok
|comsvcs|backdoor|brute|force|Port.2Scan|Exfiltration|exploit|DisableRealtimeMonitoring|beacon"),1,0)
| eval syswow64 = if(match(lower(Message),"syswow64"), "3", 0)
| eval httplocal = if(match(lower(Message),"http://127.0.0.1"), "4", 0)
| eval reflection = if(match(lower(Message),"reflection"), "1", 0)
| eval invokewmi;if(match(lower(Message), "(?i)(wmiobject|WMIMethod|RemoteWMI|PowerShellWmi|wmicommand") ),5,0)
| eval downgrade;if(match(Message, "(?i)([-]ve*r*s*i*o*n*\s+2)") OR match(lower(Message),"powershell -
version"),3,0)
| eval compressed;if(match(Message, "(?i)GZipStream|::Decompress|IO.Compression|write-zip|(expand|compress)-
Archive"),5,0)
| eval invokecmd = if(match(lower(Message),"invoke-command"), "4", 0)
| addtotals fieldname=Score DoIt, enccom, suspcmdlet, suspcmdlet, compressed, downgrade, mimikatz, iex, empire,
rundll32, webclient, syswow64, httplocal, reflection, invokewmi, invokecmd, base64, get
| stats values(Score) by DoIt, enccom, compressed, downgrade, iex, mimikatz, rundll32, empire, webclient,
syswow64, httplocal, reflection, invokewmi, invokecmd, base64, get, suspcmdlet, suspcmdlet
| `powershell_4104_hunting_filter`
```

As we played with the data more and more, we have found that it's even more useful by adding scores to each keyword. Keywords in this case are each `eval`. In this instance, the scoring is based on fidelity. \$Dolt is a function that Cobalt Strike uses, therefore the score is set to 4. Keywords like IEX are more commonly used and I've set the score to 2. An example of the scoring used in the following capture showcases how the scores can help bring up interesting PowerShell scripts. It is also easy enough to copy and paste an eval statement and add new keywords. Our example is not exhaustive, but a starting point for defenders to begin digging deeper.



Detections

Following our research effort, we were able to compile a good amount of new analytics. We hope this inspires others to contribute (via GitHub [Issues](#) or [PR](#)) to continue to enhance coverage for the community.

| Analytic | Technique | Tactic | Notes |
|--|------------------|--|---|
| Detect Empire with PowerShell Script Block Logging | T1059.001 | Execution | Identifies two values that are always found in the default PowerShell-Empire payloads. |
| Detect Mimikatz With PowerShell Script Block Logging | T1059.001 | Execution | Identifies strings typically found in PowerShell script block code related to mimikatz. |
| Powershell Fileless Process Injection via GetProcAddress | T1059.001, T1055 | Execution , Defense , Evasion , Privilege Escalation | Identifies the use of GetProcAddress within the script block. |
| Powershell Fileless Script Contains Base64 Encoded Content | T1059.001, T1027 | Execution | Identifies the use of Base64 within the script block. |
| Unloading AMSI via Reflection | T1562 | Defense , Evasion | Identifies system.management.automation.amsi within the script block, typically found in encoded commands disabling AMSI. |

| | | | |
|---|-----------|--|---|
| <u>PowerShell Domain Enumeration</u> | T1059.001 | <u>Execution</u> | Identifies commands typically found with domain and trust enumeration. |
| <u>PowerShell Loading .NET into Memory via System.Reflection.Assembly</u> | T1059.001 | <u>Execution</u> | Identifies system.reflection.assembly within the script block being used, typically found in malicious PowerShell script execution. |
| <u>Powershell Creating Thread Mutex</u> | T1027.005 | <u>Defense Evasion</u> | Identifies the `mutex` function typically found and used in malicious PowerShell script execution. |
| <u>Powershell Processing Stream Of Data</u> | T1059.001 | <u>Execution</u> | Identifies suspicious PowerShell script execution via EventCode 4104 that is processing compressed stream data. |
| <u>Powershell Using memory As Backing Store</u> | T1140 | <u>Defense Evasion</u> | Identifies within the script block the use of memory stream as new object backstore. |
| <u>Recon AVProduct Through Pwh or WMI</u> | T1592 | <u>Reconnaissance</u> | Identifies suspicious PowerShell script execution performing checks to identify anti-virus products installed on the endpoint. |
| <u>Recon Using WMI Class</u> | T1592 | <u>Reconnaissance</u> | Identifies suspicious PowerShell where WMI is performing an event query looking for running processes or running services. |

| | | | |
|--|-----------|--|---|
| <u>WMI Recon</u> <u>Running Process or Services</u> | T1592 | <u>Reconnaissance</u> | Identifies suspicious PowerShell script execution where WMI is performing an event query looking for running processes or running services. |
| <u>Allow Inbound Traffic In Firewall Rule</u> | T1021.001 | <u>Lateral Movement</u> | Identifies suspicious PowerShell commands to allow inbound traffic inbound to a specific local port within the public profile. |
| <u>Mailsniper Invoke functions</u> | T1114.001 | <u>Collection</u> | Identifies known mailsniper.ps1 functions executed on an endpoint. |
| <u>Delete ShadowCopy With PowerShell</u> | T1490 | <u>Impact</u> | Identifies PowerShell commands to delete shadow copy using the WMIC PowerShell module. |
| <u>Powershell Enable SMB1Protocol Feature</u> | T1027.005 | <u>Defense Evasion</u> | Identifies enabling of smb1 protocol through PowerShell Script Block logging. |
| <u>Detect WMI Event Subscription Persistence</u> | T1546.003 | <u>Privilege Escalation, Persistence</u> | Identifies WMI Event Subscription to establish persistence or perform privilege escalation. |

How to Enable It?

There are three effective ways to enable PowerShell Logging. Depending upon the deployment method or if needing to deploy across a large fleet, the registry or Group Policy will be the best bet. If testing in a lab setting, all three methods following will help.

Registry

This method may be useful if using a deployment or logon script.

- Enable ScriptBlock Logging
 - HKLM\SOFTWARE\Wow6432Node\Policies\Microsoft\Windows\PowerShell\ScriptBlockLogging
 - EnableScriptBlockLogging = 1
- Enable Module Logging
 - HKLM\SOFTWARE\Wow6432Node\Policies\Microsoft\Windows\PowerShell\ModuleLogging
- EnableModuleLogging = 1
 - HKLM\SOFTWARE\Wow6432Node\Policies\Microsoft\Windows\PowerShell\ModuleLogging \ModuleNames
 - * = *
- Transcription
 - HKLM\SOFTWARE\Wow6432Node\Policies\Microsoft\Windows\PowerShell\Transcription\ul> - EnableInvocationHeader = 1
 - EnableTranscripting = 1
 - OutputDirectory = <path_to_directory>

The PowerShell Operational Log may be found here:

%SystemRoot%\system32\winevt\logs\Microsoft-Windows-PowerShell%4Operational.evtx

PowerShell

In any case, Hurricane Labs references a [script](#) written by [Tim Ip](#) that we have borrowed and expanded on. We enhanced it with the following abilities:

- Enable one or all PowerShell logging methods
- Create a new inputs.conf to capture transcript logs and PowerShell Operational logs
- Disable all logging
- Enables Process Creation with Command-Line (4688)

Get Invoke-SPLPowerShellAuditLogging [here](#).

Update a currently used Windows inputs.conf on the Splunk Universal Forwarder or use Invoke-SPLPowerShellAuditLogging to create the inputs.

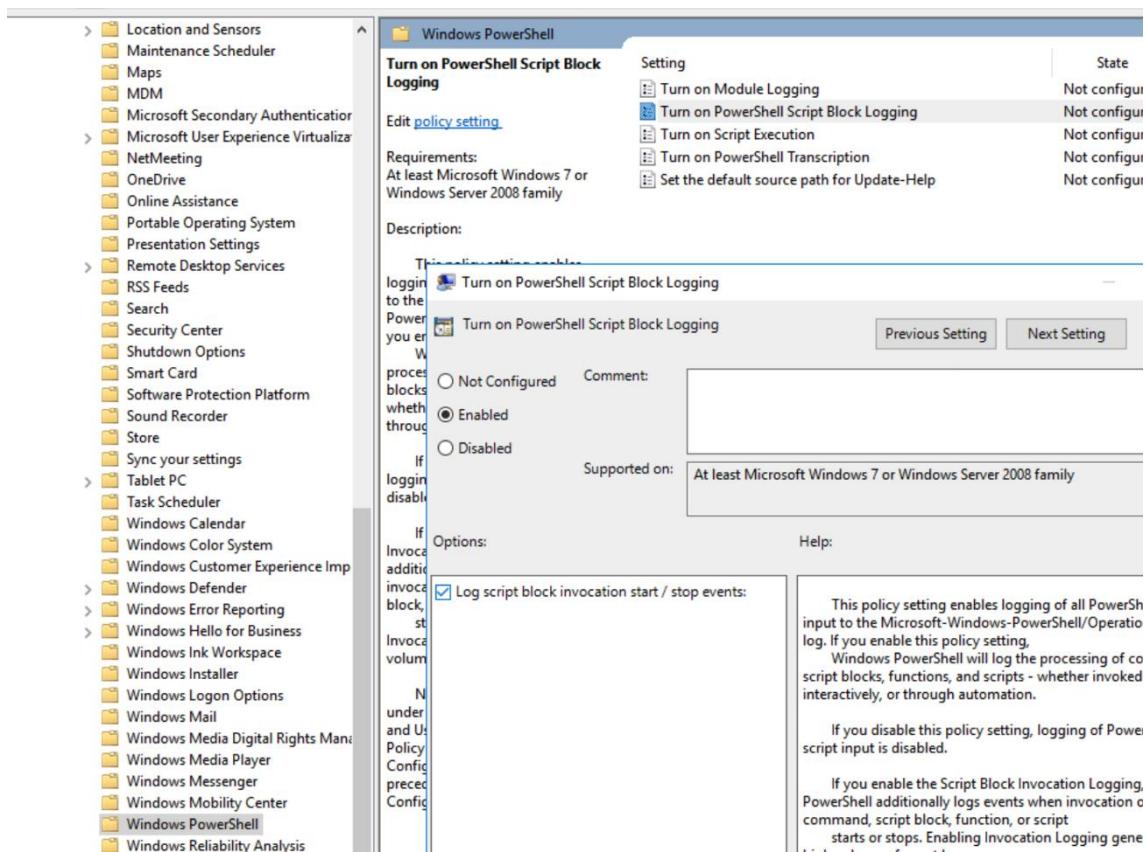
```
[WinEventLog://Microsoft-Windows-PowerShell/Operational]
source = XmlWinEventLog:Microsoft-Windows-PowerShell/Operational
renderXml = 0
disabled = false
index = win
[monitor://c:\pstransactions\]
sourcetype = powershell:transcript
disabled = false
multiline_event_extra_waittime = true
time_before_close = 300
```

```
index = win  
`Invoke-SPLPowerShellAuditLogging -method CreateInputs`
```

Enable Logging via Group Policy Objects

For a more enterprise and granular policy deployment approach, within the Group Policy Management Console, create a new or modify an existing object, browse to Computer Configuration > Policies > Administrative Templates > Windows Components > Windows PowerShell

From here, enable the policies of interest and begin logging. Deploy to critical assets or all as needed.



This work was inspired by many others who have written about PowerShell Logging, but not limited to:

- [Get Data into Splunk User Behavior Analytics](#) - PowerShell logging
- [Blueteam Powershell Recommendations](#)
- [PowerShell ♥ the Blue Team](#) - Microsoft
- [about_Logging](#) - Microsoft Docs
- [Greater Visibility Through PowerShell Logging](#) - FireEye
- [How to Use PowerShell Transcription Logs in Splunk](#) - Hurricane Labs
- [Hurricane Labs Add-on for Windows PowerShell Transcript](#)
- [How to detect suspicious PowerShell activity with Splunk?](#) - [Alex Teixeira](#)

Test Yourself

Atomic Red Team: Using Atomic Red Team, we can simulate PowerShell commands simply using [Invoke-AtomicRedTeam](#). To begin, check out the [Wiki](#) and follow along.

In a lab setting, or authorized device, run the following commands to get started:

```
IEX (IWR 'https://raw.githubusercontent.com/redcanaryco/Invoke-AtomicRedTeam/master/install-atomicredteam.ps1' -UseBasicParsing);
Install-AtomicRedTeam -getAtomicals -force
```

This will install Invoke-AtomicRedTeam. From here, we may now run [T1059.001](#) from Atomic Red Team.

```
invoke-AtomicTest T1059.001
```

Want some more data? Check out [AtomicTestHarnesses](#).

```
Out-ATHPowerShellCommandLineParameter -GenerateAllParamVariations -UseEncodedCommandParam -Execute
```

```
PS C:\Users\Administrator> Out-ATHPowerShellCommandLineParameter -GenerateAllParamVariations -UseEncodedCommandParam -Execute
TechniqueID : T1059.001
TestSuccess : True
TestGuid : 067718a0-9fb8-46ec-e6ae34808bf2
ProcessId : 3516
Commandline : powershell.exe -NoProfile -EC VwByAGkAdAB1AC0ASABvAHMAdAAgADAAnG3ADcAMQ4AGEAMAAtADkAzgB1ADgALQA0ADYAZQbjAC040AAwAGIANgAtAGUAnghAGUAMwA0ADgAMAA4AGIAZgAyAA==
TechniqueID : T1059.001
TestSuccess : True
TestGuid : cadcd251-8832-4fec-8589-f97f11cb16d5
ProcessId : 4400
Commandline : powershell.exe -NoProfile -E VwByAGkAdAB1AC0ASABvAHMAdAAgAGIAQyQAOAQmAZAAyADUAMQAtADgAOAAzADIALQA0AGYAZQbjAC040AA1ADgAOQQAtAGYAOQ3AGYAMQxAGMAYgAxADYZAA1AA==
TechniqueID : T1059.001
TestSuccess : True
TestGuid : 22ca9276-1410-4d41-ad01-1a1fc1b70043
ProcessId : 5716
Commandline : powershell.exe -NoProfile -En VwByAGkAdAB1AC0ASABvAHMAdAAgADIAgBjAGEAOQyADcAnqAtADEANAAxADAALQA0AQQAANAAxAC0A1YQb1-ADAAMQAtADEAYQAxAGYAYwAxAGTA1NwAwADAANAAzAA==
TechniqueID : T1059.001
TestSuccess : True
TestGuid : 3f6e64e-3fcfa-4730-964a-131850889276
ProcessId : 3308
Commandline : powershell.exe -NoProfile -Enc VwByAGkAdAB1AC0ASABvAHMAdAAgAdcAZAAwAdgAyW2ADQAZQAtADMZgBjAGEALQA0ADcAMwAwAC0AQ2ADQAYQAtADEAMwAxADgANQAwADgAOAA5ADIA1NwA2AA==
TechniqueID : T1059.001
TestSuccess : True
TestGuid : a25fba0b-62a3-4f09-993d-5e4cc7e0ba88
ProcessId : 6544
Commandline : powershell.exe -NoProfile -Enco VwByAGkAdAB1AC0ASABvAHMAdAAgAGEAMgA1AGIAZgBhADAAYgAtADYAMgBhADMALQA0AGYMA5AC0AQQA5ADIAZAAAtADUAZQAOAGMAYwA3AGUAMAB1AGEAOAA4AA==
```

https://www.splunk.com/en_us/blog/security/hunting-for-malicious-powershell-using-script-block-logging.html

<https://research.splunk.com/endpoint/c4db14d9-7909-48b4-a054-aa14d89dbb19/>

https://research.splunk.com/stories/malicious_powershell/

Active Directory Kerberos Attacks Analytic Story

This section describes common Kerberos attacks for which we wrote detections in the new analytic story. We are using ATT&CK Tactics to organize them. Note that this is a work in progress and does not cover all the existing Kerberos attack techniques. Feedback is [welcome!](#)

Discovery - TA0007

User Enumeration

Adversaries may abuse Kerberos to validate if a list of users is a domain user or not. This validation can be stealthy as it does not actually generate failed authentication or lockout events. This can be accomplished by submitting a TGT request with no pre-authentication. If the KDC prompts for authentication, the user is valid.

| Name | Technique ID | Tactic | Description |
|---|--------------|-----------|--|
| Kerberos User Enumeration | T1589.002 | Discovery | This analytic leverages Event Id 4768. A Kerberos authentication ticket (TGT) was requested to identify one source endpoint trying to obtain an unusual number of Kerberos TGT tickets for non-existing users. This behavior could represent an adversary abusing the Kerberos protocol to perform a user enumeration attack against an Active Directory environment. When Kerberos is sent a TGT request with no preauthentication for an invalid username, it responds with KRB5KDC_ERR_C_PRINCIPAL_UNK_NOWN or 0x6. |

| _time | Client_Address | unique_accounts | tried_accounts | comp_avg | comp_std | isOutlier |
|---------------------|----------------|-----------------|---|----------|----------|-----------|
| 2022-05-04 18:56:00 | 10.0.1.16 | 200 | 1887396168SA_123 2434161468SA_123 3014242035A_123 32533709185A_123 36848010885A_123 45180789895A_123 4692447125A_123 4854911015A_123 53771507185A_123 54797546985A_123 58880874155A_123 65118623275A_123 69421642225A_123 7323216435A_123 77864309495A_123 85863135785A_123 868453399655A_123 89991822445A_123 89418861435A_123 93389564285A_123 | 200 | 0 | 1 |

Kerberos Delegation

Kerberos delegation is an impersonation capability that enables an application to access or consume resources hosted on a different server on behalf of users. While convenient, this Kerberos feature

introduces new attack vectors that allow adversaries to abuse accounts or computers trusted for the delegation intending to steal Kerberos Ticket Granting Tickets or obtain unauthorized Kerberos Service Tickets.

| Name | Technique ID | Tactic | Description |
|---|--------------|-----------|---|
| Windows PowerView Unconstrained Delegation Discovery | T1018 | Discovery | This analytic utilizes PowerShell Script Block Logging (EventCode=4104) to identify commandlets used by the PowerView hacking tool leveraged to discover Windows endpoints with Kerberos Unconstrained Delegation. |
| Windows Get-ADComputer Unconstrained Delegation Discovery | T1018 | Discovery | This analytic utilizes PowerShell Script Block Logging (EventCode=4104) to identify the Get-ADComputer commandlet used with specific parameters to discover Windows endpoints with Kerberos Unconstrained Delegation. |
| Windows PowerView Constrained Delegation Discovery | T1018 | Discovery | This analytic utilizes PowerShell Script Block Logging (EventCode=4104) to identify commandlets used by the PowerView hacking tool leveraged to discover Windows endpoints with Kerberos Constrained Delegation. |

| EventCode | Message | ComputerName | User | count | firstTime |
|-----------|---|--------------------------------|----------------|-------|---------------------|
| 4104 | Creating Scriptblock text (1 of 1): Get-ADComputer -Filter {TrustedForDelegation -eq \$true } ScriptBlock ID: 7ca82bda-9bcb-45e4-a6bd-6758c0828067 Path: | win-host-987.attackrange.local | NOT_TRANSLATED | 1 | 2022-05-04T19:00:45 |

Credential Access - TA0006

AS-REP Roasting

Pre-Authentication is a Kerberos security feature by which users encrypt a timestamp with their secret (password) and send it to the KDC in order to request a TGT. Once the KDC validates the timestamp was encrypted with the right secret, it will issue the TGT. However, if pre-authentication is disabled, this step is skipped and adversaries are able to request a TGT for any domain user. This technique is called AS-REP roasting and it effectively allows an attacker to perform an offline brute force attack against a user's password.

| Name | Technique ID | Tactic | Description |
|--|--------------|-------------------|--|
| Disabled Kerberos Pre-Authentication Discovery With Get-ADUser | T1558.004 | Credential Access | This analytic identifies the execution of the Get-ADUser commandlet with specific parameters. Get-ADUser is part of the Active Directory PowerShell module used to manage Windows Active Directory networks. As the name suggests, Get-ADUser is used to query for domain users. With the appropriate parameters, Get-ADUser allows adversaries to discover domain accounts with Kerberos Pre Authentication disabled. |
| Disabled Kerberos Pre-Authentication Discovery With PowerView | T1558.004 | Credential Access | This analytic identifies the execution of the Get-DomainUser commandlet with specific parameters. |

| | | | |
|--|-----------|-------------------|--|
| | | | Get-DomainUser is part of PowerView, a PowerShell tool used to perform enumeration on Windows Active Directory networks. As the name suggests, Get-DomainUser is used to identify domain users and combining it with -PreauthNotRequired allows adversaries to discover domain accounts with Kerberos Pre Authentication disabled. |
| <u>Kerberos Pre-Authentication Flag Disabled in UserAccountControl</u> | T1558.004 | Credential Access | This analytic leverages Windows Security Event 4738. A user account was changed to identify a change performed on a domain user object that disables Kerberos Pre-Authentication. Disabling the Pre Authentication flag in the UserAccountControl property allows an adversary to easily perform a brute force attack against the user's password offline leveraging the ASP REP Roasting technique. |
| <u>Kerberos Pre-Authentication Flag Disabled with PowerShell</u> | T1558.004 | Credential Access | This analytic identifies the execution of the Set-ADAccountControl commandlet with specific parameters. As the name |

| | | | |
|--|--|--|--|
| | | | suggests, Set-ADAccountControl is used to modify User Account Control values for an Active Directory domain account. With the appropriate parameters, Set-ADAccountControl allows adversaries to disable Kerberos Pre-Authentication for an account. |
|--|--|--|--|

| EventCode | Message | ComputerName | User | count | firstTime |
|-----------|--|--------------------------------|----------------|-------|---------------------|
| 4184 | Creating Scriptblock text (1 of 1): Get-ADUser -Filter 'useraccountcontrol -band 4194304' -Properties useraccountcontrol ScriptBlock ID: 69d79242-fe45-49eb-866f-4dbcb80e7ef6 Path: | win-host-987.attackrange.local | NOT_TRANSLATED | 1 | 2022-05-04T19:06:55 |

Kerberoasting

In Active Directory networks, Service Principal Names (SPNs) are used to uniquely identify an instance of a network service. To enable authentication, SPNs are typically associated with a domain service account. When users request a Kerberos Service Ticket for or an SPN, part of this ticket is encrypted with the service account's password hash. This attack is known as Kerberoasting and allows adversaries to perform an offline brute force attack to attempt to obtain the service account's password.

| Name | Technique ID | Tactic | Description |
|---|--------------|-------------------|--|
| Kerberoasting spn request with RC4 encryption | T1558.003 | Credential Access | This analytic identifies a potential Kerberoasting attack against Active Directory networks. Kerberoasting allows an adversary to request Kerberos tickets for domain accounts typically used as service accounts and attempts to crack them |

| | | | |
|---|-----------|-------------------|---|
| | | | offline allowing them to obtain privileged access to the domain. This analytic looks for a specific combination of the Ticket_Options field based on common Kerberoasting tools. |
| <u>ServicePrincipalNames Discovery with PowerShell</u> | T1558.003 | Credential Access | This analytic identifies powershell.exe usage, using Script Block Logging EventCode 4104, related to querying the domain for Service Principal Names. Typically, this is a precursor activity related to Kerberoasting or the silver ticket attack. |
| <u>ServicePrincipalNames Discovery with SetSPN</u> | T1558.003 | Credential Access | This analytic identifies setspn.exe usage related to querying the domain for Service Principal Names. Typically, this is a precursor activity related to Kerberoasting or the silver ticket attack. |
| <u>Unusual Number of Kerberos Service Tickets Requested</u> | T1558.003 | Credential Access | This hunting analytic leverages Kerberos Event 4769. A Kerberos service ticket was requested to identify a potential Kerberoasting |

| | | | | |
|--|--|--|--|--|
| | | | | attack against Active Directory networks. The detection calculates the standard deviation for each host and leverages the 3-sigma statistical rule to identify an unusual number of service ticket requests. |
|--|--|--|--|--|

| _time | Client_Address | unique_services | requested_services | comp_avg | comp_std | isOutlier |
|---------------------|------------------|-----------------|--|----------|----------|-----------|
| 2022-05-04 19:10:00 | ::ffff:10.0.1.15 | 38 | srv_http1 srv_http10 srv_http2 srv_http3 srv_http4 srv_http5 srv_http6 srv_http7 srv_http8 srv_http9 srv_smb1 srv_smb10 srv_smb2 srv_smb3 srv_smb4 srv_smb5 srv_smb6 srv_smb7 srv_smb8 srv_smb9 | 38 | 8 | 1 |

Password Spraying

Password spraying is a technique by which adversaries leverage a single password or a small list of commonly used passwords against a large group of usernames to acquire valid account credentials. In an Active Directory environment, both the NTLM and Kerberos protocols can be used for this technique. Below are a few detection ideas for Kerberos. For more detection opportunities, we encourage you to visit our [Password Spraying](#) analytic story

| Name | Technique ID | Tactic | Description |
|--|--------------|-------------------|--|
| Multiple Disabled Users Failing To Authenticate From Host Using Kerberos | T1110.003 | Credential Access | Identifies one source endpoint failing to authenticate with multiple disabled domain users using the Kerberos protocol. This detection will only trigger on domain controllers, not on member servers or workstations. |

| | | | |
|--|-----------|-------------------|---|
| <u>Multiple Users Failing To Authenticate From Host Using Kerberos</u> | T1110.003 | Credential Access | Identifies one source endpoint failing to authenticate with multiple valid users using the Kerberos protocol. This detection will only trigger on domain controllers, not on member servers or workstations. |
| <u>Multiple Invalid Users Failing To Authenticate From Host Using Kerberos</u> | T1110.003 | Initial Access | Identifies one source endpoint failing to authenticate with multiple invalid domain users using the Kerberos protocol. This detection will only trigger on domain controllers, not on member servers or workstations. |

Golden Ticket

The golden ticket attack is a technique used against Active Directory environments that allows adversaries to forge an arbitrary but valid Ticket Granting Ticket (TGT) as any domain user. This effectively allows attackers to impersonate any user, including high privileged users, and perform unauthorized actions on them. A golden ticket attack can only be executed if the attacker has obtained the NTLM hash password of a special domain account, krbtgt.

| Name | Technique ID | Tactic | Description |
|---|--------------|-------------------|---|
| <u>Kerberos Service Ticket Request Using RC4 Encryption</u> | T1558.001 | Credential Access | This analytic leverages Kerberos Event 4769 to identify a potential Kerberos Service Ticket request related to a Golden Ticket attack. Adversaries who have obtained the Krbtgt account NTLM password hash may forge a Kerberos Granting Ticket (TGT) |

| | | | |
|--|--|--|---|
| | | | <p>to obtain unrestricted access to an Active Directory environment. Armed with a Golden Ticket, attackers can request service tickets to move laterally and execute code on remote systems. Looking for Kerberos Service Ticket requests using the legacy RC4 encryption mechanism could represent the second stage of a Golden Ticket attack. RC4 usage should be rare on a modern network since Windows Vista & Windows Server 2008 and newer support AES Kerberos encryption.</p> |
|--|--|--|---|

Lateral Movement - TA0008

Remote Code Execution

Once adversaries gain a foothold within an enterprise, they will seek to expand their access by leveraging techniques that facilitate lateral movement and remote code execution. Irrespective of the used technique (WMI, WinRM, SMB, etc), a lateral movement attack using the Kerberos protocol generates interesting events. For other detection ideas for lateral movement, make sure to visit our [Active Directory Lateral Movement](#) analytic story.

| Name | Technique ID | Tactic | Description |
|--|--------------|------------------|--|
| Unusual Number of Computer Service Tickets Requested | T1078 | Lateral Movement | This hunting analytic leverages Event ID 4769. A Kerberos service ticket was requested to identify an unusual number of computer service ticket requests from one source. When a |

| | | | |
|--|--|--|---|
| | | | domain-joined endpoint connects to a remote endpoint, it first will request a Kerberos Ticket with the computer name as the Service Name. An endpoint requesting a large number of computer service tickets for different endpoints could represent malicious behavior like lateral movement, malware staging, reconnaissance, etc. |
|--|--|--|---|

OverPass The Hash

Once adversaries compromise a Windows system with the highest privileges, they are able to dump credentials from memory to obtain clear text or hashed passwords. OverPass The Hash is a technique by which an attacker, who has obtained NTLM hash passwords, is able to authenticate to the Key Distribution Center using this hash and receive a valid Kerberos ticket (TGT) on behalf of the compromised user. This ticket can then be used to consume Kerberos-based services in the network.

| Name | Technique ID | Tactic | Description |
|--|--------------|------------------|---|
| <u>Unknown Process Using The Kerberos Protocol</u> | T1550 | Lateral Movement | This analytic identifies a process performing an outbound connection on port 88 used by default by the network authentication protocol Kerberos. Typically, on a regular Windows endpoint, only the lsass.exe process is the one tasked with connecting to the Kerberos Distribution Center to obtain Kerberos tickets. Identifying an unknown process using this |

| | | | |
|--|-------|------------------|--|
| | | | protocol may be evidence of an adversary abusing the Kerberos protocol. |
| Kerberos TGT Request Using RC4 Encryption | T1550 | Lateral Movement | This analytic leverages Event 4768. A Kerberos authentication ticket (TGT) was requested to identify a TGT request with encryption type 0x17, or RC4-HMAC. This encryption type is no longer utilized by newer systems and could represent evidence of an OverPass The Hash attack. Leveraging this attack, an adversary who has stolen the NTLM hash of a valid domain account can authenticate to the Kerberos Distribution Center (KDC) on behalf of the legitimate account and obtain a Kerberos TGT ticket. Depending on the privileges of the compromised account, this ticket may be used to obtain unauthorized access to systems and other network resources. |

| _time | dest | parent_process_name | process_name | process_path | process |
|---------------------|---|---------------------|--------------|---------------------|--|
| 2022-05-04 19:15:11 | ip-10-0-1-14.us-west-2.compute.internal | powershell.exe | Rubeus.exe | C:\Tools\Rubeus.exe | "C:\Tools\Rubeus.exe" asktgt /domain:attackrange.local /user:reed_schmidt /rc4:8d3faa872d8f942a7ce4db3f7e97c94d /ptt |

Pass The Ticket

Adversaries who have obtained system privileges on a Windows host are able to export the valid Kerberos Ticket Granting Tickets as well as Kerberos Service Tickets that reside in memory. In the scenario that a high-privileged account has an active session on the compromised host, tickets can be dumped and reused to consume services and resources on the network in a similar way to the [Pass The Hash](#) NTLM attack.

| Name | Technique ID | Tactic | Description |
|---|--------------|------------------|--|
| Mimikatz PassTheTicket CommandLine Parameters | T1550.003 | Lateral Movement | This analytic looks for the use of Mimikatz command line parameters leveraged to execute pass the ticket attacks. Red teams and adversaries alike may use the Pass the Ticket technique using stolen Kerberos tickets to move laterally within an environment, bypassing normal system access controls. |
| Rubeus Kerberos Ticket Exports Through Winlogon Access | T1550.003 | Lateral Movement | This analytic looks for a process accessing the winlogon.exe system process. The Splunk Threat Research team identified this behavior when using the Rubeus tool to monitor for and export Kerberos tickets from memory. Before being able to export tickets, Rubeus will try to escalate privileges to SYSTEM by obtaining a handle to winlogon.exe before trying to monitor for Kerberos tickets. Exporting tickets from memory is typically the first step of passing the ticket attacks. |
| Rubeus Command Line Parameters | T1550.003 | Lateral Movement | This analytic looks for the use of Rubeus command-line arguments utilized in common Kerberos attacks like |

| | | | |
|--|--|--|---|
| | | | exporting and importing tickets, forging silver and golden tickets, requesting a TGT or TGS, Kerberoasting, password spraying, etc. |
|  | | | |

Privilege Escalation - TA0004

SamAccountName Spoofing & Domain Controller Impersonation

On November 9, 2021, Microsoft released patches to address two vulnerabilities that affect Windows Active Directory domain controllers: sAMAccountName Spoofing (CVE-2021-42278) and Domain Controller Impersonation (CVE-2021-42287). These vulnerabilities allow an adversary with access to low-privileged domain user credentials to obtain a Kerberos Service Ticket for a Domain Controller computer account. This effectively allows a regular domain user to take control of a domain controller.

| Name | Technique ID | Tactic | Description |
|--|--------------|----------------------|---|
| Suspicious Kerberos Service Ticket Request | T1078.002 | Privilege Escalation | As part of the sAMAccountName Spoofing and Domain Controller Impersonation exploitation chain, adversaries will request and obtain a Kerberos Service Ticket (TGS) with a domain controller computer account as the Service Name. This Service Ticket can be then used to take control of the domain controller on the final part of the attack. This analytic leverages Event Id 4769, A Kerberos service ticket was requested, to identify an unusual TGS request where the Account_Name requesting the |

| | | | |
|--|-----------|----------------------|--|
| | | | ticket matches the Service_Name field. |
| <u>Suspicious Ticket Granting Ticket Request</u> | T1078.002 | Privilege Escalation | As part of the sAMAccountName Spoofing and Domain Controller Impersonation exploitation chain, adversaries will need to request a Kerberos Ticket Granting Ticket (TGT) on behalf of the newly created and renamed computer account. The TGT request will be preceded by a computer account name event. This analytic leverages Event Id 4781, The name of an account was changed and event Id 4768 A Kerberos authentication ticket (TGT) was requested to correlate a sequence of events where the new computer account on event id 4781 matches the request account on event id 4768. |

Exploitation for Privilege Escalation

In June 2021, Will Schroeder and Lee Christensen released the whitepaper "[Certified Pre-Owned: Abusing Active Directory Certificate Services](#)" which described scenarios to abuse Microsoft's PKI implementation called Active Directory Certificate Services. Combined with [PetitPotam](#), a tool that abuses native services to coerce Windows computers to authenticate to malicious endpoints, attackers can escalate their privileges in an Active Directory network. For more information, visit [this](#) analytic story.

| Name | Technique ID | Tactic | Description |
|--|--------------|-------------------|--|
| <u>PetitPotam</u> <u>Suspicious</u> | T1187 | Credential Access | This analytic identifies Event Code 4768. A Kerberos |

| | | | |
|--------------------------|--|--|--|
| Kerberos | | | authentication ticket (TGT) was requested successfully. This behavior has been identified to assist with detecting PetitPotam, CVE-2021-36942. Once an attacker obtains a computer certificate by abusing Active Directory Certificate Services in combination with PetitPotam, the next step would be to leverage the certificate for malicious purposes. One way of doing this is to request a Kerberos Ticket Granting Ticket using a tool like Rubeus. This request will generate a 4768 event with some unusual fields depending on the environment. This analytic will require tuning, we recommend filtering Account_Name to Domain Controllers for your environment. |
|--------------------------|--|--|--|

Datasets

Following the Splunk Threat Research Team's methodology to create and test the detections released in every analytic story, we simulated all the Kerberos-based attacks in a lab environment built with the [Attack Range](#) and stored the resulting telemetry in the [Attack Data](#) project.

Defenders can leverage these datasets to build or test their detections. In this section, we present a summary table containing links to the most relevant datasets. Certain attacks point to more than one dataset as we simulate the same technique in more than one way to enhance detection resilience.

| Attack | Technique ID | Tactic | Dataset Link(s) |
|--------|--------------|--------|-----------------|
|--------|--------------|--------|-----------------|

| | | | |
|---|-----------|----------------------|--|
| Kerberos User Enumeration | T1589.002 | Discovery | windows-security.log |
| Kerberos Delegation Discovery | T1018 | Discovery | windows-powershell.log windows-powershell.log windows-powershell.log windows-powershell.log |
| Kerberos Pre-Authentication Flag Disabled | T1558.004 | Credential Access | windows-security.log windows-security.log |
| Kerberos Pre-Authentication Discovery | T1558.004 | Credential Access | windows-powershell.log windows-powershell.log |
| Kerberoasting | T1558.003 | Credential Access | windows-security.log windows-security.log |
| Password Spraying | T1110.003 | Credential Access | windows-security.log windows-security.log windows-security.log |
| Golden Ticket | T1558.001 | Credential Access | windows-security.log |
| Pass The Ticket | T1078.002 | Privilege Escalation | windows-sysmon.log windows-sysmon.log |
| OverPassTheHash | T1550 | Lateral Movement | windows-security.log windows-security.log |

https://www.splunk.com/en_us/blog/security/detecting-active-directory-kerberos-attacks-threat-research-release-march-2022.html

<https://research.splunk.com/detections/>

https://research.splunk.com/stories/active_directory_kerberos_attacks/

Cyber Kill Chain

What is the Cyber Kill Chain?

The cyber kill chain is an adaptation of the military's kill chain, which is a step-by-step approach that identifies and stops enemy activity. Originally developed by Lockheed Martin in 2011, the cyber kill chain outlines the various stages of several common cyberattacks and, by extension, the points at which the information security team can prevent, detect or intercept attackers.

The cyber kill chain is intended to defend against sophisticated cyberattacks, also known as advanced persistent threats (APTs), wherein adversaries spend significant time surveilling and planning an attack. Most commonly these attacks involve a combination of malware, ransomware, Trojans, spoofing and social engineering techniques to carry out their plan.

8 Phases of the Cyber Kill Chain Process

Lockheed Martin's original cyber kill chain model contained seven sequential steps:

Phase 1: Reconnaissance

During the Reconnaissance phase, a malicious actor identifies a target and explores vulnerabilities and weaknesses that can be exploited within the network. As part of this process, the attacker may harvest login credentials or gather other information, such as email addresses, user IDs, physical locations, software applications and operating system details, all of which may be useful in phishing or spoofing attacks. Generally speaking, the more information the attacker is able to gather during the Reconnaissance phase, the more sophisticated and convincing the attack will be and, hence, the higher the likelihood of success.

Phase 2: Weaponization

During the Weaponization phase, the attacker creates an attack vector, such as remote access malware, ransomware, virus or worm that can exploit a known vulnerability. During this phase, the attacker may also set up back doors so that they can continue to access to the system if their original point of entry is identified and closed by network administrators.

Phase 3: Delivery

In the Delivery step, the intruder launches the attack. The specific steps taken will depend on the type of attack they intend to carry out. For example, the attacker may send email attachments or a malicious link to spur user activity to advance the plan. This activity may be combined with social engineering techniques to increase the effectiveness of the campaign.

Phase 4: Exploitation

In the Exploitation phase, the malicious code is executed within the victim's system.

Phase 5: Installation

Immediately following the Exploitation phase, the malware or other attack vector will be installed on the victim's system. This is a turning point in the attack lifecycle, as the threat actor has entered the system and can now assume control.

Phase 6: Command and Control

In Command & Control, the attacker is able to use the malware to assume remote control of a device or identity within the target network. In this stage, the attacker may also work to move laterally throughout the network, expanding their access and establishing more points of entry for the future.

Phase 7: Actions on Objective

In this stage, the attacker takes steps to carry out their intended goals, which may include data theft, destruction, encryption or exfiltration.

Over time, many information security experts have expanded the kill chain to include an eighth step: Monetization. In this phase, the cybercriminal focuses on deriving income from the attack, be it through some form of ransom to be paid by the victim or selling sensitive information, such as personal data or trade secrets, on the dark web.

Generally speaking, the earlier the organization can stop the threat within the cyber attack lifecycle, the less risk the organization will assume. Attacks that reach the Command and Control phase typically require far more advanced remediation efforts, including in-depth sweeps of the network and endpoints to determine the scale and depth of the attack. As such, organizations should take steps to identify and neutralize threats as early in the lifecycle as possible in order to minimize both the risk of an attack and the cost of resolving an event.

Evolution of the Cyber Kill Chain

As noted above, the cyber kill chain continues to evolve as attackers change their techniques. Since the release of the cyber kill chain model in 2011, cybercriminals have become far more sophisticated in their techniques and more brazen in their activity.

While still a helpful tool, the cyberattack lifecycle is far less predictable and clear cut today than it was a decade ago. For example, it is not uncommon for cyber attackers to skip or combine steps, particularly in the first half of the lifecycle. This gives organizations less time and opportunity to discover and neutralize threats early in the lifecycle. In addition, the prevalence of the kill chain model may give cyberattackers some indication of how organizations are structuring their defense, which could inadvertently help them avoid detection at key points within the attack lifecycle.

Critiques and Concerns Related to the Cyber Kill Chain

While the cyber kill chain is a popular and common framework from which organizations can begin to develop a cybersecurity strategy, it contains several important and potentially devastating flaws.

Perimeter Security

One of the most common critiques of the cyber kill chain model is that it focuses on perimeter security and malware prevention. This is an especially pressing concern as organizations shift away from traditional on-prem networks in favor of the cloud.

Likewise, an acceleration of the remote work trend and a proliferation of personal devices, IoT technology and even advanced applications like robotic process automation (RPA) has exponentially increased the attack surface for many enterprise organizations. This means that cybercriminals have far more points of access to exploit—and companies will have a more difficult time securing each and every endpoint.

Attack Vulnerabilities

Another potential shortcoming of the kill chain is that it is limited in terms of the types of attacks that can be detected. For example, the original framework is not able to detect insider threats, which is among the most serious risks to an organization and one of the attack types that has the highest rates of success. Attacks that leverage compromised credentials by unauthorized parties also cannot be detected within the original kill chain framework.

Web-based attacks may also go undetected by the cyber kill chain framework. Examples of such attacks include [Cross Site Scripting \(XSS\)](#), SQL Injection, DoS/DDoS and some [Zero Day Exploits](#). The massive 2017 Equifax breach, which occurred in part because of a compromised software patch, is a high-profile example of a web attack that went undetected due to insufficient security.

Finally, while the framework is intended to detect sophisticated, highly researched attacks, the cyber kill chain often misses those attackers who do not conduct significant reconnaissance. For example, those who use a

“spray and pray” technique often avoid carefully laid detection snares by pure happenstance.

Role of the Cyber Kill Chain in Cybersecurity

Despite some shortcomings, the Cyber Kill Chain plays an important role in helping organizations define their cybersecurity strategy. As part of this model, organizations must adopt services and solutions that allow them to:

- Detect attackers within each stage of the threat lifecycle with threat intelligence techniques
- Prevent access from unauthorized users
- Stop sensitive data from being shared, saved, altered, exfiltrated or encrypted by unauthorized users
- Respond to attacks in real-time
- Stop lateral movement of an attacker within the network

<https://www.crowdstrike.com/cybersecurity-101/cyber-kill-chain/>

Event Correlation

How does IT event correlation work?

IT event correlation relies on automation and software tools called event correlators, which receive a stream of monitoring and event management data automatically generated from across the managed environment. Using AI algorithms, the correlator analyzes these monitoring alerts to correlate events by consolidating them into groups, which are then compared to data about system changes and network topology to identify the cause and ideal solutions of the problems. Consequently, it's imperative to maintain strong data quality and set definitive correlation rules, particularly when supporting related tasks such as dependency mapping, service mapping and event suppression.

The entire event correlation process generally plays out in the following steps:

- **Aggregation:** Infrastructure monitoring data is collected from various devices, applications, monitoring tools and trouble ticket systems and fed to the correlator.
- **Filtering:** Events are filtered by user-defined criteria such as source, timeframe or event level. This step may alternately be performed before aggregation.

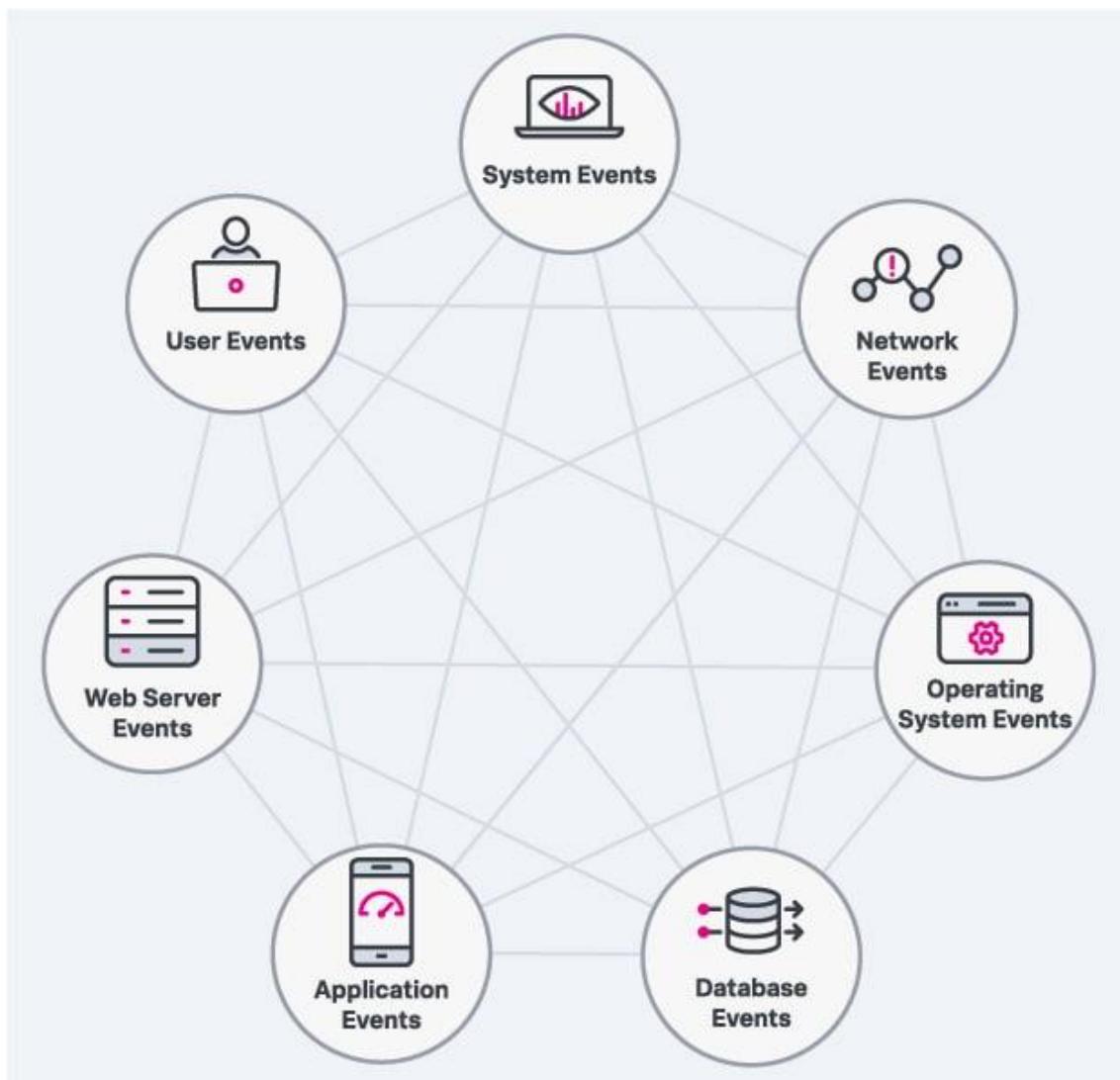
- **Deduplication:** The tool identifies duplicate events triggered by the same issue. Duplication can happen for many reasons (e.g. 100 people receive the same error message, generating 100 separate alerts). Often, there is only a single issue to address, despite multiple alerts.
- **Normalization:** Normalization converts the data to a uniform format so the event correlation tool's AI algorithm interprets it all the same way, regardless of the source.
- **Root cause analysis:** The most complex step of the process, event interdependencies are finally analyzed to determine the root cause of the event. (e.g., events on one device are examined to determine its impact on every device in the network).

Once the correlation process is complete, the original volume of events will have been reduced to a handful that require some action. In some event correlation tools, this will trigger a response such as a recommendation of further investigation, escalation or automated remediation, allowing IT administrators to better engage in troubleshooting tasks.

While many organizations correlate different types of events according to their particular IT environments and business needs, there are a few common types of event correlations:

- **System events:** These events describe anomalous changes in system resources or health. A full disk or high CPU load are both examples of system events.
- **Network events:** Network events depict the health and performance of switches, routers, ports and other components of the network, as well as network traffic if it falls out of defined thresholds.
- **Operating system events:** These events are generated by operating systems, such as Windows, Linux, Android and iOS, and describe changes in the interface between hardware and software.

- **Database events:** These events help analysts and administrators understand how database data is read, stored and updated.
- **Application events:** Generated by software applications, these events can provide insight into application performance.
- **Web server events:** These events describe activity in the hardware and software that deliver web page content.
- **User events:** These indicate infrastructure performance from the perspective of the user and are generated by synthetic monitoring or real-user monitoring systems.



IT event correlation tools make sense of various types of events that will trigger a response or action.

How are events correlated with machine learning?

Event correlation uses a variety of techniques to identify associations between event data and uncover the cause of an issue. The process is driven by machine learning algorithms that excel at identifying patterns and problem causation in massive volumes of data.

These are some of the common event correlation techniques:

- **Time-based:** This technique examines what happened immediately prior to or during an event to identify relationships in the timing and sequence of events. The user defines a time range or a latency condition for correlation.
- **Rule-based:** Rule-based correlation compares events to specific variables such as timestamp, transaction type or customer location. New rules must be written for each variable, making this approach impractical for many organizations.
- **Pattern-based:** This approach combines time- and rule-based techniques to find relationships between events that match a defined pattern. Pattern-based correlation is more efficient than a rule-based approach, but it requires an event correlation tool with integrated machine learning.
- **Topology-based:** This technique maps events to the topology of affected network devices or applications, allowing users to more easily visualize incidents in the context of their IT environment.
- **Domain-based:** A domain-based approach ingests monitoring data from individual areas of IT operations such as network performance or web applications and correlates the events. An event correlation tool may also gather data from all domains and perform cross-domain correlation.
- **History-based:** This technique allows you to learn from historical events by comparing new events to past ones to see if they match. The history-based approach is similar to pattern-based correlation, but history-based correlation can only compare

identical events, whereas pattern-based correlation has no such limitations.

| Time | Event |
|------------------------|--|
| 3/19/18 8:18:00 PM | Mon Mar 19 20:18:00 2018 Info: MID 19990410 RID [0] Response '2.6.0 <7436c832-4f04-4385-a6a1-980350db5a51> Queued mail for delivery' host = buttercup-mbpr15.sv.splunk.com source = cisco_esa.txt sourcetype = cisco:esa |
| 3/19/18 8:18:12.000 PM | Mon Mar 19 20:18:12 2018 Info: MID 19991599 Subject 'Path for the Ranbaxy cases' host = buttercup-mbpr15.sv.splunk.com source = cisco_esa.txt sourcetype = cisco:esa |
| 3/19/18 8:18:10.000 PM | Mon Mar 19 20:18:10 2018 Info: Delayed: DCID 8414159 MID 19410641 From: <eduardo.rodriguez@example.net> To: <pinkie@buttercupgames.com> <rarity@buttercupgames.com> <zecora@buttercupgames.com> RID 0 - 4.3.2 - Not accepting messages at this time ('421', ['4.3.2 try again later']) host = buttercup-mbpr15.sv.splunk.com mailto = pinkie@buttercupgames.com source = cisco_esa.txt sourcetype = cisco:esa |

Event correlation techniques rely on machine learning algorithms to identify patterns and problems from massive volumes of data

How do you identify patterns in various IT events?

You can easily find patterns and detect anomalies in IT events using an event correlation tool. After you run an initial search of your event data, an analyst can use the tool to group the results into event patterns. Because it surfaces the most common types of events, event pattern analysis is particularly helpful when a search returns a diverse range of events.

Event correlation tools usually include [anomaly detection](#) and other pattern identification functions as part of their user interface. Launching a patterns function for anomaly detection, for example, would trigger a secondary search on a subset of the current search results to analyze them for common patterns. The patterns are based on large groups of events to ensure accuracy, listed in order from most prevalent to least prevalent. An event correlation tool lets you save a pattern search as an event type and create an alert that triggers when it detects an anomaly or aberration in the pattern.

What are some of the benefits of IT event correlation?

IT event correlation has many use cases and benefits, including:

- **Cybersecurity** and **real-time malware visibility and detection**: IT teams can correlate monitoring logs from antivirus software, firewalls and other security management tools for actionable threat intelligence, which helps identify security breaches and detect threats in real time.
- **Reduced IT operational costs**: Event correlation automates necessary but time-consuming network management processes, reducing time teams spend trying to understand recurring alerts and providing more time to resolve threats and problems.
- **Greater efficiency**: Manual event correlation is laborious and time-consuming and requires expertise — factors that make it increasingly more challenging to conduct as infrastructure expands. Conversely, automated tools increase efficiency and make it easy to scale to align with your SLAs and infrastructure.
- **Easier compliance**: Event correlation facilitates continuous monitoring of all IT infrastructures and allows you to generate reports detailing security threat and regulatory compliance measures.
- **Reduced noise**: Of the thousands of network events that occur every day, some are more serious than others. Event correlation software can quickly sift through the reams of incidents and events to determine the most critical ones and elevate them as top priorities.

Essentially IT event correlation helps businesses ensure the reliability of their IT infrastructure. Any IT issue can threaten a business's ability to serve its customers and generate revenue. According to a 2020 survey, 25% of respondents worldwide reported the average hourly downtime cost of their servers was as high as US \$400,000. Event

correlation helps mitigate these downtime costs by supporting increased infrastructure reliability.

Can you protect IT infrastructure with IT event correlation monitoring?

Event correlation can support network security by analyzing a large set of event data and identifying relationships or patterns that suggest a security threat.

As an example, imagine you notice multiple login attempts in a short amount of time on an account that has been dormant for years. After successfully logging in, the account begins executing suspicious commands. With the help of event correlation, an intrusion detection system could recognize these related events as a potential cyberattack and alert the appropriate team.

An event correlation tool can map and contextualize the data it ingests from infrastructure sources to identify suspicious patterns in real time. Some event correlation tools will also produce correlation reports for [common types of attacks](#), including user account threats, database threats, Windows and Linux threats and ransomware, among others. Event correlation equips IT teams to better respond to security threats and develop stronger policies to prevent them.

What are some of the biggest challenges to efficient event correlation?

Since the dawn of enterprise computing, event correlation has been an essential practice for identifying and resolving IT problems that can have negative business impacts.

Historically, event correlation was a manageable manual process for IT teams when networks were simpler and predominantly contained on-premises. But today's dynamic network environments can produce thousands or millions of events in a single day. Keeping up with the volume of events that modern infrastructures generate, let alone parsing them into actionable information, is beyond human capabilities. Event correlation technology can perform this task more quickly and cost effectively while freeing IT teams to focus more on resolving the problems instead of detecting them.

How do you integrate IT event correlation into SIEM?

IT event correlation integrates into security information and event management ([SIEM](#)) by taking the incoming logs and correlating and normalizing them to make it easier to identify security issues in your environment. The process requires both the SIEM software and a separate event correlation engine. As such, it's important to consider how each works to understand the benefit of using them together.

Learn about [Splunk SIEM for Enterprise Security](#).

At its most basic level, SIEM collects and aggregates the log data generated throughout an organization's IT infrastructure. This data comes from network devices, servers, applications, domain controllers and other disparate sources in a variety of formats. Because of its diverse origins, there are few ways to correlate the data to detect trends or patterns, which creates obstacles to determining if an unusual event signals a security threat or just an aberration. Event correlation takes in all the logs entering your system and converts them into a consistent, readable format. Once logs are normalized, analysts can string together the clues spread among multiple types of logs and detect incidents and security events in real time. Event correlation also brings more clarity to log sources so you can recognize trends in incoming events.

How do you get started with event correlation?

To get started with event correlation, you need to find an event correlation solution that meets your organization's specific needs. Consider the following when evaluating event correlators:

- **User experience:** As with any new software, it's important to consider how easy — or difficult — it will be for users to learn, understand and use. A good event correlator will have a modern interface with intuitive navigation and a management console that integrates with your IT infrastructure. Its native analytics

should be easy to set up and understand, and it should also easily integrate with the best third-party analytics systems.

- **Features and functionality:** It's critical to know what data sources a data correlator can ingest and in what formats. It's also important to look at what types of events the tool can correlate (monitoring, observability, changes, etc.) and what steps it takes to process event data (normalization, deduplication, root cause analysis, etc.). The ability to trigger appropriate, corresponding actions (such as automated remediation) is also a desired feature.
- **Machine learning and anomaly detection capabilities:** While you don't have to be a data scientist to use an event correlator, it helps to have a basic understanding of machine learning to better inform your purchasing decision. There are essentially two types of machine learning: supervised and unsupervised.
 - Supervised machine learning uses a structured dataset that includes examples with known, specific outcomes to guide the algorithm. The algorithm is told what variables to analyze and given feedback on the accuracy of its predictions. In this way, the algorithm is "trained" using existing data to predict the outcome of new data.
 - Unsupervised machine learning, on the other hand, explores data without any reference to known outcomes. This allows it to identify previously unknown patterns in unstructured data and cluster them according to their similarities. Machine-generated data formats widely vary, ranging from structured syslog data to unstructured multi-line application data, making it essential that a correlator supports both supervised and unsupervised machine learning.

Beyond these criteria, it's also important to check that any event correlator you're considering can integrate with other tools and vendor partners you're currently working with. In addition, it should also help

you meet your business's or industry's compliance requirements, as well as offer robust customer support.

Once you've gotten started, optimize the practice with [event correlation best practices](#).

What is the future of IT event correlation?

The growing complexity of modern infrastructures and a more aggressive, sophisticated threat landscape have combined to make it more challenging than ever for IT teams to detect and resolve performance problems and security incidents. As these factors compound, event correlation will become an increasingly important tool for organizations to make certain their IT services are reliable. To that end, IT event correlation will continue to support and optimize network self-healing.

Event correlation will also need to continue to harness advances in analytics and artificial intelligence to keep pace with this dynamic environment. It will be especially significant in [AIOps](#) as organizations seek to process and analyze a flood of alerts in real time before they escalate into outages or network disruptions.

The Bottom Line: Event correlation makes sense of your infrastructure

The clues to performance issues and security threats within your environment are in your event data. But IT systems can generate terabytes' worth of data each day, making it virtually impossible to determine which events need to be acted upon and which do not. Event correlation is the key to making sense of your alerts and taking faster and more effective corrective action. It can help you better understand your IT environment and ensure it's always serving your customers and your business.

https://www.splunk.com/en_us/data-insider/it-event-correlation.html

Event Correlation Use Cases and Techniques

In essence, event correlation is a technique that relates various events to identifiable patterns. If those patterns threaten security, then an action can be imposed. Event correlation can also be performed as soon as the data is indexed. Some important use cases include:

- Data intelligence
- Operations support
- Root cause analysis
- Fraud detection

You can handle events through something as simple as sys-logging, which allows you to view new events as they arrive, but event correlation is the technique that associates varying events with one another. This is often achieved with the use of event **correlation tools** and alerting systems. Furthermore, correlating events can help security teams identify those that are most important.

Searches in SIEM with Regex

Sigma is a generic and open signature format that allows you to describe relevant log events in a straightforward manner. The rule format is very flexible, easy to write and applicable to any type of log file. The main purpose of this project is to provide a structured form in which researchers or analysts can describe their once developed detection methods and make them shareable with others.

<https://github.com/SigmaHQ/sigma>

Other Tool

<https://uncoder.io/>

How to Use Regex

The erec command

When using regular expression in Splunk, use the **erec command** to extract data from a field when you do not know the regular expression to use.

Syntax for the command:

```
| erec <thefieldname> examples="exampletext1,exampletext2"
```

Let's take a look at an example.

In this screenshot, we are in my index of CVEs. I want to have Splunk learn a new regex for extracting all of the CVE names that populate in this index, like the example CVE number that I have highlighted here:

The screenshot shows the Splunk interface with a search titled 'New Search'. The search bar contains the query 'useragent ~ "MSIE 6.0" AND user ~ "192.168.1.1"'. The results table has columns: _index, _score, _type, _id, _source, and _raw. The first few rows show log entries from 'Windows-7' hosts. The row containing 'CVE-2013-0001' is highlighted with a red box.

Figure 1 – a CVE index with an example CVE number highlighted

Next, by using the `erex` command, you can see in the job inspector that Splunk has ‘successfully learned regex’ for extracting the CVE numbers. I have sorted them into a table, to show that other CVE_Number fields were extracted:

The screenshot shows the 'Job Inspector' window with a table titled 'Fields by type: INDEXED'. The table lists various fields extracted from the logs, including 'CVE_2013_0001', 'CVE_2013_0002', 'CVE_2013_0003', 'CVE_2013_0004', 'CVE_2013_0005', 'CVE_2013_0006', 'CVE_2013_0007', and 'CVE_2013_0008'. Each row includes the field name, type, count, and summary.

Figure 2 – the job inspector window shows that Splunk has extracted CVE_Number fields

The rex Commands

When using regular expression in Splunk, use the **rex command** to either extract fields using regular expression-named groups or replace or substitute characters in a field using those expressions.

Syntax for the command:

```
| rex field=field_to_rex_from  
"FrontAnchor(?<new_field_name>{characters}+)>BackAnchor"
```

Let's take a look at an example.

This SPL allows you to extract from the field of useragent and create a new field called WebVersion:

```
index=web host=www1
| rex field=useragent "^(?<WebVersion>[\S+]*)"
```

Figure 3 – this SPL uses rex to extract from “useragent” and create “WebVersion”

As you can see, a new field of WebVersion is extracted:

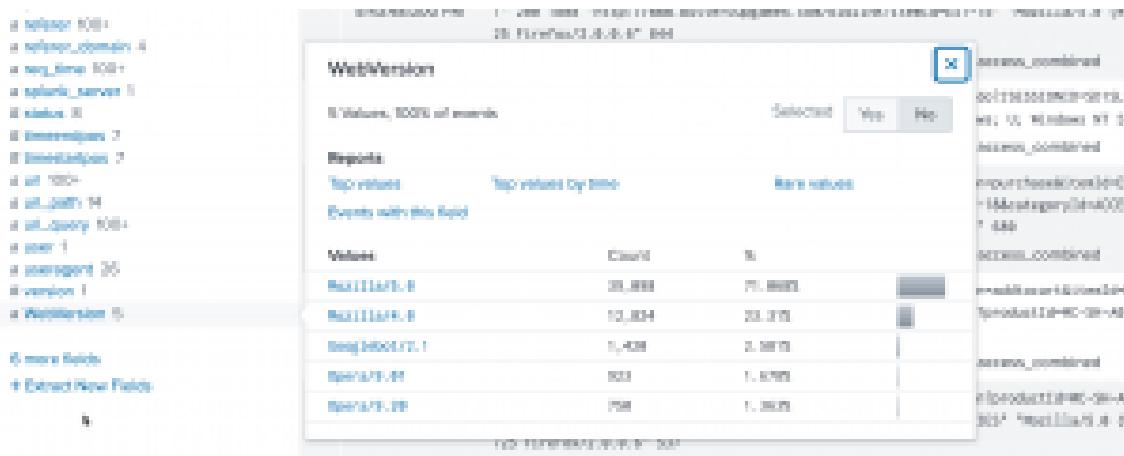
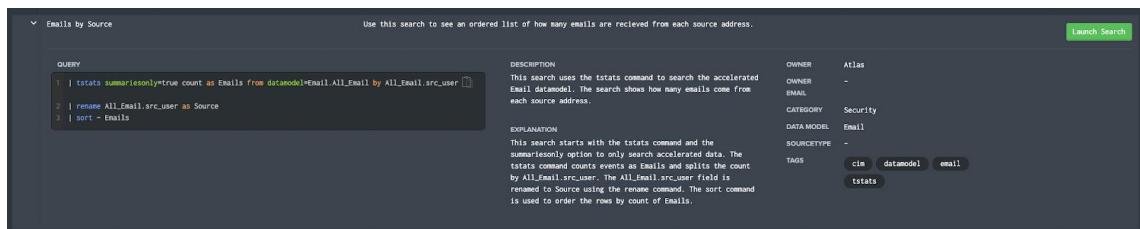


Figure 4 – the new field in WebVersion

Splunk Pro Tip: There's a super simple way to run searches simply—even with limited knowledge of SPL—using [Search Library in the Atlas app on Splunkbase](#). You'll get access to thousands of pre-configured Splunk searches developed by Splunk Experts across the globe. Simply find a search string that matches what you're looking for, copy it, and use right in your own Splunk environment. Try speeding up your regex search right now using these SPL templates, completely free.



The Basics of Regex

The Main Rules

^ = match beginning of the line

\$ = match end of the line

Regex Flags

/g = global matches (match all), don't return after first match

/m = multi-line

/gm = global and multi-line are set

/i = case insensitive

Setting Characters

\w = word character

\W = not a word character

\s = white space

\S = not white space

\d = a digit

\D = not a digit

\. = the period key

Setting Options

* = zero or more

+ = 1 or more

? = optional, zero or 1

| = acts as an “or” expression

\ = escape special characters

() = allows for character groupings, wraps the regex sets

Some Examples

\d{4} = match 4 digits in a row of a digit equal to [0-9]

\d{4,5} = match 4 digits in a row or 5 digits in a row whose values are [0-9]

[a-z] = match between a-z

[A-Z] = match between A-Z

[0-9] = match between 0-9

(t|T) = match a lowercase “t” or uppercase “T”

(t|T)he = look for the word “the” or “The”

Regex Examples

If you’re looking for a **phone number**, try out this regex setup:

$\backslash d\{10\}$ = match 10 digits in a row

OR

$\backslash d\{3\}-?\backslash d\{3\}-?\backslash d\{4\}$ = match a number that may have been written with dashes
123-456-7890

OR

$\backslash d\{3\}[-]?\backslash d\{3\}[-]?\backslash d\{4\}$ = match a phone number that may have dashes or
periods as separators

OR

$(\backslash d\{3\})[-]?(\backslash d\{3\})[-]?(\backslash d\{4\})$ = using parentheses allows for character grouping.
When you group, you can assign names to the groups and label one. For
example, you can label the first group as “area code”.

If you’re looking for a **IP address**, try out this regex setup:

$\backslash d\{1,3\}\.\backslash d\{1,3\}\.\backslash d\{1,3\}\.\backslash d\{1,3\}$ = searches for digits that are 1-3 in length,
separated by periods.

Use regex101.com to practice your RegEx:



Figure 5 – a practice search entered into regex101.com

<https://kinneygroup.com/blog/regular-expressions-in-splunk>

https://www.youtube.com/watch?app=desktop&v=GWI-TuAAF-k&ab_channel=SplunkHow-To

Regexp query [edit](#)

Returns documents that contain terms matching a [regular expression](#).

A regular expression is a way to match patterns in data using placeholder characters, called operators. For a list of operators supported by the regexp query, see [Regular expression syntax](#).

Example request [edit](#)

The following search returns documents where the `user.id` field contains any term that begins with k and ends with y. The `.*` operators match any characters of any length, including no characters. Matching terms can include `ky`, `kay`, and `kimchy`.

```
GET /_search
{
  "query": {
    "regexp": {
      "user.id": {
        "value": "k.*y",
        "flags": "ALL",
        "case_insensitive": true,
        "max_determinized_states": 10000,
        "rewrite": "constant_score"
      }
    }
  }
}
```

[Copy as curl](#) [View in Console](#)

Top-level parameters for `regexp` [edit](#)

`<field>`

(Required, object) Field you wish to search.

Parameters for `<field>` [edit](#)

`value`

(Required, string) Regular expression for terms you wish to find in the provided <field>. For a list of supported operators, see [Regular expression syntax](#).

By default, regular expressions are limited to 1,000 characters. You can change this limit using the `index.max_regex_length` setting.

The performance of the `regexp` query can vary based on the regular expression provided. To improve performance, avoid using wildcard patterns, such as `.*` or `.*?+`, without a prefix or suffix.

flags

(Optional, string) Enables optional operators for the regular expression. For valid values and more information, see [Regular expression syntax](#).

case_insensitive [7.10.0]

(Optional, Boolean) Allows case insensitive matching of the regular expression value with the indexed field values when set to true. Default is false which means the case sensitivity of matching depends on the underlying field's mapping.

max_determinized_states

(Optional, integer) Maximum number of [automaton states](#) required for the query. Default is 10000.

Elasticsearch uses [Apache Lucene](#) internally to parse regular expressions. Lucene converts each regular expression to a finite automaton containing a number of determinized states.

You can use this parameter to prevent that conversion from unintentionally consuming too many resources. You may need to increase this limit to run complex regular expressions.

rewrite

(Optional, string) Method used to rewrite the query. For valid values and more information, see the [rewrite parameter](#).

<https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl-regexp-query.html>

<https://www.mezmo.com/learn-observability/using-kibana-to-search-your-logs>

ChatGPT Regex

https://www.linkedin.com/posts/tyler-hodges-a3464355_using-chatgpt-to-create-regex-patterns-for-activity-7016061921422438400-dfQz/?originalSubdomain=py

1. Simplify SIEM Query Writing with ChatGPT

With ChatGPT, you can simply type in whatever you want in the blog.

ChatGPT makes it effortless to write SIEM queries, just type in your desired information and it will output the query for you.

As a demonstration, let's examine a Splunk example:



write a regular expression query in splunk to filter ip



Here's an example Splunk regular expression (regex) query to filter events that contain an IP address:

javascript

Copy code

```
ip=(\b(?:25[0-5]|2[0-4][0-9]|01?[0-9][0-9]?)\.){3}(?:25[0-5]|2[0-4][0-9]|01?)
```

This query returns all events that contain an IP address by matching a pattern that corresponds to the structure of an IP address (four groups of 1 to 3 digits separated by dots). The regex `(\b(?:25[0-5]|2[0-4][0-9]|01?[0-9][0-9]?)\.){3}(?:25[0-5]|2[0-4][0-9]|01?[0-9][0-9]?)` matches IP addresses in the format `xxx.xxx.xxx.xxx`, where `xxx` is a number between 0 and 255.

You can modify this query to match different types of IP addresses by changing the regex pattern. For example, you could use a simpler pattern like `/^\b\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}\b/` to match IP addresses with any value, regardless of their validity as actual IP addresses.

Another example is generating a sentinel query.



write a sentinel query to identify the number of active directory logon failures for users within 24 hour period



Here's an example of a Sentinel query to identify the number of active directory logon failures for users within a 24 hour period:

csharp

Copy code

```
let startTime = ago(24h);
let endTime = now();

AzureActivity
| where TimeGenerated >= startTime and TimeGenerated <= endTime
| where OperationName == "User Logon" and ResultDescription == "Failure"
| summarize count() by UserPrincipalName
```

2. Generate Security Detections

Using ELK (Elasticsearch, Logstash, and Kibana) to generate security detections can be done easily using ChatGPT.



write ELK query to detect registry change



Here's an example ELK (Elasticsearch, Logstash, and Kibana) query to detect registry changes thumb-up comment in Windows event logs:

```
swift
{
  "query": {
    "bool": {
      "must": [
        {
          "match": {
            "event_id": "13 vevent_id:13 AND 4740"
          }
        },
        {
          "match": {
            "registry_key_path": "HKLM\\SYSTEM\\ControlSet001\\Control\\Registry"
          }
        }
      ]
    }
  }
}
```

write a splunk query to detect command and control beaconing activity using subdomain DNS requests



write a splunk query to detect command and control beaconing activity using subdomain DNS requests.



Here's an example of a Splunk query to detect command and control (C2) beaconing activity using subdomain DNS requests:

graphql

Copy code

```
index=dns
| stats count by host, query
| search count>5 AND query="*.*"
| sort -count
```

This query starts by searching the DNS index for all events. The `stats` command is used to count the number of DNS requests per host and query. The `search` command filters the results to only show entries where the count is greater than 5 and the query contains a single asterisk followed by another asterisk (`*.*`), which is a common pattern used in C2 beaconing. Finally, the `sort` command sorts the results in descending order based on the count.

This query can be refined further to improve the accuracy of the results by adding additional

3. Write Security Policies

Writing a comprehensive security policy can be a time-consuming and complex process. There are many elements to consider, such as risk assessments, threat modeling, incident response procedures, and more. However, by leveraging the power of advanced artificial intelligence technology, such as ChatGPT, this process can be simplified and streamlined. ChatGPT can assist in generating policy content, as well as assist in reviewing and refining existing policies, to ensure that they are up-to-date, comprehensive, and effective in protecting the organization.

write a data security policy for my organization of 500 employees



write a data security policy for my organization of 500 employees



Data Security Policy for [Organization Name]

Introduction:

This policy outlines the guidelines for protecting [Organization Name]'s information and data, which is a critical business asset. The policy applies to all employees, contractors, and third-party service providers who access [Organization Name]'s information systems, including physical and electronic data.

Scope:

This policy covers all types of data, including but not limited to:

- Confidential information such as personal data, trade secrets, and financial information
- Electronic documents and records stored on [Organization Name]'s servers and systems
- Physical records and documents stored in [Organization Name]'s offices

Responsibilities:

- All employees are responsible for ensuring the security of [Organization Name]'s data and must comply with this policy.
- Employees must keep their passwords confidential and secure and must not share them with others.
- Employees must report any security incidents such as lost or stolen devices or unauthorized access to data, to the Stop generating liability.

4. Identify Vulnerabilities

Vulnerability management is a critical aspect of any organization's cybersecurity strategy. Cyber criminals are always on the lookout for vulnerabilities in systems, applications, and networks that they can exploit to gain unauthorized access. The earlier a vulnerability is discovered, the easier it is to fix it. However, finding and fixing vulnerabilities can be a tedious and time-consuming process for IT teams, especially for large organizations with multiple systems and applications to manage. ChatGPT can be a useful tool to simplify this process. With its advanced natural language processing capabilities, ChatGPT can assist IT teams in quickly identifying potential vulnerabilities and providing recommendations for remediation.

In the following example, we gave GPT code examples, and asked What's wrong with this piece of code?



what's wrong with this piece of code?



```
@app.route("/login")
def login():

    username = request.values.get('username')
    password = request.values.get('password')

    # Prepare database connection
    db = pymysql.connect("localhost")
    cursor = db.cursor()

    # Execute the vulnerable SQL query concatenating user-provided input.
    cursor.execute("SELECT * FROM users WHERE username = '%s' AND password = '%s'" %
    (username, password))

    # If the query returns any matching record, consider the current user logged in.
    record = cursor.fetchone()
    if record:
        session['logged_user'] = username

    # disconnect from server
    db.close()
```



The code you posted is vulnerable to SQL Injection attacks. The problem lies in the way the SQL query is constructed. By concatenating user-provided `username` and `password` inputs directly into the query string, the code makes it possible for an attacker to inject malicious SQL code into the query. This can res

Stop generating

being stolen, altered, or deleted,



The code you posted is vulnerable to SQL Injection attacks. The problem lies in the way the SQL query is constructed. By concatenating user-provided `username` and `password` inputs directly into the query string, the code makes it possible for an attacker to inject malicious SQL code into the query. This can result in sensitive data being stolen, altered, or deleted, and can compromise the security and stability of the entire system.



To avoid this vulnerability, it is recommended to use parameterized queries (also known as prepared statements) instead. This involves passing the parameters separately to the query, so that the database library can properly escape and sanitize the values, avoiding the risk of SQL injection.

Here's an example of a safer implementation:

```
less Copy code

@app.route("/login")
def login():
    username = request.values.get('username')
    password = request.values.get('password')

    # Prepare database connection
    db = pymysql.connect("localhost")
    cursor = db.cursor()

    # Use parameterized query to avoid SQL injection
    query = "SELECT * FROM users WHERE username = %s AND password = %s"
    cursor.execute(query, (username, password))

    # If the query returns any
    if cursor.fetchone():
        # Regenerate response under the current user logged in
        # ...
    else:
        # Regenerate response under the guest user
        # ...


```

5. Write Automation Scripts for Daily Operations

Automation script is useful but writing automation scripts right takes time. With ChatGPT's advanced capabilities, it can quickly generate code based on your requirements and reduce the amount of time you need to spend on writing scripts.

Generate a Powershell script for Malware scanning on a list of computers.



Generate a PowerShell Script for Malware Scanning on a list of Computers.



Here's an example of a PowerShell script to scan a list of computers for malware using Windows Defender:



```
php

# Import the required module
Import-Module Defender

# Define the list of computers to scan
$computers = @("computer1", "computer2", "computer3")

# Loop through the list of computers
foreach ($computer in $computers) {
    Write-Host "Scanning $computer..."
    # Use the Start-MpScan cmdlet to start a scan on the remote computer
    Start-MpScan -ScanType QuickScan -ScanPath "\\$computer\C$"
    # Wait for the scan to finish
    do {
        Start-Sleep -Seconds 5
        $scanStatus = Get-MpScanResult -ComputerName $computer
    } while ($scanStatus.ScanStatus -ne "Completed")
    # Check the results of the scan
    if ($scanStatus.Infected) {
        Write-Host "Malware detected on $computer"
```

Generate a bash script that automates port scanning with Nmap



generate a bash script that automates port scanning with Nmap



Here's an example bash script that automates port scanning with Nmap:



```
bash
Copy code

#!/bin/bash

echo "Enter target IP or hostname: "
read target

echo "Enter port range (e.g. 1-1000): "
read port_range

echo "Running Nmap scan..."
nmap -p $port_range $target > nmap_output.txt

echo "Scan complete. Results saved in nmap_output.txt."
```

This script will prompt the user for the target IP or hostname and a port range, then run an Nmap scan and save the output to a file named "nmap_output.txt".

6. Assist Incident Response

The complexity of security alerts can often lead to confusion and a feeling of being overwhelmed. However, with the help of GPT, this problem can be greatly reduced. By using GPT, you can receive suggestions on how to handle security investigations, threat hunting, or summarizing security issues. These suggestions are based on the vast knowledge and information that GPT has been trained on, making it a valuable asset for anyone dealing with security alerts. Whether you are a security analyst, engineer, or administrator, having the ability to quickly and effectively respond to security incidents can greatly improve your overall security posture. So, incorporating ChatGPT into your security toolkit can not only simplify the process but also provide you with a more comprehensive solution to address security alerts.

For example, here is a suspicious alert

```
    "alert_time": "2022-01-03 17:31:00",
    "alert_type": "ENDPOINT",
    "analyst_rating": "0",
    "asset_coverage_score": "6",
    "asset_endpoint_id": "af78afdf-b58b-4fcc-8d46-ab6702ee3db4",
    "asset_id": "370",
    "asset_importance_score": "6",
    "asset_name": "HQ-SALES-LAP101",
    "asset_os": "Windows 2016",
    "asset_risk_score": "10",
    "asset_role": "General Server",
    "asset_service": "Shared Folder,Remote Desktop,RPC",
    "asset_severity_score": "8",
    "asset_subnet_id": "355",
    "asset_type": "Endpoint",
    "client_ip": "::",
    "client_port": "0",
    "conclusion_id": "351337",
    "conclusion_status": "NEW",
    "dest_ip": "::",
    "dest_port": "0",
    "endpoint_activity": "Detected",
    "endpoint_agent_ip": "192.168.13.101",
    "endpoint_alert_id": "9678758",
    "entropy": "7.087",
    "filename": "nslookup.exe",
    "filesize": "86528",
    "filetype": "exe",
    "fss_uuid": "3609144a-6cc3-11ec-a840-0050569a5072",
    "group_name": "default",
    "has_endpoint_agent": "Yes",
    "host_ip": "192.168.13.101",
    "insert_time": "2022-01-03 18:30:22",
    "label": "Detection Rules,Endpoint Alerts",
    "md5": "af1787f1dbe0053d74fc687e7233f8ce",
    "mitre_tactic": "Discovery",
    "mitre_tactic_id": "TA0007",
    "mitre_technique": "Remote System Discovery",
    "mitre_technique_id": "T1018",
    "priority": "2",
    "rule_name": "Endpoint: Process: T1018 - Remote System Discovery",
    "server_ip": "::",
    "server_port": "0",
```

You can ask ChatGPT to provide suggestions.

Can you write a report on this and suggest additional evidence to collect with details?



This security alert is malicious. Can you write a report on this and suggest additional evidence to collect with details? {



```
"alert_id": "14517821",
"alert_threat_score": "40",
"alert_time": "2022-01-03 17:31:08",
"alert_type": "ENDPOINT",
"analyst_rating": "0",
"asset_coverage_score": "6",
"asset_endpoint_id": "af78afdf-b58b-4fcc-8d46-ab6702ee3db4",
"asset_id": "370",
"asset_importance_score": "6",
"asset_name": "HQ-SALES-LAP101",
"asset_os": "Windows 2016",
"asset_risk_score": "10",
"asset_role": "General Server",
"asset_service": "Shared Folder,Remote Desktop,RPC",
"asset_severity_score": "8",
"asset_subnet_id": "355",
"asset_type": "Endpoint",
"client_ip": "::",
"client_port": "0",
"conclusion_id": "351337",
"conclusion_status": "NEW",
"dest_ip": "::",
"dest_port": "0",
"endpoint_activity": "Detected",
"endpoint_agent_ip": "192.168.13.101",
"endpoint_alert_id": "9678758",
```

```
src_ip : ::,  
"src_port": "0",  
"summary": "Endpoint alert on hq-sales-lap101.hq.internal: Artifact: nslookup.exe,  
Adversaries will likely attempt to get a listing of other systems by IP address, hostname, or  
other logical identifier on a network that may be used for Lateral Movement from the  
current system. Functionality could exist within r",  
"uuid": "3609144a-6cc3-11ec-a840-0050569a5072",  
"weighted_severity": "5",  
"with_malware": "No"  
}
```



Security Alert Report:



Malicious Activity Detected on Endpoint "HQ-SALES-LAP101"

I. Overview:

A security alert was triggered on endpoint "HQ-SALES-LAP101" running "Windows 2016" operating system. The alert was generated by the endpoint agent with IP address "192.168.13.101". The alert was classified as "Medium" severity with a threat score of "40". The endpoint activity was detected as "Remote System Discovery".

II. Threat Details:

The malicious activity was detected as a result of a process execution of "nslookup.exe" with file size "86528" and MD5 hash "af1787f1dbe0053d74fc687e7233f8ce". The process execution was categorized as "Remote System Discovery" technique (MITRE ID T1018) under the "Discovery" tactic (MITRE ID TA0007).

"192.168.13.101". The alert was classified as "Medium" severity with a threat score of "40".
The endpoint activity was detected as "Remote System Discovery".

II. Threat Details:

The malicious activity was detected as a result of a process execution of "nslookup.exe" with file size "86528" and MD5 hash "af1787f1dbe0053d74fc687e7233f8ce". The process execution was categorized as "Remote System Discovery" technique (MITRE ID T1018) under the "Discovery" tactic (MITRE ID TA0007).

III. Recommendations:

1. Further investigation is required to determine the source and nature of the malicious activity.
2. Collect additional evidence such as network traffic logs, process execution logs, and endpoint process memory dumps.
3. Implement network segmentation and access controls to limit the spread of the malicious activity.
4. Conduct a thorough review of security configurations and ensure endpoint protection measures are in place and updated.

IV. Conclusion:

A malicious activity was detected on endpoint "HQ-SALES-LAP101" running "Windows 2016" operating system. The activity was categorized as "Remote System Discovery" and classified as "Medium" severity. Further investigation is recommended to understand the scope and impact of the attack.

And take a step further, you can ask it to automate the response for you.

Generate the script to Get Malicious IPs from SIEM and Block it on my Firewall



Generate a script to Get Malicious IPs form SIEM and Block it on my Firewall



Assuming you have a firewall and SIEM (Security Information and Event Management) system setup, the following script can be used to retrieve malicious IPs from the SIEM and block them on your firewall.



Script in Python:

```
python

import requests

# SIEM URL to retrieve malicious IPs
SIEM_URL = "https://<SIEM_HOST>/malicious_ips"

# Firewall API endpoint to block IPs
FIREWALL_URL = "https://<FIREWALL_HOST>/block_ip"

# headers for the SIEM and firewall API calls
HEADERS = {'Content-Type': 'application/json'}

# retrieve malicious IPs from the SIEM
response = requests.get(SIEM_URL, headers=HEADERS)
malicious_ips = response.json()

# loop through the list of malicious IPs and block them on the firewall
```

Copy code

<https://www.dtonomy.com/chatgpt-for-cyber-security/>