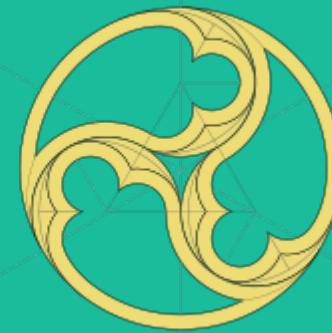


Network Traffic Analysis with

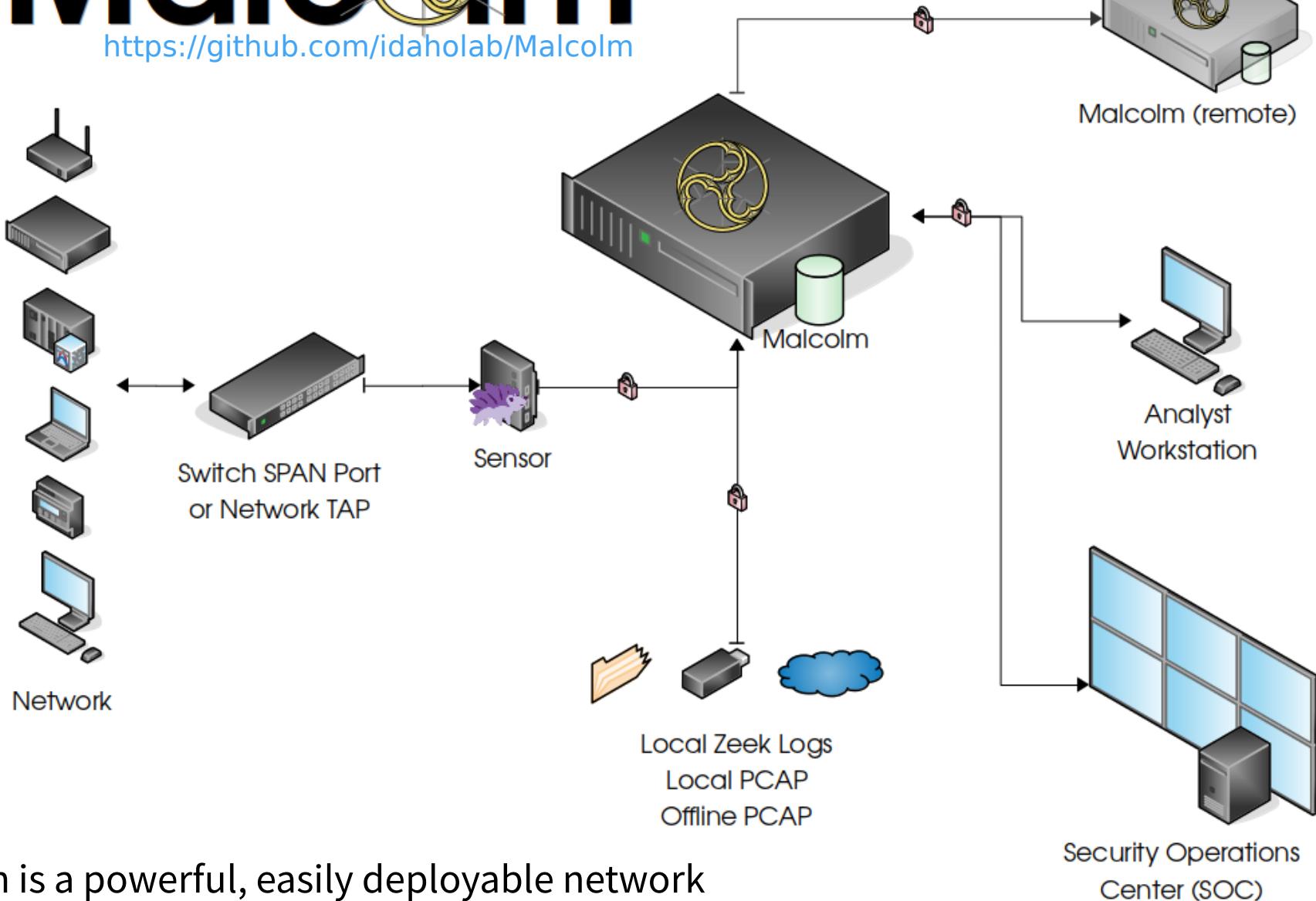
Malcolm



<https://github.com/idaholab/Malcolm>

Malcolm

<https://github.com/idaholab/Malcolm>



Malcolm is a powerful, easily deployable network traffic analysis tool suite for full packet capture artifacts (PCAP files) and Zeek logs.

Malcolm leverages several industry-standard open source tools, including:



... and more!

Supported Protocols

Link and Internet Layers

Border Gateway Protocol (BGP)

Building Automation and Control (BACnet)

Distributed Computing Env. / Remote Procedure Calls (DCE/RPC)

Dynamic Host Configuration Protocol (DHCP)

Distributed Network Protocol 3 (DNP3)

Domain Name System (DNS)

EtherNet/IP / Common Industrial Protocol (CIP)

FTP (File Transfer Protocol)

Google Quick UDP Internet Connections (gQUIC)

Hypertext Transfer Protocol (HTTP)

Internet Relay Chat (IRC)

Kerberos

Lightweight Directory Access Protocol (LDAP)

Modbus

MQ Telemetry Transport (MQTT)

MySQL

NT Lan Manager (NTLM)

Network Time Protocol (NTP)

Oracle

PostgreSQL

Process Field Net (PROFINET)

Remote Authentication Dial-In User Service (RADIUS)

Remote Desktop Protocol (RDP)

Remote Framebuffer (RFB)

S7comm / Connection Oriented Transport Protocol (COTP)

Session Initiation Protocol (SIP)

Server Message Block (SMB) / Common Internet File System (CIFS)

Simple Mail Transfer Protocol (SNMP)

Simple Network Management Protocol (SMTP)

SOCKS

Secure Shell (SSH)

Secure Sockets Layer (SSL) / Transport Layer Security (TLS)

Syslog

Tabular Data Stream (TDS)

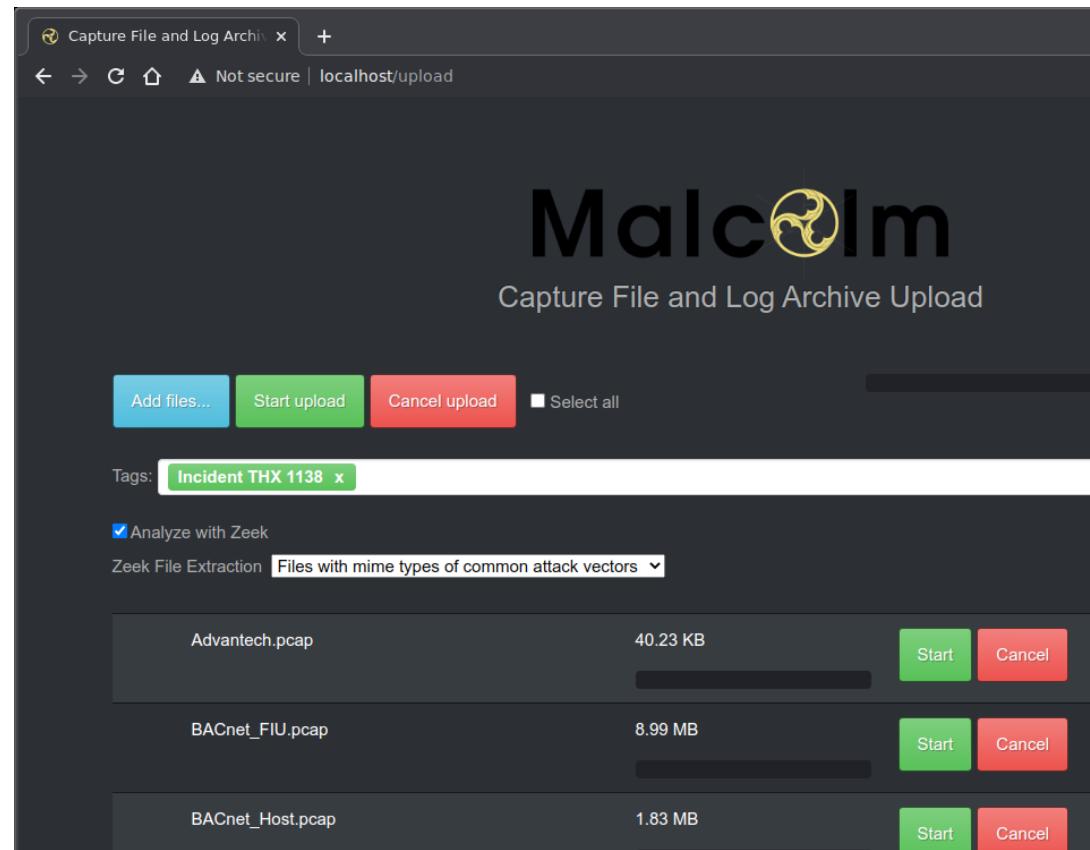
Tunnel protocols (e.g., WireGuard, GTP, GRE, Teredo, AYIYA, IP-in-IP, ...)

Installation and configuration

- Runs on any OS with Docker (Linux, macOS, Windows)
- Alternately, Linux-based Malcolm OS can be [installed](#) on hardware or in a virtual machine
- [Minimum requirements](#): 12+ GB RAM, 4+ CPU cores, “enough” storage space for traffic and logs
- Read
 - <https://github.com/idaholab/Malcolm#QuickStart>
- Watch
 - [Malcolm Network Traffic Analysis Tool Suite](#) 

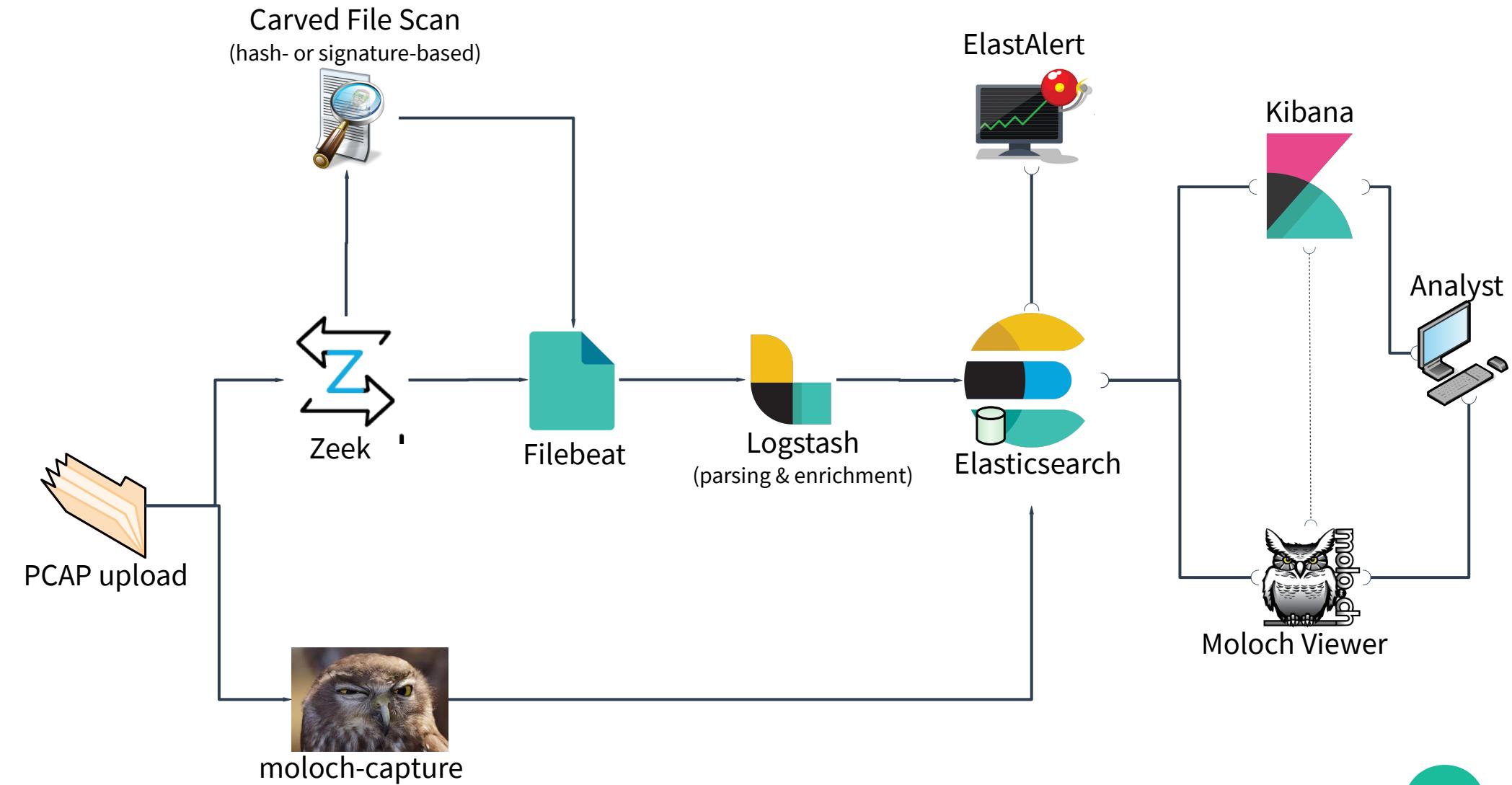
Artifact upload

- <https://localhost/upload> once Malcolm is running*
- Apply searchable tags
- Enable Zeek analysis and file extraction
 - Can also be enabled in global defaults
- Upload PCAP files or archived Zeek logs
 - pcapng not supported yet



*Links and examples assume Malcolm is accessible on localhost, YMMV depending configuration

Malcolm PCAP processing pipeline



Log enrichment

- Logstash enriches Zeek log data
 - MAC addresses to hardware vendor
 - GeolP and ASN lookups
 - Internal/external traffic based on IP ranges
 - Reverse DNS lookups
 - DNS query and hostname entropy analysis
 - Connection fingerprinting (JA3 for TLS, HASSH for SSH, Community ID for flows)

Custom host and subnet name assignment

- <https://localhost/name-map-ui>

- Assign custom names to network hosts and segments



Host and Network Segment Name Mapping

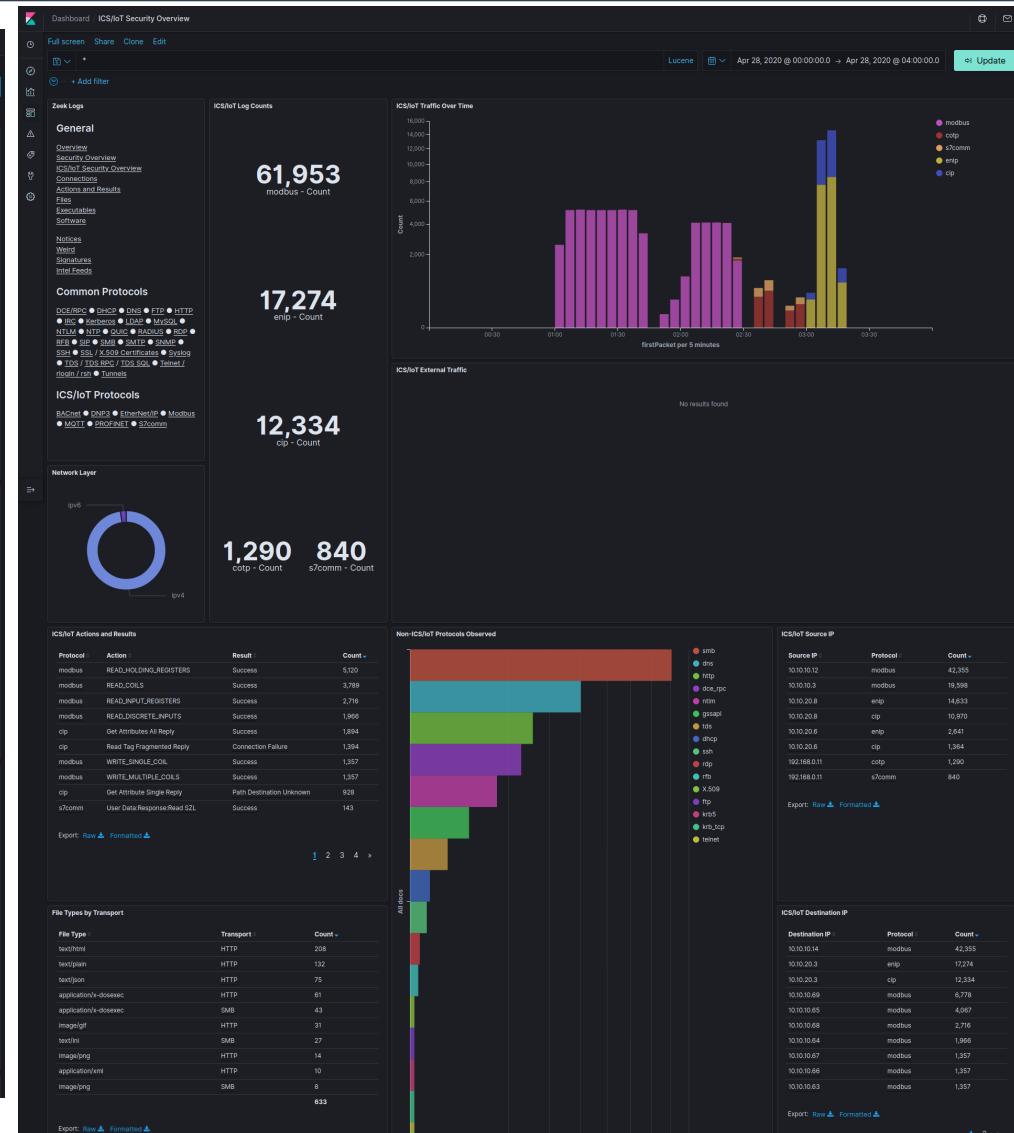
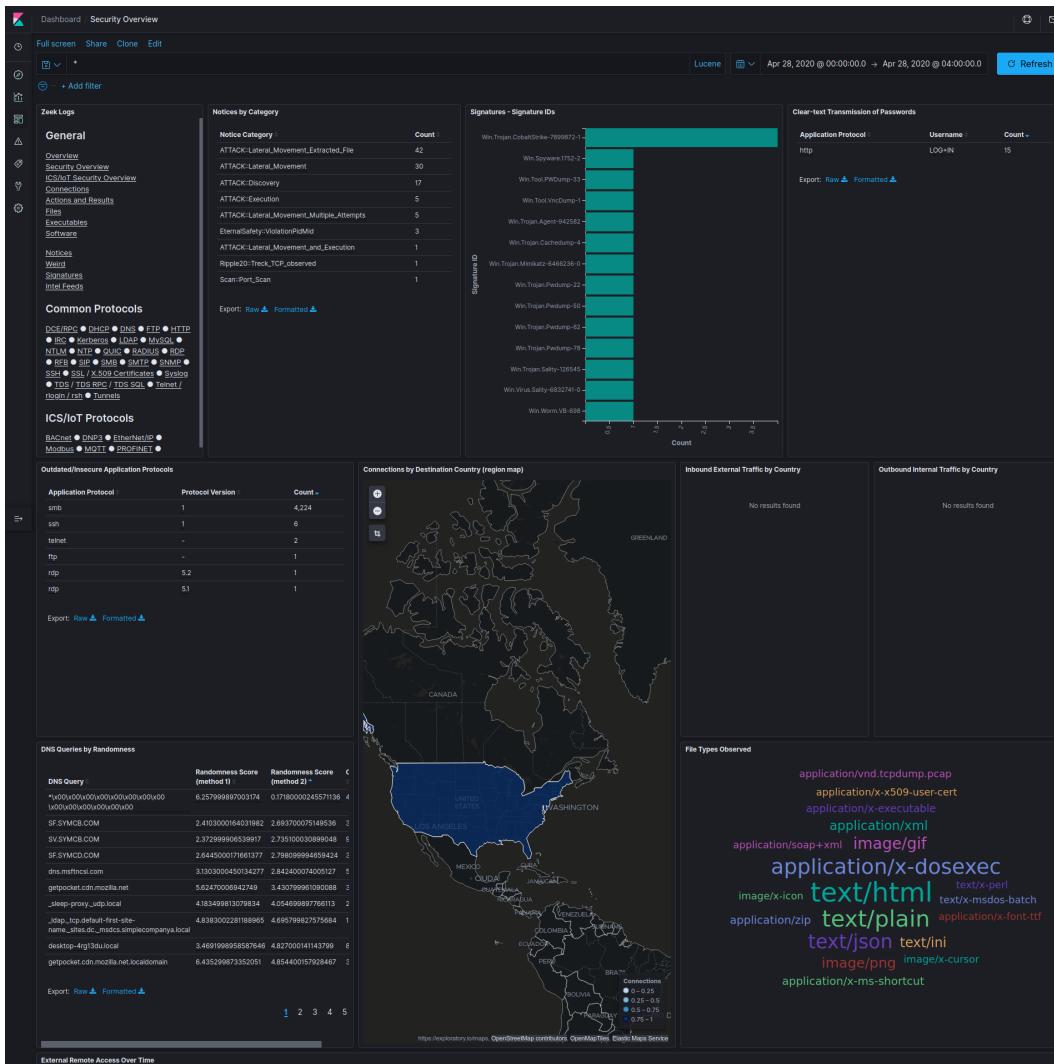
Type	Address	Name	Tag	Search mappings
segment	10.0.0.0/24	Site Office Network	Cyberville	
segment	10.10.10.0/24	Battery Network	Cyberville	
host	10.10.10.3	Schneider Battery HMI	Cyberville	
host	10.10.10.5,10.10.20.5,10.10.30.5,192.168.0.5,10.10.100.5,10.0.0.5	Historian	Cyberville	
host	10.10.10.11	Cellular Modem	Cyberville	
host	10.10.10.14,10.10.10.15,10.10.10.16,10.10.10.17,10.10.10.18	Schneider Modbus Slave	Cyberville	
segment	10.10.20.0/24	Combined Cycle BOP	Cyberville	
segment	10.10.30.0/24	Wind Turbine Network	Cyberville	
segment	10.10.100.0/24	Substation Network	Cyberville	
segment	192.168.0.0/24	Solar Panel Network	Cyberville	

host Name Tag (optional)

Kibana

- <https://localhost/kibana>
- Front end for enriched [Zeek logs](#)
- Search with [Kibana Query Language](#) or [Lucene Query Syntax](#)
- [Dashboards](#)
 - Overviews: Overview, Security Overview, ICS/IoT Security Overview, Connections, Actions and Results, Files, Executables, Software, Notices, Weird and Signatures
 - Protocol-specific: prebuilt for all [protocols](#) Malcolm understands
 - WYSIWYG editors to create custom [visualizations](#) and dashboards
 - Great for drilling down from high-level trends of network traffic to specific items of interest

Kibana: Security & ICS/IoT Security Overview

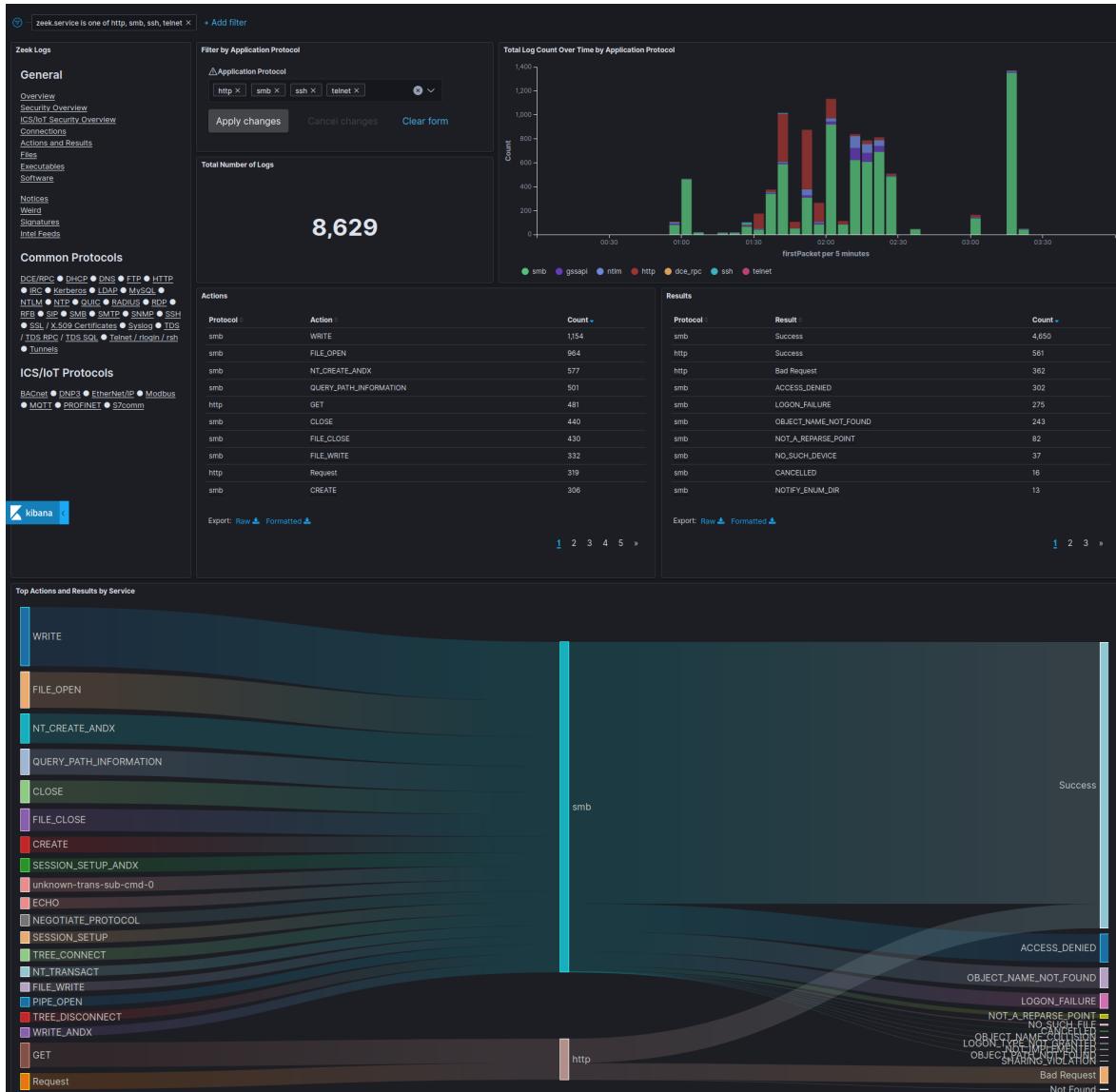


Kibana: Signatures



- Reports hits from file scanning engines (e.g., ClamAV) against files carved from network traffic

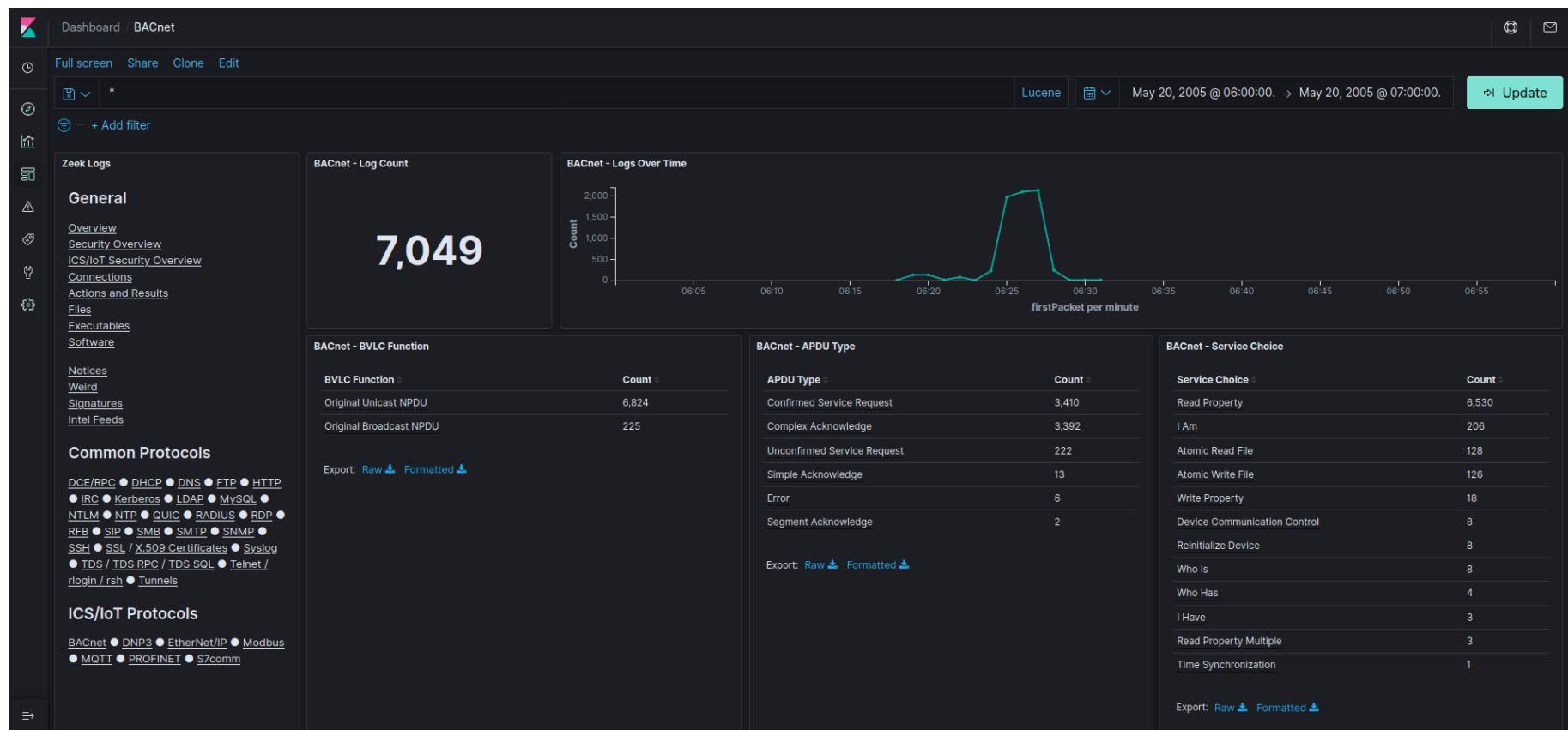
Kibana: Actions and Results



- Malcolm normalizes “action” (e.g., write, read, create file, logon, logoff, etc.) and “result” (e.g., success, failure, access denied, not found) across protocols

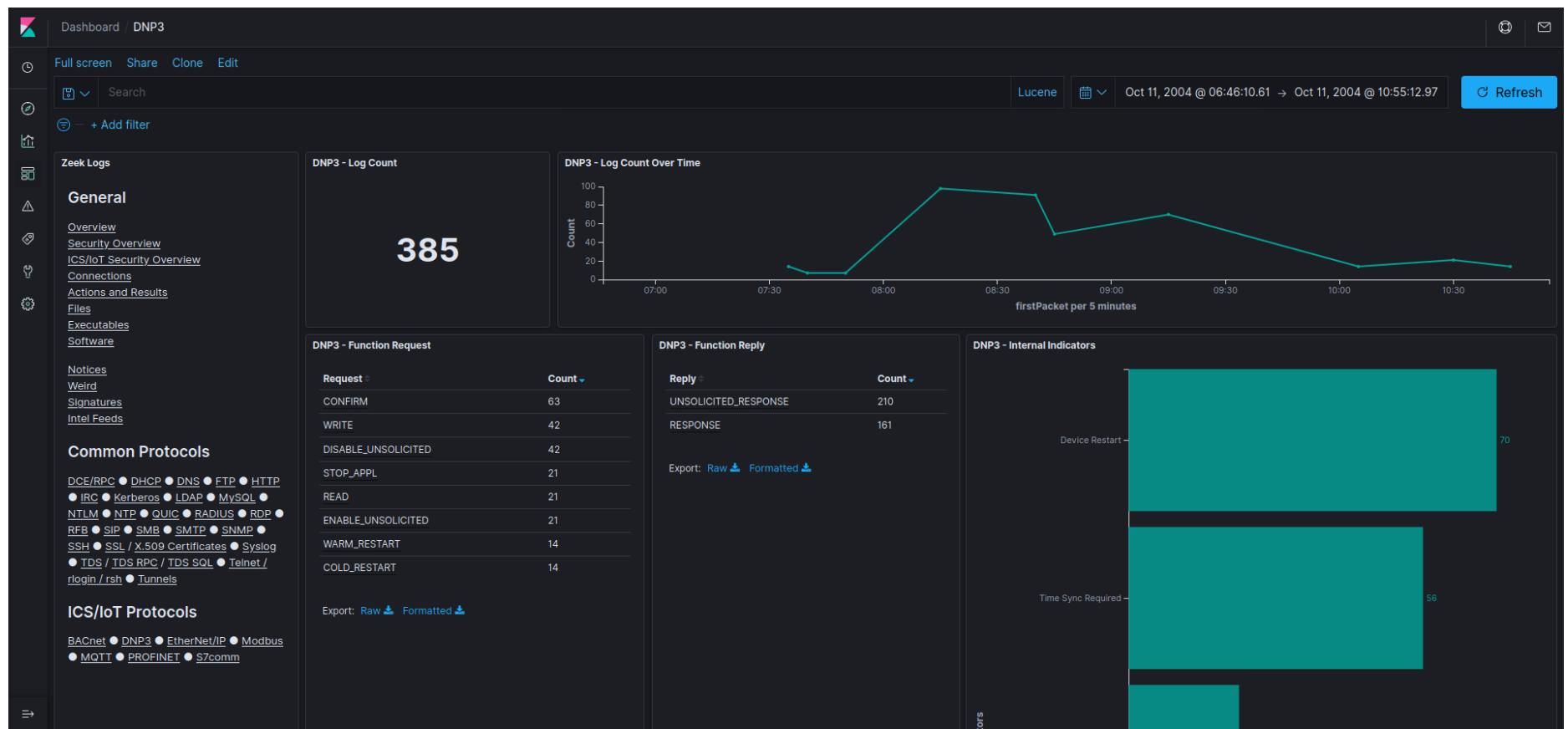
Kibana: ICS protocol dashboards

BACnet



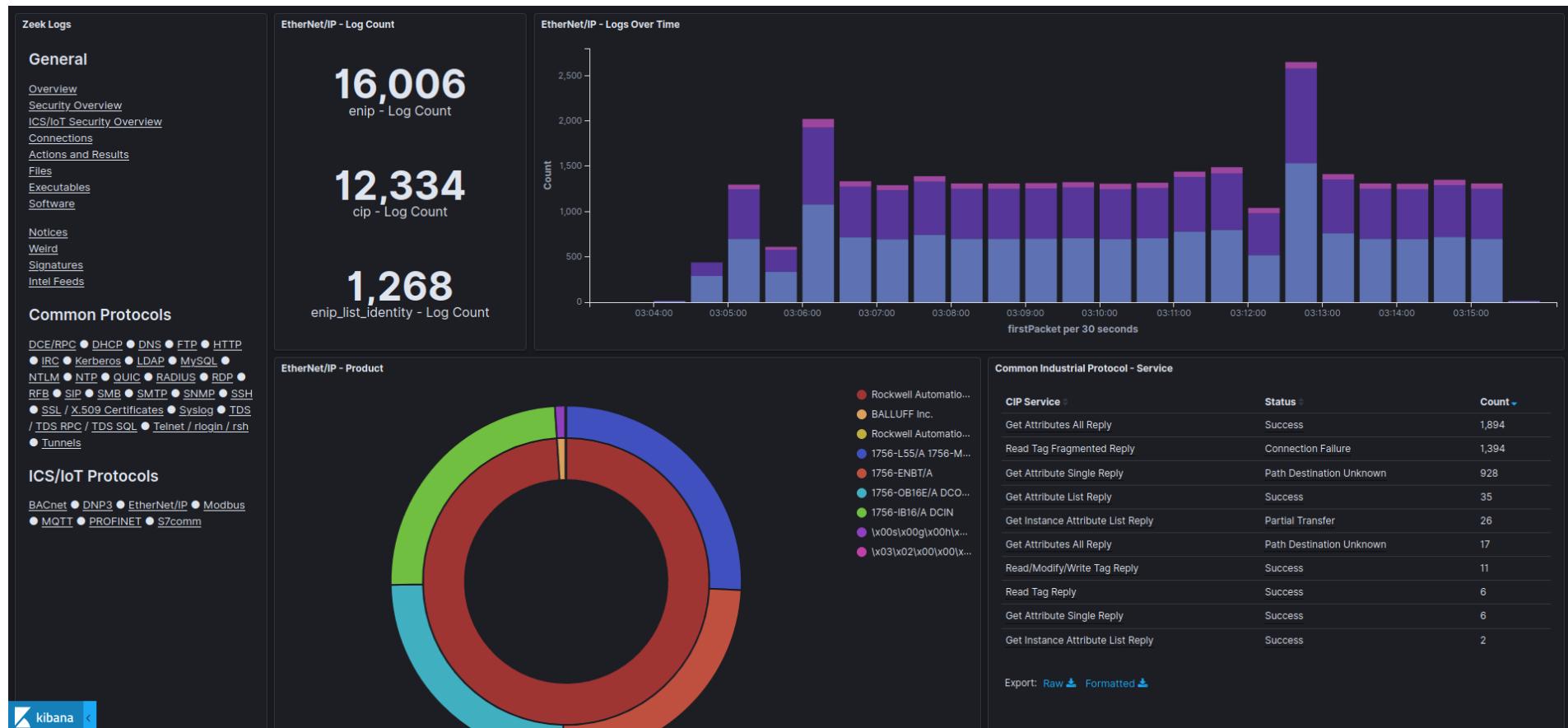
Kibana: ICS protocol dashboards

DNP3



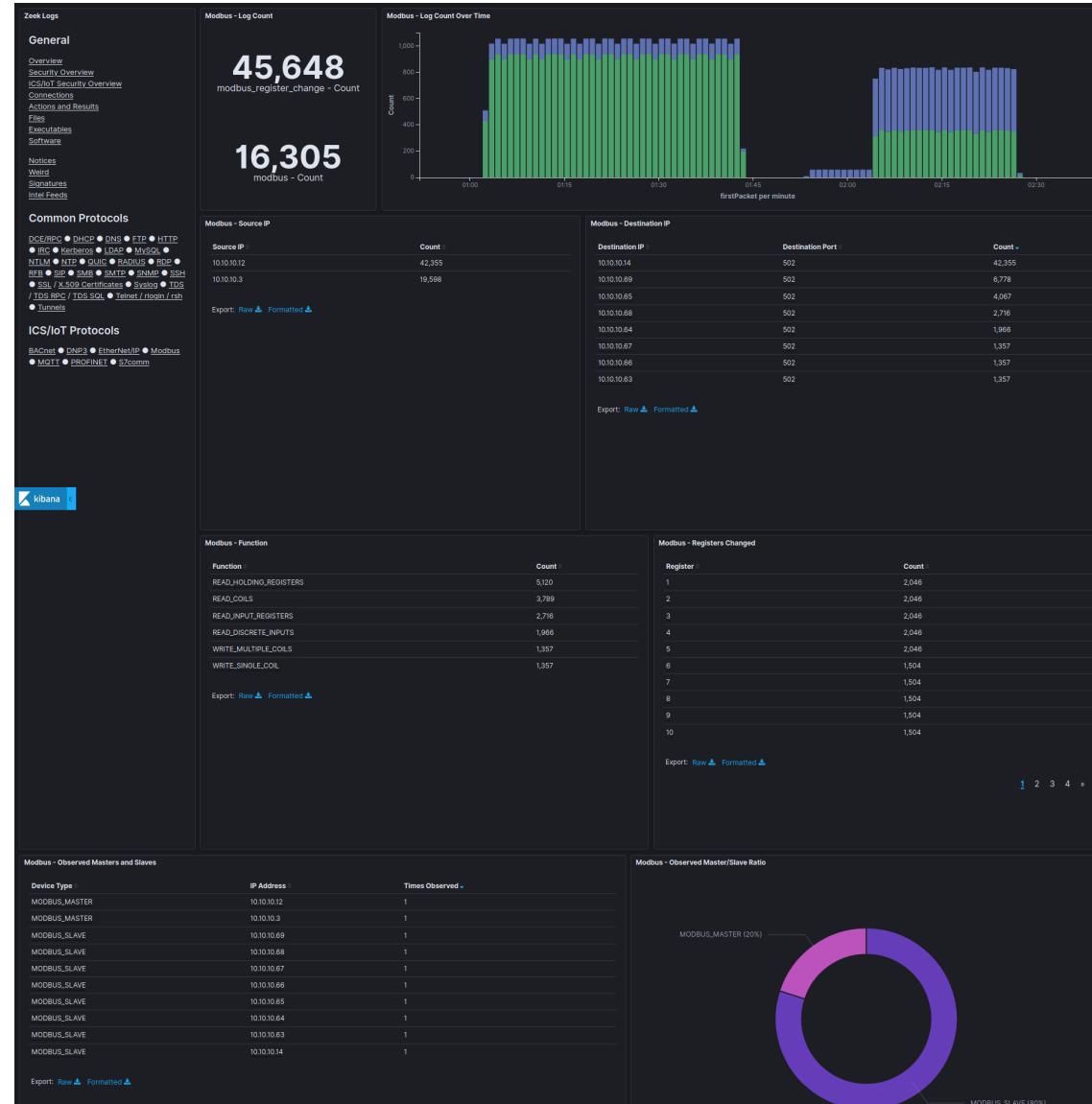
Kibana: ICS protocol dashboards

EtherNet/IP

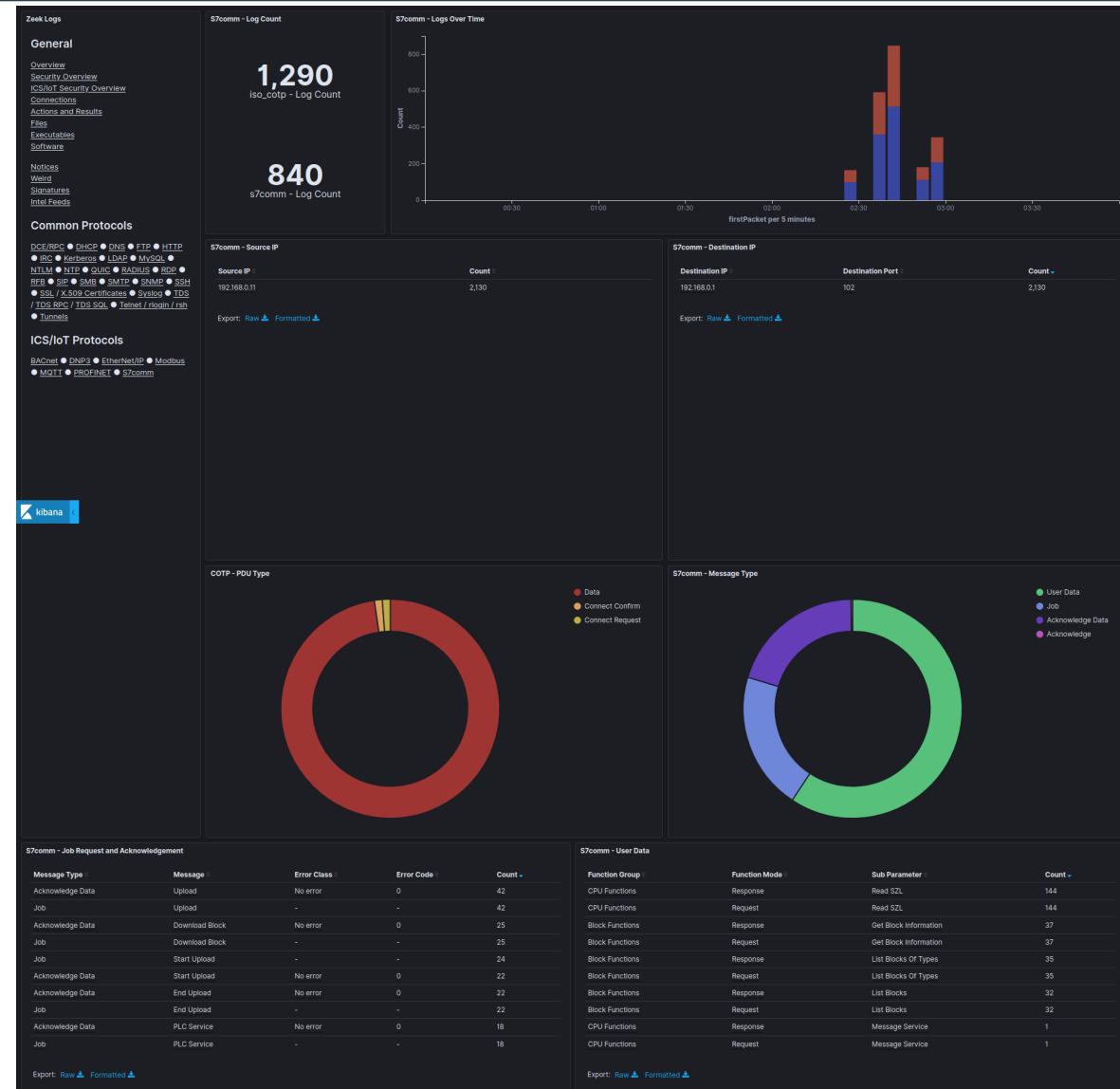


Kibana: ICS protocol dashboards

Modbus



Kibana: ICS protocol dashboards

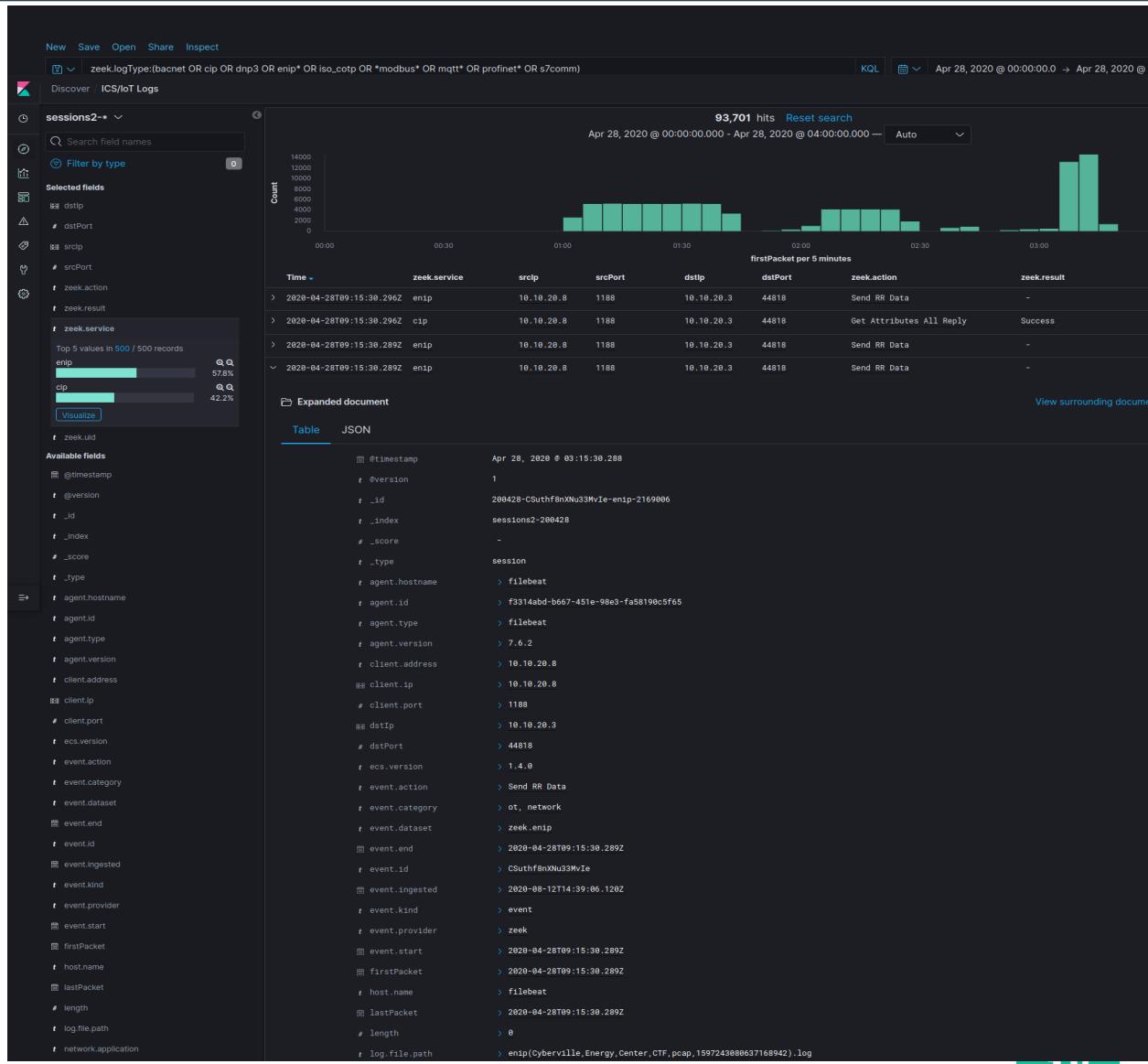


S7comm

Kibana

• Discover

- Field-level details of logs matching filter criteria
- Create and view saved searches
- View other events just before and after an event



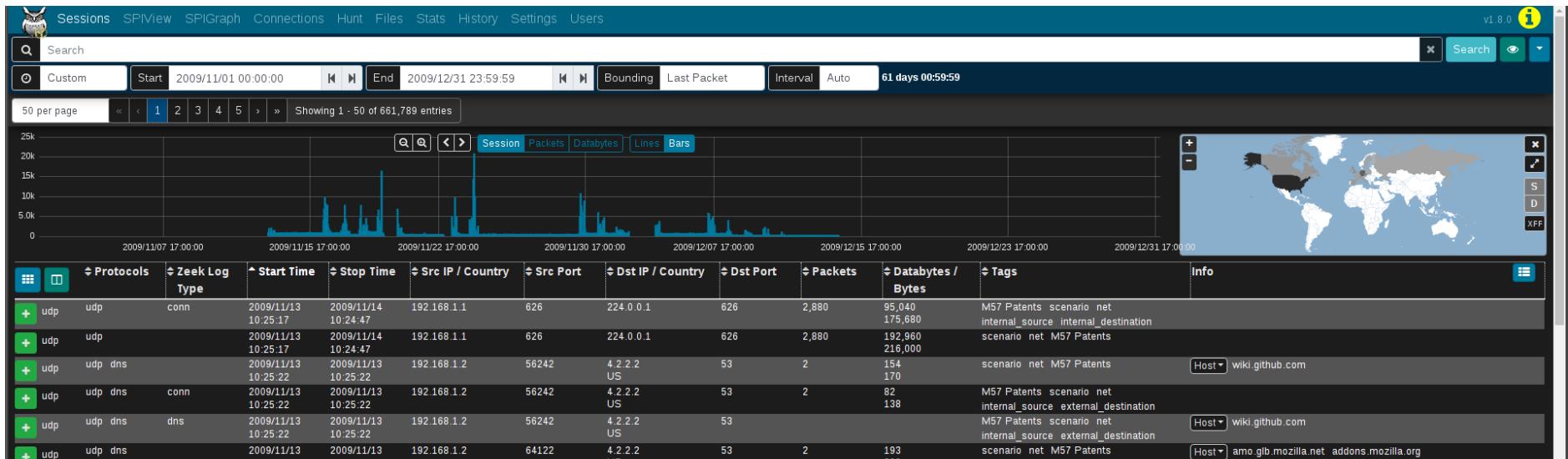
Moloch

- <https://localhost/>
- Front end for **both** enriched Zeek logs and Moloch sessions
- Filter by Zeek logs or Moloch sessions; or, view both data sources together
- “Wireshark at scale”: full PCAP availability for viewing packet payloads, exporting filtered and joined PCAP sessions and running deep-packet searches



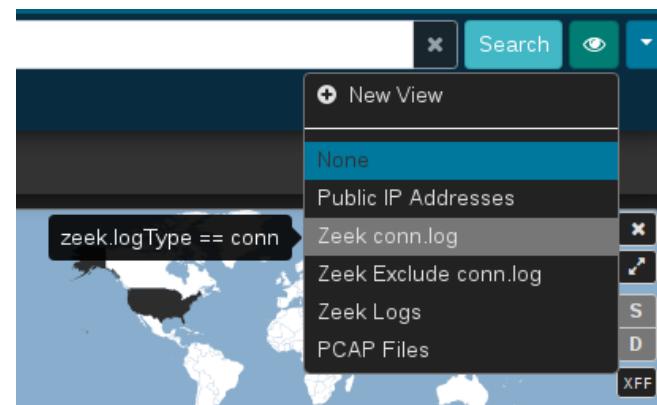
Moloch: Sessions

- Field-level details of sessions/logs matching filters



- Views: overlay saved filters on current search

- E.g., show Moloch sessions vs. Zeek logs



Moloch: Packet payloads

- Displayed for Moloch sessions with full PCAP
- File carving on the fly
- Download session PCAP
- Examine payload with CyberChef

The screenshot shows the Moloch web interface. At the top, there is a search bar with the query "ip.src == 10.10.10.3 && protocols == http && bytes > 10000". Below the search bar are various filters and controls: "Custom", "Start" (2020/04/28 00:00:00), "End" (2020/04/28 04:00:00), "Bounding", "Last Packet", "Interval" (Auto), and a timestamp "04:00:00". The main area displays a table of 83 entries, with the first few rows shown below:

Packets	Src	Dst	Timestamp	Length	Source	Destination
1	tcp	http	tcp	2020/04/28 01:59:42	10.10.10.3	57557
2	tcp	http	tcp	2020/04/28 02:01:50	10.10.10.3	57628
3	tcp	http	tcp	2020/04/28 01:57:06	10.10.10.3	57474
4	tcp	http	tcp	2020/04/28 01:57:24	10.10.10.3	57485

On the right side, there is a "File Bytes" section showing a hex dump of the payload. The bottom right corner of the slide has a small green circle with the number "21".

Moloch: PCAP Export

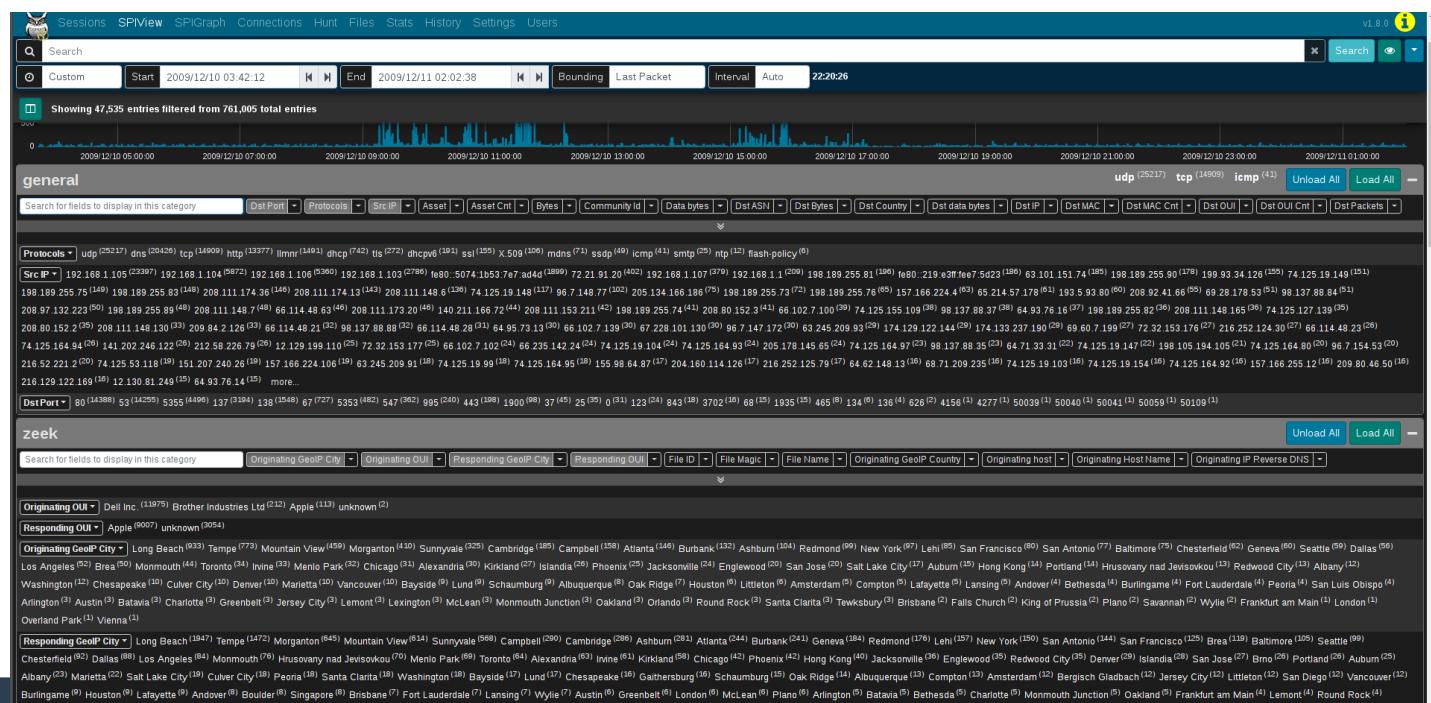
- Creates a new PCAP file from filtered sessions
- Include open, visible or all matching sessions
- Apply “PCAP Files” view to sessions first
- Narrow as much as possible prior to exporting (huge PCAP files are a pain)

The screenshot shows the Moloch web application interface. At the top, there's a navigation bar with links for Sessions, SPIView, SPIGraph, Connections, Hunt, Files, Stats, History, Settings, and Users. A search bar contains the query "country == US && protocols == http". Below the search bar, there are buttons for "Custom" and "Apply action to 7075 query matching sessions". The date range is set to "End" at "2009/11/25 09:18:20". The "Matching Items" tab is selected, showing a list of sessions. The table has columns for Protocols, Zeek Log Type, Start Time, Stop Time, Src IP / Country, Src Port, Dst IP / Country, Dst Port, Packets, Bytes, and Info. The sessions listed are all TCP connections to port 80 from various IP addresses in the US. To the right of the table is a world map showing network traffic distribution. At the bottom right, there's a button labeled "Export PCAP".

Protocols	Zeek Log Type	Start Time	Stop Time	Src IP / Country	Src Port	Dst IP / Country	Dst Port	Packets	Bytes	Info
tcp	http tcp	2009/11/25 06:15:38 MST	2009/11/25 06:15:38 MST	192.168.1.103 US	1263	198.189.255.89 US	80	33	28,482 30,358	scenario net M57 Patents URI aa.avg.com/softw/90/update/u7iaw2525u252475.bin
tcp	http tcp	2009/11/25 06:15:37 MST	2009/11/25 06:15:37 MST	192.168.1.103 US	1261	198.189.255.89 US	80	106	106,451 112,439	scenario net M57 Patents URI aa.avg.com/softw/90/update/u7avi1787u170575.bin
tcp	http tcp	2009/11/25 06:02:51 MST	2009/11/25 06:02:51 MST	192.168.1.106 US	1252	198.189.255.89 US	80	33	28,487 30,363	scenario net M57 Patents URI aa.avg.com/softw/90/update/u7iaw2525u252475.bin
tcp	http tcp	2009/11/25 06:02:51 MST	2009/11/25 06:02:51 MST	192.168.1.106 US	1250	198.189.255.89 US	80	104	106,456 112,316	scenario net M57 Patents URI aa.avg.com/softw/90/update/u7avi1787u170575.bin
tcp	http tcp	2009/11/25 04:36:23 MST	2009/11/25 04:36:23 MST	192.168.1.104 US	1072	198.189.255.75 US	80	34	28,487 30,427	scenario net M57 Patents URI aa.avg.com/softw/90/update/u7iaw2525u252475.bin
tcp	http tcp	2009/11/25 04:36:22 MST	2009/11/25 04:36:23 MST	192.168.1.104 US	1070	198.189.255.75 US	80	106	106,456 112,456	scenario net M57 Patents out-of-order-dst URI aa.avg.com/softw/90/update/u7avi1787u170575.bin
tcp	http tcp	2009/11/25	2009/11/25	192.168.1.104 US	1058	198.189.255.75 US	80	11	3,605	scenario net M57 Patents URI iawdls-ssd.sun.com/update/1.6.0/man-1.6.0.xml

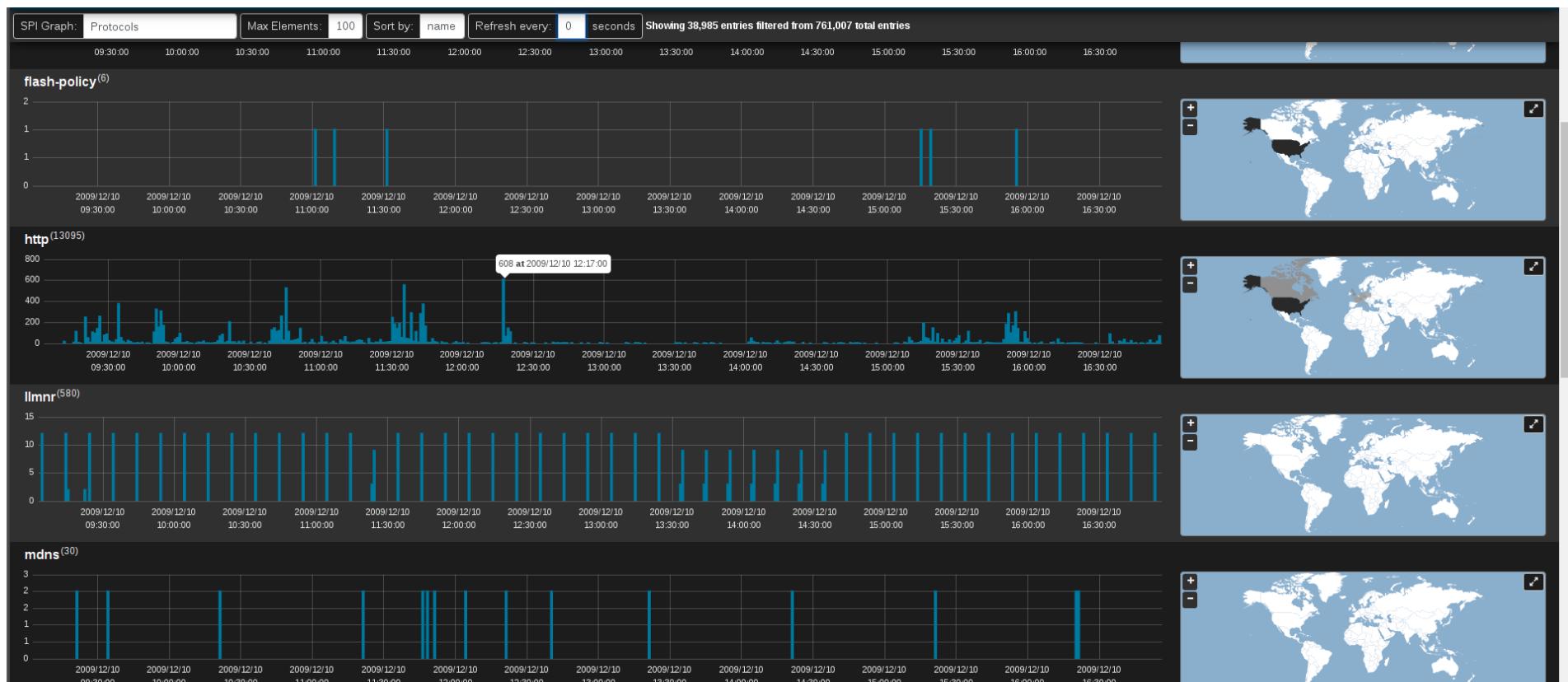
Moloch: SPIView

- Explore “top n ” and field cardinality for all fields of both Moloch sessions and Zeek logs
- Apply filters or pivot to Sessions or SPIGraph view for field values of interest
- Limit search to ≤ 1 week before using as it runs many queries



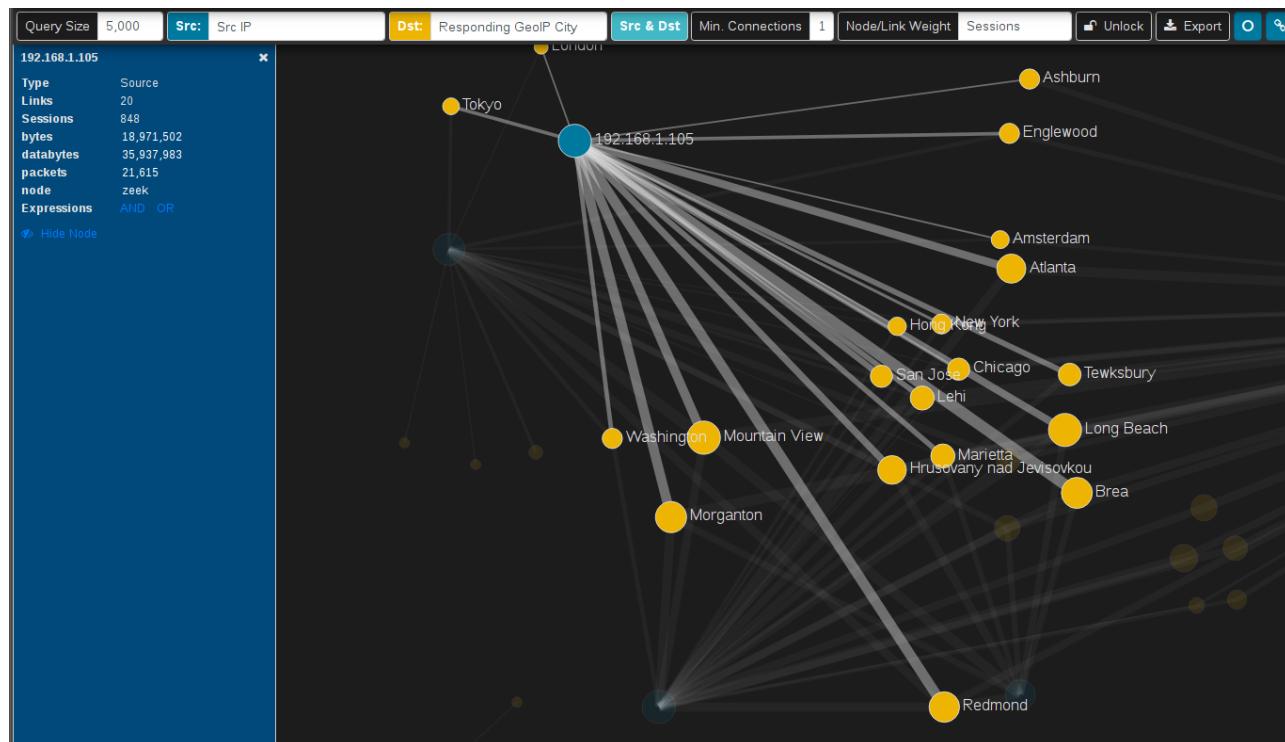
Moloch: SPIGraph

- View “top n ” field values chronologically and geographically
- Identify trends and patterns in network traffic



Moloch: Connections

- Visualize logical relationship between hosts
- Use any combination of fields for source and dest. nodes
- Compare current vs. baseline traffic



Moloch: Hunt

- Deep-packet search (“PCAP grep”) of session payloads
- Search for ASCII, hex codes or regular expression matches
- Apply “PCAP Files” view to sessions first

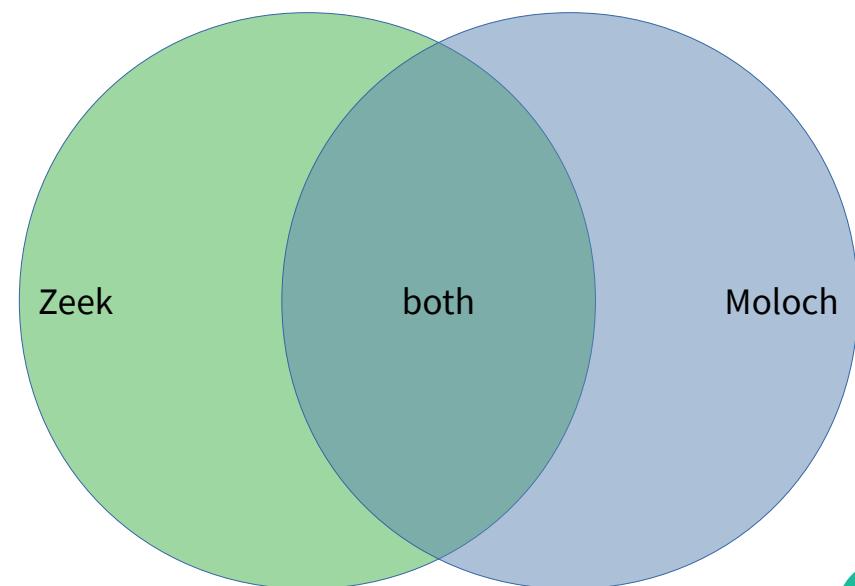
The screenshot shows the Moloch Hunt interface. At the top, there's a navigation bar with links: Sessions, SPIView, SPIGraph, Connections, Hunt, Files, Stats, History, Settings, and Users. Below the navigation bar is a search bar containing the query "protocols == http". Underneath the search bar are buttons for "All (careful)", "Start" (set to 1969/12/31 17:00:00), "End" (set to 2019/05/28 09:19:44), and "Bounding" and "Last Packet" buttons. A status message indicates "Creating a new packet search job will search the packets of 43,818 sessions." Below this is a "Hunt Job Queue" table with the following data:

Status	Matches	Name	User	Search text	Notify	Created	ID
* 62.3%	297	HTTP with password	analyst	password (ascii)		2019/05/28 09:20:28	vRgH_2oB-8T3HMe4R6WA

Below the table, there's a summary message: "This hunt is running". It also displays the last update time (2019/05/28 09:21:06), the number of raw source and destination packets per session (50), the total number of sessions searched (27,299), the total sessions to search (43818), the session query expression (protocols == http), and the session search time range (1969/12/31 17:00:00 to 2019/05/28 09:19:44).

Using Moloch and Kibana together in Malcolm

- Search syntax is different between Moloch and Kibana (and in some cases, so are **field names**): see [Malcolm documentation](#) for examples and [Moloch help](#)
- Despite considerable overlap, there are differences in **protocol parser support** between Zeek and Moloch
 - Learning the strengths of each will help you more effectively find the good stuff

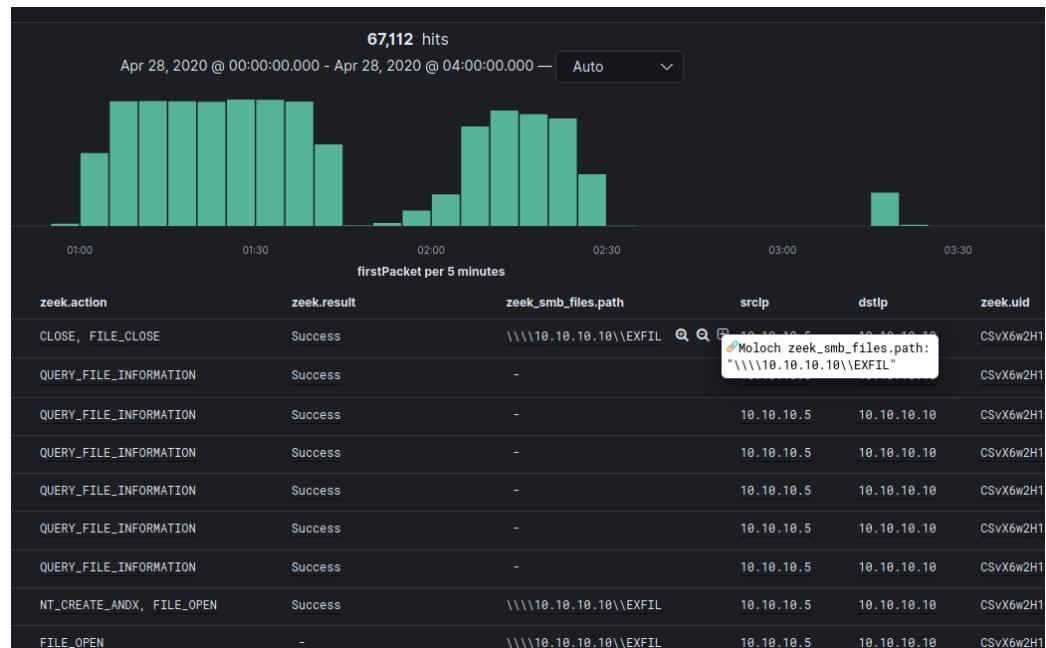


Pivots: Moloch ↔ Kibana

- Moloch → Kibana

The screenshot shows the Moloch interface. At the top, there's a navigation bar with links: Sessions, SPIView, SPIGraph, Connections, Hunt, Files, Stats, History, and Settings. Below the navigation bar is a search bar with a custom date range from 2020/04/28 00:00:00 to 2020/04/28 04:00:00. The main content area displays SMB activity for host 10.10.10.10, showing shares EXFIL and file \\alarms.png. Below this, the "Zeek Common Fields" section lists various fields with their values: Zeek Connection ID (CSvX6w2H1DADdrC3k8), Zeek Log Type (smb_cmd smb_files), Zeek Node (filebeat), Originating Host (10.10.10.5), Originating Host Name (Historian), Originating Network Se... (Battery Network), Responding Host (10.10.10.10), Responding Network S... (Battery Network), Originating Port, Responding Port, Protocol, Service, Service Version, Action, and Result. A tooltip for the "Result" field shows options: New Sessions Tab, Copy value, and Success.

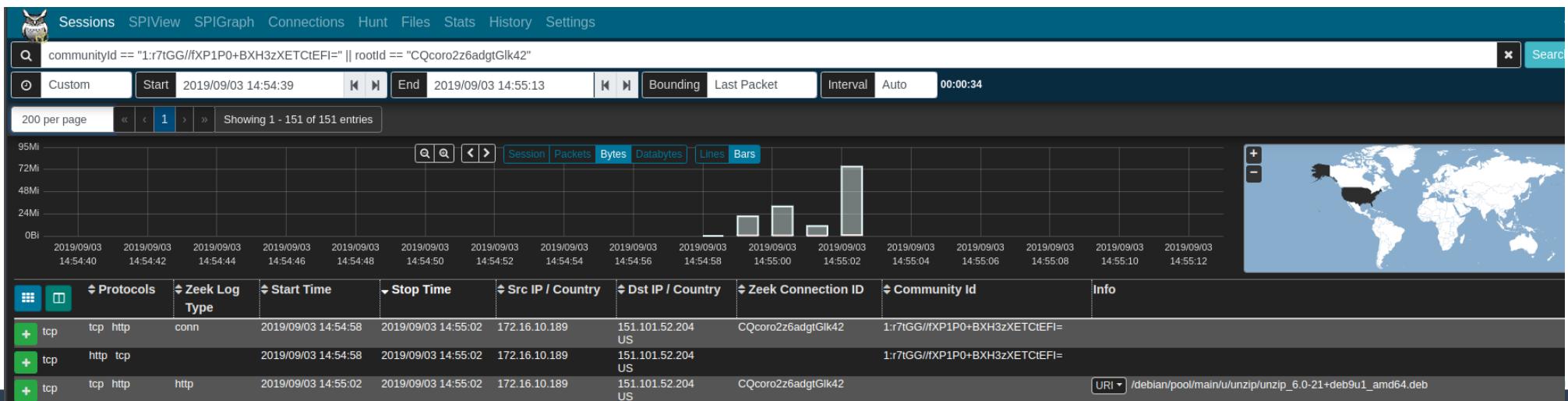
- Kibana → Moloch



Check query search time frame after pivot:
sometimes it gets lost in translation.

Correlating Moloch sessions and Zeek logs

- Correlate Zeek logs and Moloch sessions using common fields
- communityId fingerprints flows in both and can bridge the two
- zeek.uid/rootId filters Zeek logs for the same session
- Filter community ID OR'ed with zeek UID to see all Moloch sessions and Zeek logs for the same traffic
 - `communityId == "1:r7tGG//fXP1P0+BXH3zXETCtEFI=" || rootId == "CQcoro2z6adgtGlk42"`



Search tips

- Always check your search time frame
- “Zoom in” (apply filters) for a particular field value, pivot to another field then “zoom out” (remove filters)
- Most UI controls can work with any data field (1000+)
- Filter on zeek . logType (e.g., conn to see conn.log)
- Filter on protocol or both Moloch and Zeek regardless of data source (e.g., protocol:http in Kibana and protocols == http in Moloch)
- tags field
 - Populated for both Moloch sessions and Zeek logs with tags provided on upload and words extracted from filenames
 - internal_source, internal_destination, external_source, external_destination, cross_segment

Thanks! Visit [Malcolm on GitHub](#)
**to read the docs, make
suggestions, report issues and**
★Star to show your support!



Malcolm is Copyright © 2020 Battelle Energy Alliance, LLC, and is developed and released through the cooperation of the Cybersecurity and Infrastructure Security Agency of the US Department of Homeland Security.

Network Traffic Analysis with

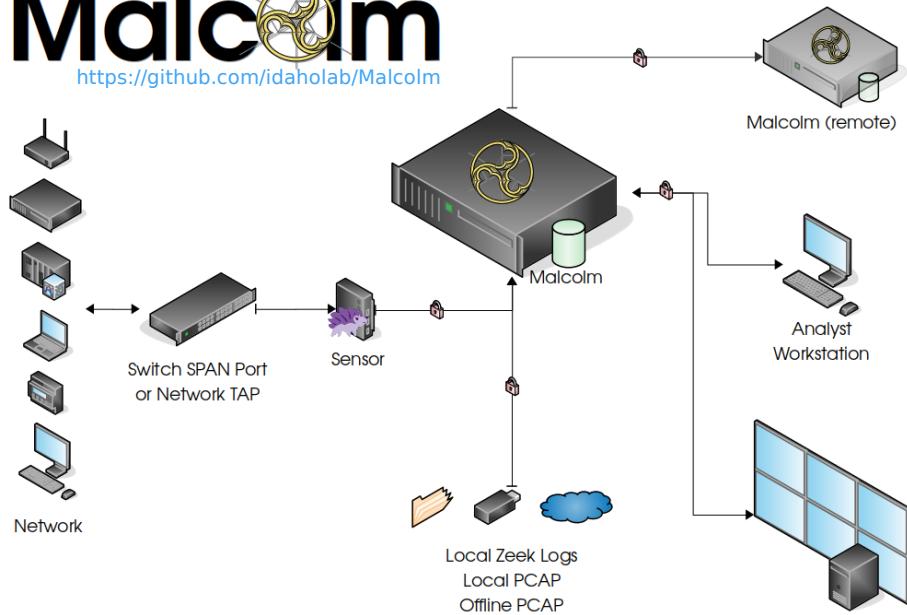
Malcolm



<https://github.com/idaholab/Malcolm>

Malcolm

<https://github.com/idaholab/Malcolm>



Malcolm is a powerful, easily deployable network traffic analysis tool suite for full packet capture artifacts (PCAP files) and Zeek logs.

2

Malcolm is an easily deployable collection of open source tools for network traffic analysis that aims to be “greater than the sum of its parts.” It provides a framework of interoperability for these tools (which we’ll discuss through the course of this presentation) to provide visibility into the network traffic it inspects.

Malcolm is flexible: it can be used as a standalone tool for deployments or one-off analyses, or as part of a larger long-term network deployment with dedicated network sensors, integration with domain authentication servers and forwarding collected enriched log data offsite to another Malcolm instance.

Developed at the Idaho National Lab under the direction of the US Department of Homeland Security’s CISA, Malcolm aims to give special focus to traffic found in industrial control systems networks, while still remaining useful in typical corporate IT environments.

Malcolm leverages several industry-standard open source tools, including:



Supported Protocols

Link and Internet Layers	Network Time Protocol (NTP)
Border Gateway Protocol (BGP)	Oracle
Building Automation and Control (BACnet)	PostgreSQL
Distributed Computing Env. / Remote Procedure Calls (DCE/RPC)	Process Field Net (PROFINET)
Dynamic Host Configuration Protocol (DHCP)	Remote Authentication Dial-In User Service (RADIUS)
Distributed Network Protocol 3 (DNP3)	Remote Desktop Protocol (RDP)
Domain Name System (DNS)	Remote Framebuffer (RFB)
EtherNet/IP / Common Industrial Protocol (CIP)	S7comm / Connection Oriented Transport Protocol (COTP)
FTP (File Transfer Protocol)	Session Initiation Protocol (SIP)
Google Quick UDP Internet Connections (gQUIC)	Server Message Block (SMB) / Common Internet File System (CIFS)
Hypertext Transfer Protocol (HTTP)	Simple Mail Transfer Protocol (SNMP)
Internet Relay Chat (IRC)	Simple Network Management Protocol (SMTP)
Kerberos	SOCKS
Lightweight Directory Access Protocol (LDAP)	Secure Shell (SSH)
Modbus	Secure Sockets Layer (SSL) / Transport Layer Security (TLS)
MQ Telemetry Transport (MQTT)	Syslog
MySQL	Tabular Data Stream (TDS)
NT Lan Manager (NTLM)	Tunnel protocols (e.g., WireGuard, GTP, GRE, Teredo, AYIYA, IP-in-IP, ...)

Common ICS protocols are listed in **bold**

3

The components comprising Malcolm are industry-standard open source tools, which makes it easy to integrate Malcolm with other solutions in those tools' respective ecosystems and straightforward to contribute to the project.

Malcolm's primary components are the Elastic Stack (namely, Elasticsearch, Logstash, Kibana and Beats); the Zeek network security monitor (formerly Bro); and Moloch, a tool for large-scale indexed packet capture and search.

Malcolm can interpret network traffic across dozens of application protocols, including several protocols commonly seen in OT networks. Much of Malcolm's development is dedicated to improving Malcolm's coverage of protocols used by ICS devices.

Installation and configuration

- Runs on any OS with Docker (Linux, macOS, Windows)
- Alternately, Linux-based Malcolm OS can be [installed](#) on hardware or in a virtual machine
- [Minimum requirements](#): 12+ GB RAM, 4+ CPU cores, “enough” storage space for traffic and logs
- Read
 - <https://github.com/idaholab/Malcolm#QuickStart>
- Watch
 - [Malcolm Network Traffic Analysis Tool Suite](#) 

4

Malcolm can be installed and run on any system that supports Docker, a containerization platform for software services. It has been tested on Linux, macOS and Microsoft Windows.

Most modern commodity hardware (including laptops, desktops and servers) are configured with the resources needed to run Malcolm. Available system memory tends to be the most crucial, with 12GB as a minimum and 16GB or more recommended.

This presentation does not cover installation and configuration of Malcolm. Please refer to the documentation and examples found at the links provided.

Artifact upload

- <https://localhost/upload> once Malcolm is running*
- Apply searchable tags
- Enable Zeek analysis and [file extraction](#)
 - Can also be enabled in global defaults
- Upload PCAP files or archived Zeek logs
 - pcapng not supported yet



5

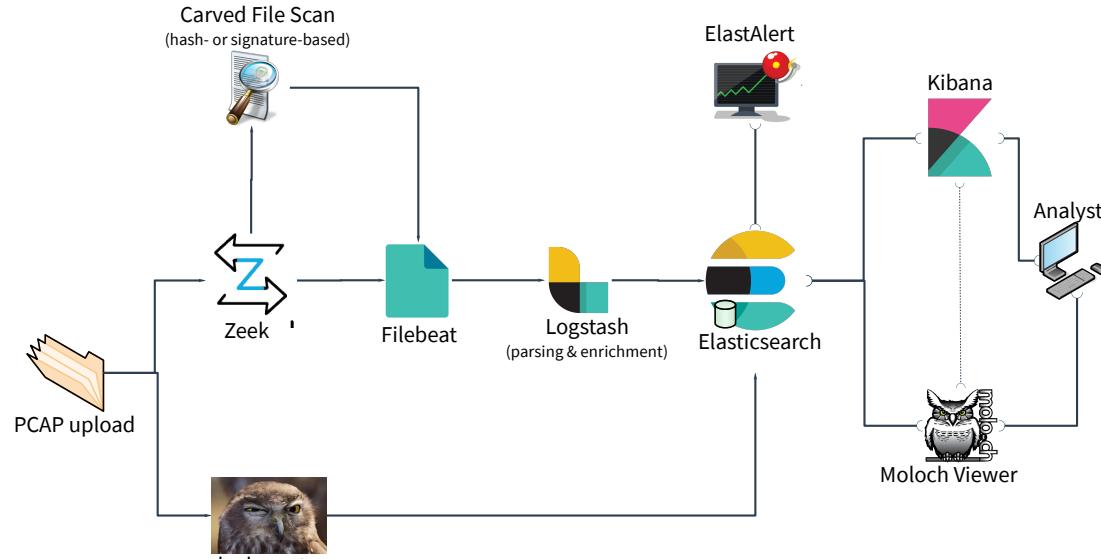
*Links and examples assume Malcolm is accessible on localhost, YMMV depending configuration

Once Malcolm is up and running, it must be provided with captured network traffic to interpret, index and present for analysis. While this may be accomplished by dedicated network sensors, this presentation will address the use case of previously captured network traffic artifacts: namely, PCAP files.

Malcolm's user interfaces are accessible via web browser. PCAP files can be uploaded into Malcolm for processing by accessing /upload on the host on which Malcolm is running.

Prior to starting the upload, "tags" may be added which will allow the data from the PCAP file(s) being uploaded to be searchable using those tags later on. Other behavior relating to how the PCAP file is parsed can also be customized on this page.

Malcolm PCAP processing pipeline



6

An uploaded PCAP file goes through several steps on its way to becoming enriched, indexed and user-searchable.

Upon upload, Malcolm generates metadata for the network traffic represented in a PCAP file using both Zeek and moloch-capture. moloch-capture aggregates metadata for a particular network connection into what it calls a “session” record, which is written to Elasticsearch for indexing.

Zeek generates several log files, primarily broken out by application protocol, which contains metadata similar to that generated by moloch-capture. Malcolm can also leverage Zeek’s ability to “carve” out files transferred over the network. These files can be scanned (for example, by an antivirus tool) or preserved for analysis with external tools. The Zeek logs are forwarded by Filebeat to Logstash for further enrichment, normalized to the same field schema as the corresponding Moloch sessions and then indexed into Elasticsearch.

Once ingested by Elasticsearch, Malcolm provides two interfaces for visualizing network traffic: Kibana and Moloch Viewer.

Log enrichment

- Logstash enriches Zeek log data
 - MAC addresses to hardware vendor
 - GeoIP and ASN lookups
 - Internal/external traffic based on IP ranges
 - Reverse DNS lookups
 - DNS query and hostname [entropy analysis](#)
 - Connection fingerprinting ([JA3](#) for TLS, [HASSH](#) for SSH, [Community ID](#) for flows)

7

Before taking a harder look at the Kibana and Moloch UIs, let's talk a moment about fields Logstash can use to enrich log data before it is written to the database.

- MAC addresses are mapped to hardware manufacturers where possible (to indicate, for example, whether a device was manufactured by Schneider Electric, Schweitzer Engineering or Rockwell Automation)
- GeoIP, ASN and (optionally) reverse DNS lookups are performed for routable IP addresses
- Character frequency analysis is performed for DNS responses and some other hostnames to detect DGA (domain generation algorithm) hostnames often used by malware
- Community-standard fingerprinting algorithms are applied where applicable to allow Malcolm's data to be cross-referenced with other tools

Custom host and subnet name assignment

- <https://localhost/name-map-ui>
- Assign custom names to network hosts and segments

Type	Address	Name	Tag	Actions
segment	10.0.0.0/24	Site Office Network	Cyberville	
segment	10.10.10.0/24	Battery Network	Cyberville	
host	10.10.10.3	Schneider Battery HMI	Cyberville	
host	10.10.10.5,10.10.20.5,10.10.30.5,192.168.0.5,10.10.100.5,10.0.0.5	Historian	Cyberville	
host	10.10.10.11	Cellular Modem	Cyberville	
host	10.10.10.14,10.10.15,10.10.16,10.10.17,10.10.18	Schneider Modbus Slave	Cyberville	
segment	10.10.20.0/24	Combined Cycle BOP	Cyberville	
segment	10.10.30.0/24	Wind Turbine Network	Cyberville	
segment	10.10.100.0/24	Substation Network	Cyberville	
segment	192.168.0.0/24	Solar Panel Network	Cyberville	

8

The Host and Network Segment Name Mapping interface allows you to assign names for network segments and hosts based on IP and/or MAC addresses in Zeek logs.

As Zeek logs are processed into Malcolm's Elasticsearch instance, the log's source and destination IP and MAC address fields (`zeek.orig_h`, `zeek.resp_h`, `zeek.orig_l2_addr`, and `zeek.resp_l2_addr`, respectively) are compared against the list of "host" addresses provided. When a match is found, a new field is added to the log: `zeek.orig_hostname` or `zeek.resp_hostname`, depending on whether the matching address belongs to the originating or responding host. For traffic matching the list of "segment" addresses provided, `zeek.orig_segment` and `zeek.resp_segment` fields are added. If both `zeek.orig_segment` and `zeek.resp_segment` are added to a log, and if they contain different values, the tag `cross_segment` will be added to the log's `tags` field for convenient identification of cross-segment traffic.

If the "required tag" field is specified, a log must also contain that value in its `tags` field in addition to matching the IP or MAC address specified in order for the corresponding name assignment to be made.

Kibana

- <https://localhost/kibana>
- Front end for enriched Zeek logs
- Search with [Kibana Query Language](#) or [Lucene Query Syntax](#)
- [Dashboards](#)
 - Overviews: Overview, Security Overview, ICS/IoT Security Overview, Connections, Actions and Results, Files, Executables, Software, Notices, Weird and Signatures
 - Protocol-specific: prebuilt for all [protocols](#) Malcolm understands
 - WYSIWYG editors to create custom [visualizations](#) and dashboards
 - Great for drilling down from high-level trends of network traffic to specific items of interest

9

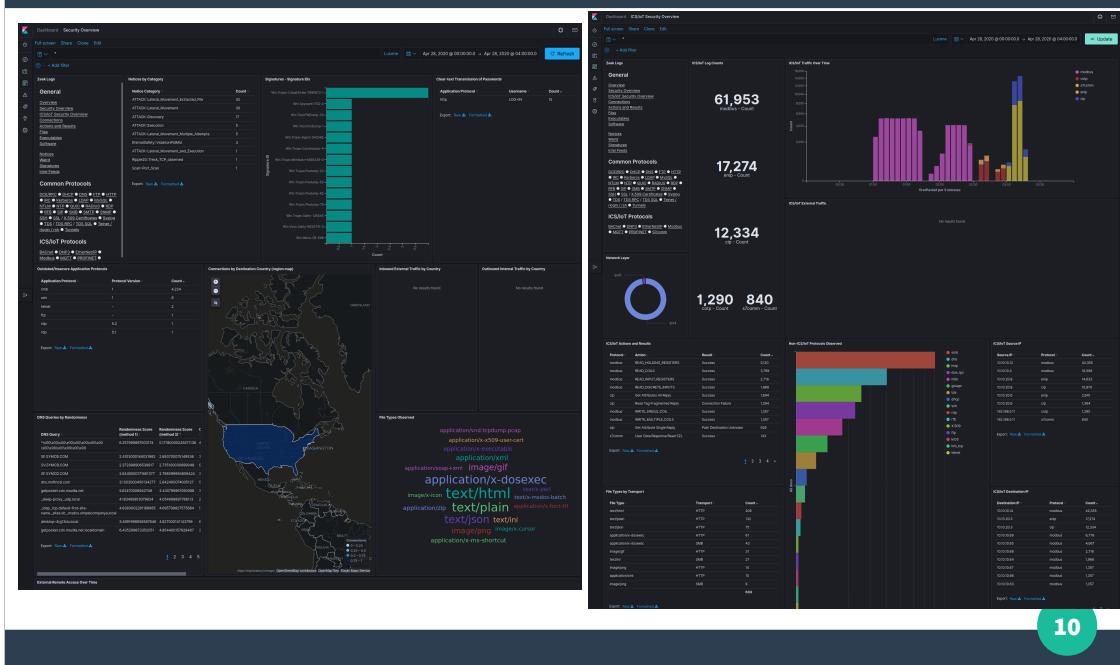
Kibana is one of Malcolm’s two user interfaces for visualizing log data.

Where Kibana really shines is in providing intuitive interactive representations of log data that simplify the process of recognizing and narrowing in on important network events: starting from a high-level overview and being able to quickly “drill-down” to the traffic of an individual host or connection of interest.

Malcolm comes with dozens of prebuilt visualizations specifically for data ingested from Zeek logs. Its dashboards fall into two categories: overview dashboards and protocol-specific dashboards. We’ll review some of these in a moment.

Aside from its prebuilt offerings, Kibana provides an easy drag-and-drop WYSIWYG editor for creating new visualizations on the fly.

Kibana: Security & ICS/LoT Security Overview



The Security Overview and ICS/LoT Security Overview dashboards highlight events that may be of particular interest from a security standpoint, including Zeek notices, signatures triggered from file scans, clear-text transmission of passwords, outdated or insecure versions of application protocols, traffic originating from or directed to public IP addresses, file transfers and more. These dashboards are a good place to start when looking for indicators of compromise or vulnerabilities in network traffic.

Kibana: Signatures



- Reports hits from file scanning engines (e.g., ClamAV) against files carved from network traffic

11

If Zeek file carving is enabled, questionable files will be written to the signatures log and reported in the signatures dashboard.

The Zeek connection UID (`zeek.uid`) and file UID (`zeek.fuid`) fields in these logs can be used to cross reference to other visualizations to provide the context for how file was transferred.

In addition to being reported in the signatures log, Malcolm can be configured to preserve files carved from network traffic (either all files or just suspicious ones) for further examination.

Kibana: Actions and Results



- Malcolm normalizes “action” (e.g., write, read, create file, logon, logoff, etc.) and “result” (e.g., success, failure, access denied, not found) across protocols

12

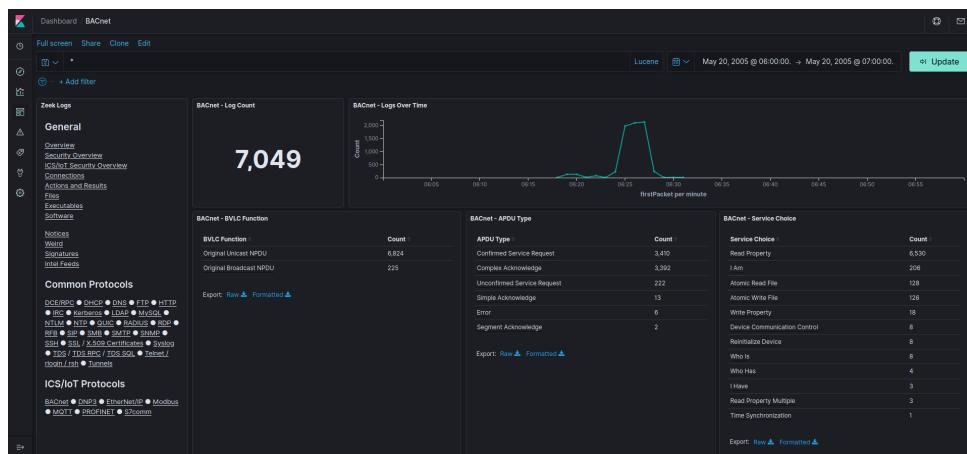
Where possible, Malcolm correlates common fields from across different protocols to allow you to view one device’s or application’s network traffic in the context of the other traffic occurring around it.

A good example of this is the Actions and Results dashboard, in which actions (such as “a file was written,” “a logon was attempted,” “a web page was requested”) and “results” (“success,” “access denied,” “page not found”) can be inspected together regardless of protocol.

For example, multiple failed HTTP authentication attempts, followed by a successful authenticated HTTP POST operation followed by successful reads and writes to a file server could indicate that a foothold was obtained in an HTTP server that allowed an adversary to pivot to another service in the network.

Kibana: ICS protocol dashboards

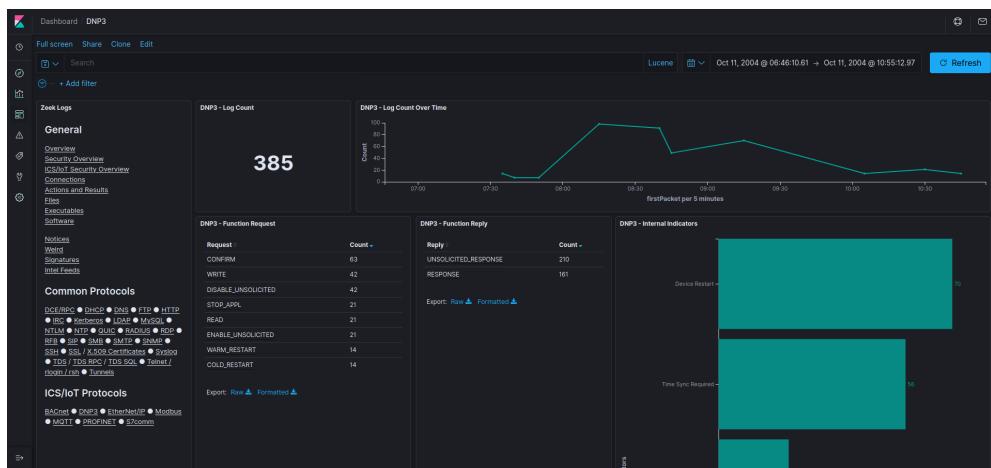
BACnet



In addition to the overview dashboards, Malcolm provides dozens of dashboards tailored to specific application protocols, including protocols commonly used in industrial control systems networks.

Kibana: ICS protocol dashboards

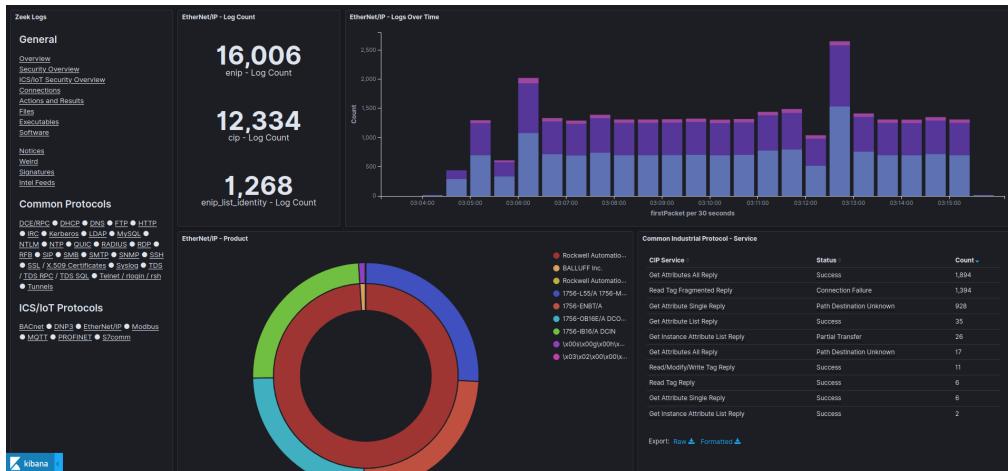
DNP3



14

Kibana: ICS protocol dashboards

EtherNet/IP



15

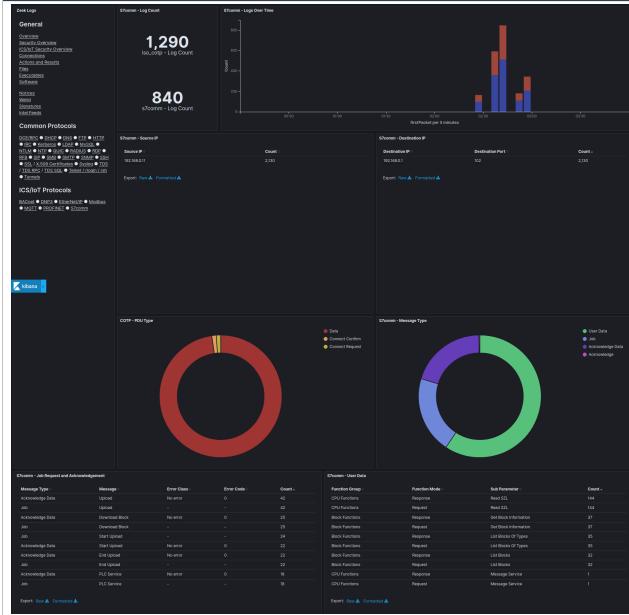
Kibana: ICS protocol dashboards



Modbus

16

Kibana: ICS protocol dashboards



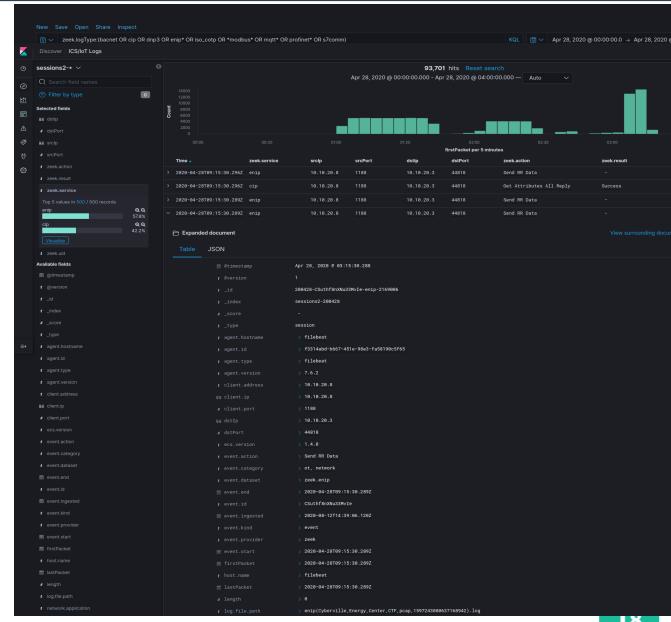
S7comm

17

Kibana

- **Discover**

- Field-level details of logs matching filter criteria
- Create and view saved searches
- View other events just before and after an event



The Discover view enables you to view events on a record-by-record basis, similar to a session record in Moloch, which we'll discuss in a moment, or to an individual line from a Zeek log.

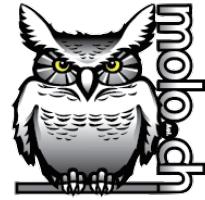
The data table in the Discover view can be customized to display only the fields relevant to the traffic you're interested in: for example, a “play-by-play” of an HTTP session could be reviewed by filtering on `zeek.logType: http`, sorting by Time and including the following fields in the table:

- `srcIp`
- `zeek_http.user_agent`
- `zeek_http.referrer`
- `dstIp`
- `zeek_http.host`
- `zeek_http.uri`
- `zeek_http.status_msg`

This configuration could be stored as a saved search and returned to for future investigation.

Moloch

- <https://localhost/>
- Front end for **both** enriched Zeek logs and Moloch sessions
- Filter by Zeek logs or Moloch sessions; or, view both data sources together
- “Wireshark at scale”: full PCAP availability for viewing packet payloads, exporting filtered and joined PCAP sessions and running deep-packet searches



19

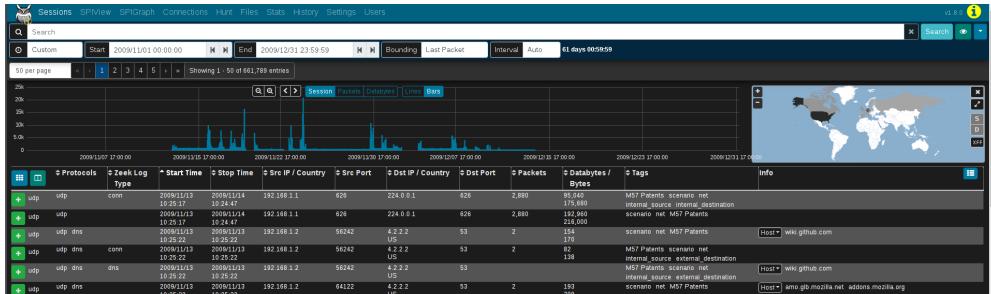
While Kibana is great for “at-a-glance” views and for creating custom visualizations, Moloch provides another interface for examining network traffic that may be better suited to in-depth analysis and network forensics.

Earlier when we talked about the Malcolm PCAP processing pipeline, we mentioned two metadata representations of the same network traffic: the logs generated by Zeek and the session records generated by moloch-capture. While Malcolm’s Kibana dashboards are focused on the Zeek logs, its instance of Moloch can be used to view **both** Zeek logs and Moloch sessions together in the same interface.

Another strength of Moloch is its ability to tie the session metadata back to the original packets’ payloads, allowing you to view, search and export the data deeper in the PCAP that may not be referenced in the metadata. Moloch is able to efficiently deal with very large PCAP file sets, something that Wireshark struggles to do.

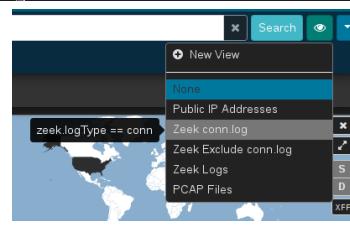
Moloch: Sessions

- Field-level details of sessions/logs matching filters



- Views: overlay saved filters on current search

- E.g., show Moloch sessions vs. Zeek logs



20

Moloch's Sessions tab provides low-level details of the sessions being investigated (similar to Kibana's Discover interface), whether they be Moloch sessions created from PCAP files or Zeek logs mapped to the Moloch session database schema.

The Sessions interface has various controls for applying time, geographic and field value filters to narrow the set of matching sessions.

The set of fields present in the sessions table can be customized, saved and later recalled. In addition, Moloch's views (indicated by the eyeball icon) allow overlaying additional previously-specified filters onto the current sessions filters. For convenience, Malcolm provides several Moloch preconfigured views including several on the zeek . logType field.

Moloch: Packet payloads

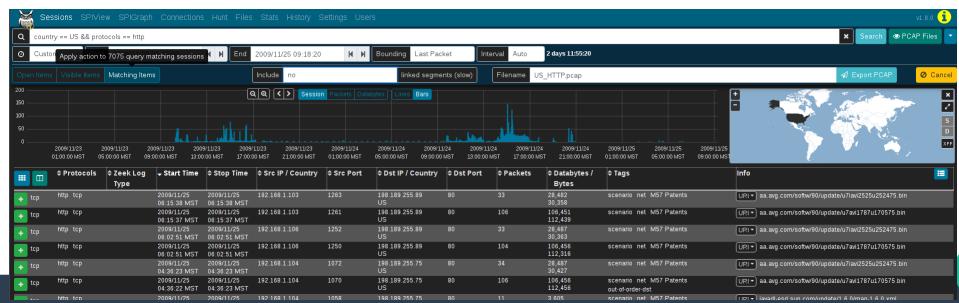
- Displayed for Moloch sessions with full PCAP
- File carving on the fly
- Download session PCAP
- Examine payload with [CyberChef](#)

The screenshot shows the Moloch web interface with a dark theme. At the top, there's a navigation bar with links for Sessions, SPView, SPGraph, Connections, Hunt, Files, Stats, History, Settings, and Users. Below the navigation is a search bar with a placeholder 'q' and a query 'k.src == 10.10.10.3 && protocol == http & bytes > 10000'. The main area displays session details for a session starting at 2020/04/28 00:00:00 and ending at 2020/04/28 04:00:00. It shows 'Showing 1 - 50 of 63 entries' and a table with columns for 'Server Version', 'Body MD5', and 'State content type'. The table lists several entries, each with a 'Decompress' button. Below the table, there's a 'Destination' section showing a single entry for 'HTTP/1.1 200 OK'. At the bottom, there's a 'Packets' section with a table showing 200 rows, and a 'File Bytes' section with a histogram.

As mentioned, Moloch's ability to tie a session record back to its original packet(s) is one of its greatest strengths. Details for individual sessions/logs can be expanded by clicking the plus + icon on the left of each row. For Moloch session records, an additional *Packets* section will be visible underneath the metadata sections. When the details of a session of this type are expanded, Moloch will read the packet(s) comprising the session for display here. Various controls can be used to adjust how the packet is displayed (enabling natural decoding and enabling *Show Images & Files* may produce visually pleasing results), and other options (including PCAP download, carving images and files, applying decoding filters, and examining payloads in CyberChef) are available.

Moloch: PCAP Export

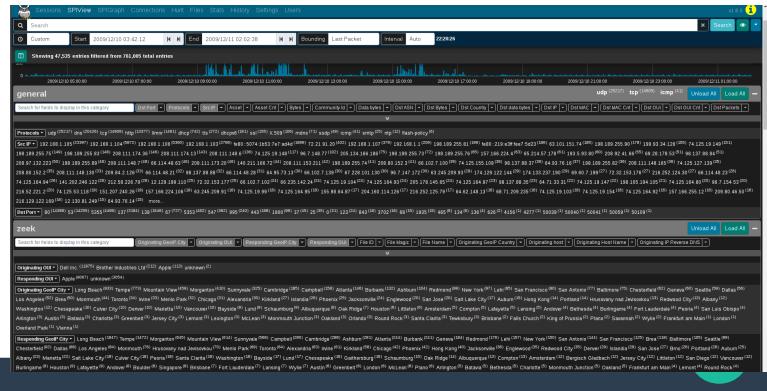
- Creates a new PCAP file from filtered sessions
- Include open, visible or all matching sessions
- Apply “PCAP Files” view to sessions first
- Narrow as much as possible prior to exporting (huge PCAP files are a pain)



Clicking the down arrow ▼ icon to the far right of the search bar presents a list of actions including *PCAP Export*. When full PCAP sessions are displayed, the *PCAP Export* feature allows you to create a new PCAP file from the matching Moloch sessions, including controls for which sessions are included (open items, visible items, or all matching items) and whether or not to include linked segments. Click *Export PCAP* button to generate the PCAP, after which you'll be presented with a browser download dialog to save or open the file. Note that depending on the scope of the filters specified this might take a long time (or, possibly even time out).

Moloch: SPIView

- Explore “top n ” and field cardinality for all fields of both Moloch sessions and Zeek logs
- Apply filters or pivot to Sessions or SPIGraph view for field values of interest
- Limit search to ≤ 1 week before using as it runs many queries



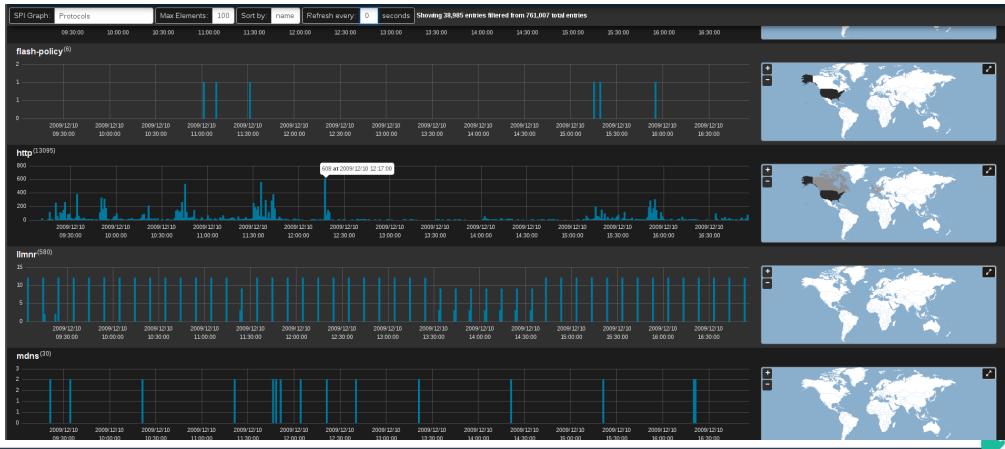
Moloch's SPI (Session Profile Information) View provides a quick and easy-to-use interface for exploring session/log metrics. The SPIView page lists categories for general session metrics (e.g., protocol, source and destination IP addresses, sort and destination ports, etc.) as well as for all of various types of network understood by Moloch and Zeek. These categories can be expanded and the top n values displayed, along with each value's cardinality, for the fields of interest they contain.

Click the the plus + icon to the right of a category to expand it. The values for specific fields are displayed by clicking the field description in the field list underneath the category name. The list of field names can be filtered by typing part of the field name in the *Search for fields* dialog to display in this category text input. The *Load All* and *Unload All* buttons can be used to toggle display of all of the fields belonging to that category. Once displayed, a field's name or one of its values may be clicked to provide further actions for filtering or displaying that field or its values. Of particular interest may be the *Open [fieldname] SPI Graph* option when clicking on a field's name. This will open a new tab with the SPI Graph populated with the field's top values.

Note that because the SPIView page can potentially run many queries, SPIView limits the search domain to seven days (in other words, seven indices, as each index represents one day's worth of data). When using SPIView, you will have best results if you limit your search time frame to less than or equal to seven days.

Moloch: SPIGraph

- View “top n” field values chronologically and geographically
- Identify trends and patterns in network traffic



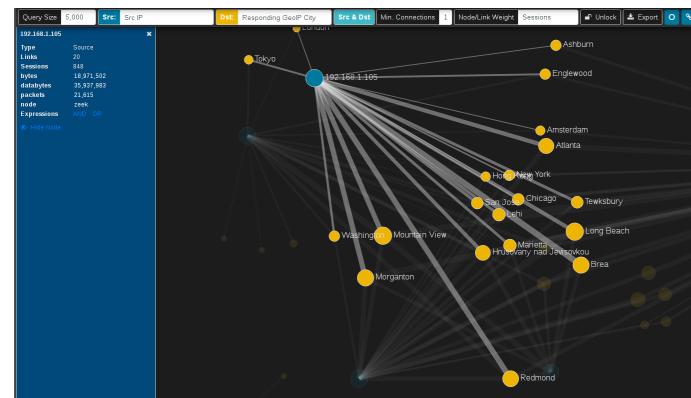
24

Moloch's SPI (Session Profile Information) Graph visualizes the occurrence of some field's top n values over time, and (optionally) geographically. This is particularly useful for identifying trends in a particular type of communication over time: traffic using a particular protocol when seen sparsely at regular intervals on that protocol's date histogram in the SPIGraph may indicate a connection check, polling, or beaconing (for example, see the llmnr protocol in the screenshot).

Controls can be found underneath the time bounding controls for selecting the field of interest, the number of elements to be displayed, the sort order, and a periodic refresh of the data.

Moloch: Connections

- Visualize logical relationship between hosts
- Use any combination of fields for source and dest. nodes
- Compare current vs. baseline traffic



25

The Connections page presents network communications via a force-directed graph, making it easy to visualize logical relationships between network hosts. Controls are available for specifying the query size (where smaller values will execute more quickly but may only contain an incomplete representation of the top n sessions, and larger values may take longer to execute but will be more complete), which fields to use as the source and destination for node values, a minimum connections threshold, and the method for determining the "weight" of the link between two nodes. As is the case with most other visualizations in Moloch, the graph is interactive: clicking on a node or the link between two nodes can be used to modify query filters, and the nodes themselves may be repositioned by dragging and dropping them. A node's color indicates whether it communicated as a source/originator, a destination/responder, or both.

While the default source and destination fields are Src IP and Dst IP:Dst Port, the Connections view is able to use any combination of any of the fields populated by Moloch and Zeek. For example:

- Src OUI and Dst OUI (hardware manufacturers)
- Src IP and Protocols
- Originating Network Segment and Responding Network Segment
- Originating GeoIP City and Responding GeoIP City

or any other combination of these or other fields.

A recent addition to this feature (and one developed specifically for Malcolm and then contributed back upstream to the Moloch project) is the ability to specify a "baseline" time frame in the Connections view to visualize changes to a network over time (e.g., new hosts or protocols appearing in your network).

Moloch: Hunt

- Deep-packet search (“PCAP grep”) of session payloads
- Search for ASCII, hex codes or regular expression matches
- Apply “PCAP Files” view to sessions first

The screenshot shows the Moloch interface with the 'Hunt' tab selected. At the top, there's a search bar with the query 'protocols == http'. Below it, a search configuration panel shows 'All (careful)' selected, 'Start' set to '1969/12/31 17:00:00', and 'End' set to '2019/05/28 09:19:44'. A progress bar indicates the search is '62.3%' complete, having found 297 matches. The main area displays a table titled 'Hunt Job Queue' with one row:

Status	Matches	Name	User	Search text	Notify	Created	ID
X	297	HTTP with password	analyst	password (ascii)		2019/05/28 09:20:28	VRgH_20B-8T3HMe4R6WA

Below the table, a message states: 'This hunt is running. This hunt was last updated at: 2019/05/28 09:21:06. Examining 50 raw source and destination packets per session. Found 297 of 27,299 searched sessions out of 43818 total sessions to search. The sessions query expression was: protocols == http. The sessions query time range was from 1969/12/31 17:00:00 to 2019/05/28 09:19:44.'

26

Moloch's Hunt feature allows an analyst to search within the packets themselves (including payload data) rather than simply searching the session metadata. The search string may be specified using ASCII (with or without case sensitivity), hex codes, or regular expressions. Once a hunt job is complete, matching sessions can be viewed in the Sessions view.

Clicking *Create a packet search job* on the Hunt page will allow you to specify the following parameters for a new hunt job:

- a packet search job name
- a maximum number of packets to examine per session
- the search string and its format (ascii, ascii (case sensitive), hex, regex, or hex regex)
- whether to search source packets, destination packets, or both
- whether to search raw or reassembled packets

Click the **+** *Create* button to begin the search. Moloch will scan the source PCAP files from which the sessions were created according to the search criteria. Note that whatever filters were specified when the hunt job is executed will apply to the hunt job as well; the number of sessions matching the current filters will be displayed above the hunt job parameters with text like "① Creating a new packet search job will search the packets of # sessions."

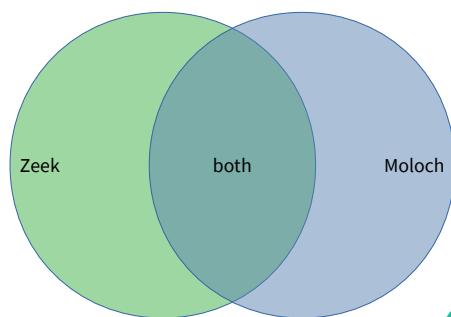
Once a hunt job is submitted, it will be assigned a unique hunt ID (a long unique string of characters like yuBHAGsBdljYmwGkbEMm) and its progress will be updated periodically in the Hunt Job Queue. More details for the hunt job can be viewed by expanding its row with the plus **+** icon on the left.

When the hunt job is complete (and a minute or so has passed, as the hunt Id must be added to the matching session records in the database), click the folder icon on the right side of the hunt job row to open a new Sessions tab with the search bar prepopulated to filter for sessions with packets matching the search criteria. From this list of filtered sessions you can expand session details and explore packet payloads which matched the hunt search criteria.

The hunt feature is available only for sessions created from full packet capture data, not Zeek logs. This being the case, it is a good idea to click the eyeball **👁** icon and select the *PCAP Files* view to exclude Zeek logs from candidate sessions prior to using the hunt feature.

Using Moloch and Kibana together in Malcolm

- Search syntax is different between Moloch and Kibana (and in some cases, so are [field names](#)): see [Malcolm documentation](#) for examples and [Moloch help](#)
- Despite considerable overlap, there are differences in [protocol parser support](#) between Zeek and Moloch
 - Learning the strengths of each will help you more effectively find the good stuff



27

Although Malcolm's Kibana and Moloch interfaces provide two different views into the same network data, there are a few notable differences between the two:

- The Moloch and Kibana [search syntax is different](#).
- Moloch uses its own field names in its user interface: for example, searching `protocols == http` in Moloch is equivalent to searching `protocol:http` in Kibana. Enabling *Display Database Fields* in the *Fields* section of Moloch's help can help you map them.
- Despite considerable overlap, there are differences in [protocol parser support](#) between Zeek and Moloch. In particular, Malcolm's configuration of Zeek parses many more ICS protocols than Moloch.

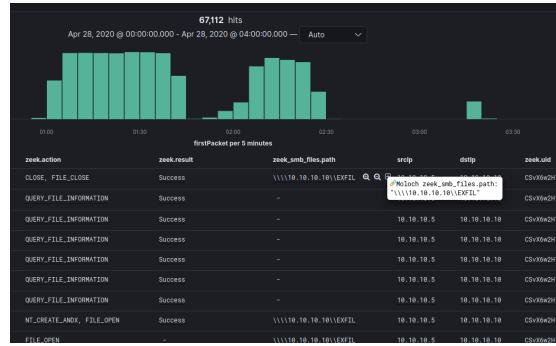
Pivots: Moloch ↔ Kibana

- Moloch → Kibana

The screenshot shows the Moloch interface with the following details:

- Hosts: 10.10.10.10
- Shares: EXFIL
- Files: \alarms.png
- Zeek Common Fields:
 - Zeek Connection ID: CSyX6wZH1DADDnC3k8
 - Zeek Log Type: smb_cmd smb_files
 - Zeek Node: filebeat
 - Originating Host: 10.10.10.5
 - Originating Host Name: Historian
 - Originating Network: Battery Network
 - Responding Host: 10.10.10.10
 - Responding Network: Battery Network
 - and Battery Network and not Battery Network or Battery Network or not Battery Network or not Battery Network
 - Originating Port: 445
 - Responding Port: 445
 - Protocol: TCP
 - Service: Kibana
 - Service Version: 7.6.2
 - Action: Copy value
 - Result: Success

- Kibana → Moloch



Check query search time frame after pivot:
sometimes it gets lost in translation.

28

Malcolm makes it easy to pivot between Moloch and Kibana.

To pivot from Moloch to Kibana, click on a field's value to open its drop-down menu and choose *Kibana [field name] [field value]*. A new browser tab should open to Kibana's Discover interface with that field value filter already applied.

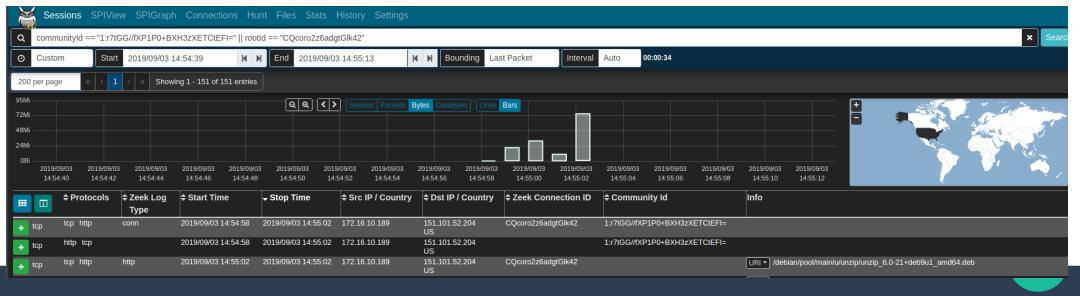
To pivot from Kibana to Moloch, click the or (*Details*) icon next to a field's value in a Kibana table component, then select *Moloch [field name]: “[field value]”*. A new browser tab should open to Moloch's Sessions tab with the search filter applied.

Note that sometimes, especially when pivoting from Kibana to Moloch, the search time frame is not preserved and will have to manually be adjusted in the new tab.

Correlating Moloch sessions and Zeek logs

- Correlate Zeek logs and Moloch sessions using common fields
- `communityId` fingerprints flows in both and can bridge the two
- `zeek.uid/rootId` filters Zeek logs for the same session
- Filter community ID OR'ed with zeek UID to see all Moloch sessions and Zeek logs for the same traffic

```
- communityId == "1:r7tGG//fXP1P0+BXH3zXETCtEFI=" || rootId == "CQcoro2z6adgtGlk42"
```



As previously discussed, Moloch generates session records containing metadata about network connections. Zeek generates similar session metadata, linking network events to sessions via a connection UID. Malcolm aims to facilitate analysis of Zeek logs by mapping values from Zeek logs to the Moloch session database schema for equivalent fields, and by creating new "native" Moloch database fields for all the other Zeek log values for which there is not currently an equivalent in Moloch. The *Fields* section of Moloch's help provides a list of known fields across both the Moloch and Zeek data sources.

In this way, when full packet capture is an option, analysis of PCAP files can be enhanced by the additional information Zeek provides. When full packet capture is not an option, similar analysis can still be performed using the same interfaces and processes using the Zeek logs alone. The values of records created from Zeek logs can be expanded and viewed like any native Moloch session by clicking the plus + icon to the left of the record in the Sessions view. However, note that when dealing with these Zeek records the full packet contents are not available, so buttons dealing with viewing and exporting PCAP information will not behave as they would for records from PCAP files. Other than that, Zeek records and their values are usable in Malcolm just like native PCAP session records.

A few fields of particular mention that help limit returned results to those Zeek logs and Moloch session records generated from the same network connection are Community ID and Zeek's connection UID (`zeek.uid`), which Malcolm maps to Moloch's `rootId` field.

Community ID is a specification for standard flow hashing published by Corelight with the intent of making it easier to pivot from one dataset (e.g., Moloch sessions) to another (e.g., Zeek conn.log entries). In Malcolm both Moloch and Zeek populate this value, which makes it possible to filter for a specific network connection and see both data sources' results for that connection.

The `rootId` field is used by Moloch to link session records together when a particular session has too many packets to be represented by a single session. When normalizing Zeek logs to Moloch's schema, Malcolm piggybacks on `rootId` to store Zeek's connection UID to crossreference entries across Zeek log types. The connection UID is also stored in `zeek.uid`.

Filtering on community ID OR'ed with zeek UID (e.g., `communityId == "1:r7tGG//fXP1P0+BXH3zXETCtEFI=" || rootId == "CQcoro2z6adgtGlk42"`) is an effective way to see both the Moloch sessions and Zeek logs generated by a particular network connection.

Search tips

- Always check your search time frame
- “Zoom in” (apply filters) for a particular field value, pivot to another field then “zoom out” (remove filters)
- Most UI controls can work with any data field (1000+)
- Filter on zeek . logType (e.g., conn to see conn.log)
- Filter on protocol or both Moloch and Zeek regardless of data source (e.g., protocol:http in Kibana and protocols == http in Moloch)
- tags field
 - Populated for both Moloch sessions and Zeek logs with tags provided on upload and words extracted from filenames
 - internal_source, internal_destination, external_source, external_destination, cross_segment

30

Finally, here are a few tips for effective searching in Kibana and Moloch:

- Always check your search time frame. If you’re not seeing the data you’re expecting to see, often it’s because the data lies outside of the window of time you’re searching.
- An effective technique for investigating is to “zoom in” (for example, narrowing in on a particular file type transferred), pivot to another field (select the source IP address that transferred the file) then “zoom out” (removing the file type filter to see what other activity that source IP was involved in).
- Most elements in both Kibana and Moloch are interactive and can be configured to work with any of the more than 1000+ data fields Malcolm knows about.
- Learn how to filter on common fields like Zeek log type and network application protocol.
- Utilize the tags field in your searches, including prepopulated tags like based on public/private IP address space, tags populated based on the host and segment mapping you’ve configured and tags populated based on PCAP filenames.

Thanks! Visit [Malcolm on GitHub](#)
to read the docs, make
suggestions, report issues and
★Star to show your support!



Malcolm is Copyright © 2020 Battelle Energy Alliance, LLC, and is developed and released through the cooperation of the Cybersecurity and Infrastructure Security Agency of the US Department of Homeland Security.