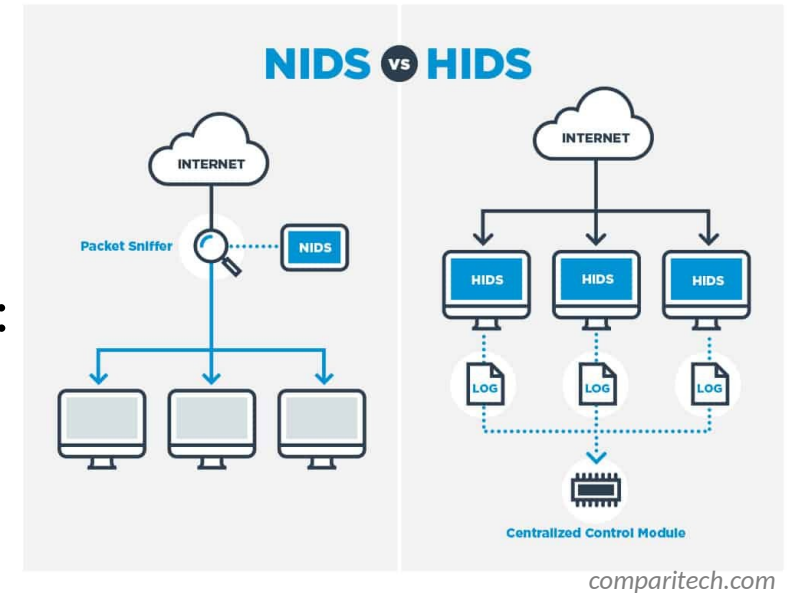


# Network Traffic Analysis with Malcolm

Seth Grover, Malcolm developer • Cybersecurity R&D • Idaho National Lab

# Intrusion Detection Systems

- HIDS: Host Intrusion Detection Systems
  - Agents run on individual hosts or devices on a network
  - Not what we're talking about today
- NIDS: Network Intrusion Detection Systems
  - Monitor and analyze network traffic for anomalies: suspicious activity, policy violations, etc.
  - Generally passive/out-of-band; otherwise it's an Intrusion Prevention System
  - Detection methods
    - Signature-based detection
    - Statistical anomaly-based detection
    - Stateful protocol analysis detection





# IDS: Types of Attacks

- Scanning Attack
  - Determine network topology
  - IDS highlights connections from one host to many other hosts in the network, or connection attempts to sequential IP addresses and/or ports
- Denial of Service Attack
  - Interrupt service by flooding requests or flaws in protocol implementations
  - IDS identifies large volume of traffic from or to a particular host or invalid connection states (e.g., TCP SYN/ACK with no ACK)
- Penetration Attack
  - Gain access to system resources by exploiting a software or configuration flaw
  - Trickier, but IDS may detect vulnerable software versions or simply alert on unusual operations (e.g., a “write” operation in an already-configured environment with mostly “read” operations)





- Extensible, open-source passive network analysis framework
- More than just an Intrusion Detection System:
  - Packet capture (like **TCPDUMP**)
  - Traffic inspection (like  **Wireshark**)
  - Intrusion detection (like **SNORT**)
  - Log recording (like NetFlow and syslog)
  - Scripting framework (like  **python**)



## Strengths

- Analyzes both link-layer and application-layer behavior
- Content extraction
- Behavioral analysis
- Session correlation
- Can add support for uncommon protocols through scripts/plugins

## Weaknesses

- Session metadata only (not full payload)
- Setup and configuration can be complicated
- Produces flat textual log files which can be unwieldy for in-depth analysis

- Network Protocols
- Files
- Detection
- Network Observations

Field	Type	Description
to	code	Destination of the HTTP request
url & url	string	Underlying connection info. <a href="#">See client.log</a>
bytes, length	count	Expected length into the connection
method	string	HTTP Request verb: GET, POST, etc.
host	string	Name of the host header
uri	string	URI used in the request
referer	string	Name of the "Referer" header
user-agent	string	Name of the user-agent header
content-length, len	count	Uncompressed content size of body data
content-length, len	count	Uncompressed content size of body data
status, code	count	Status code returned by the server
status, code	string	Status message returned by the server
url, code	count	Last seen file URL, reply code by server
url, code	string	Last seen file URL, reply message by server
log	string	Indicates if various attributes discovered
username	string	Username if basic-auth is performed
password	string	Password if basic-auth is performed
request	string	Headers initiated of a successful request
url, bytes	count	The unique Doxygen ID
url, filename	string	The filename from ID
url, mime, type	string	The mime from ID
url, hash	count	The unique Doxygen ID
url, filename	string	The filename from ID
url, mime, type	string	The mime from ID
client, header	string	The names of HTTP headers sent by client
server, header	string	The names of HTTP headers sent by server
cookie, url	string	Variable names extracted from cookies
url, url	string	Variable names extracted from the URI

If [pretty-print](#) or [http-raw](#) is used:

If [pretty-print](#) or [http-raw](#) is used:

If [pretty-print](#) or [http-raw](#) is used:

[illegible]

# Network Protocols

- `conn` – Network session tracking
  - Identified by session 4-tuple (originating IP:port, responding IP:port)
  - One session (line in a log file) for every IP connection
  - Unique identifier (UID) ties lines from other logs to a session
- `http`, `modbus`, `ftp`, `dns`, **etc.**
  - Protocol-specific log files created as traffic is seen
  - Contain application-layer metadata about network activities

# Files

- `files` – File analysis results
  - Each transferred file identified with FUID
  - Associated with connection UID(s) over which file was transferred
  - File name, mime type, file size, etc. provided when available
- `pe` – Analysis of Portable Executable (PE) files
  - Target platform, architecture, OS, etc. for executables transferred across the network
- `x509` – Analysis of X.509 public key certificates



# Detection

- `notice` – Zeek concept of “alarms,” notices draw extra attention to an event
  - `Conn::Content_Gap`, `DNS::External_Name`,  
`FTP::Bruteforcing`, `Heartbleed::SSL_Heartbeat_Attack`,  
`HTTP::SQL_Injection_Attacker`, `Scan::Address_Scan`,  
`Scan::Port_Scan`, `Software::Vulnerable_Version`,  
`SSH::Password_Guessing`, `SSL::Certificate_Expired`,  
`Weird::Activity`, ...
  - <https://docs.zeek.org/en/stable/zeek-noticeindex.html>

# Detection (cont.)

- `weird` – Unexpected network-level activity
  - > 150 weirdness indicators across many protocols
  - <https://docs.zeeb.org/en/stable/scripts/base/frameworks/notice/weird.zeeb.html#id1>
- `signatures` – Signature matches, including hits from enabled carved file scanners like ClamAV, YARA and capa

# Network Observations

- Periodic dump of entities seen over the last day
  - `known_certs` – SSL certificates
  - `known_devices` – MAC addresses
  - `known_hosts` – Hosts with TCP handshakes
  - `known_modbus` – Modbus masters and slaves
  - `known_services` – Services (TCP “servers”)
  - `software` – Software being used on the network (e.g., Apache, OpenSSH, etc.)
    - Could be used for identifying vulnerable versions of software or firmware



# Arkime

## Strengths

- Large scale index packet capture and search tool
- Packet analysis engine with support for many common IT protocols
- Web interface for browsing, searching, analysis and PCAP carving for exporting
- PCAP payloads (not just session header/metadata) are viewable and searchable

## Weaknesses

- No OT protocol support
- Adding new protocol parsers requires C programming

# Malcolm

A powerful open-source network traffic analysis tool suite.

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



## Streamlined deployment

- Suitable for field use (hunt or incident response) or SOC deployment. Runs in Docker on Linux, macOS and Windows platforms. Provides easy-to-use web-based user interfaces.

## Industry-standard tools

- Uses Arkime and Zeek for network traffic capture, Logstash for parsing and enrichment, OpenSearch for indexing and Dashboards and Arkime Viewer for visualization. Also leverages OpenSearch Anomaly Detection, YARA, capa, ClamAV, CyberChef and other proven tools for analysis of traffic and artifacts.

## Expanding control systems visibility

- Analyzes more protocols used in operational technology (OT) networks than other open-source or paid solutions. Ongoing development is focused on increasing the quantity and quality of industrial control systems (ICS) traffic.

## Dedicated sensor appliance

- Includes Hedgehog Linux, a hardened Linux distribution for capturing network traffic and forwarding its metadata to Malcolm.

# Malcolm

## Components

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



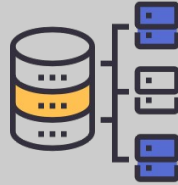
Capture



File Scanning



Forwarding &  
Enrichment



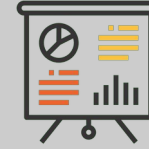
Storage



Anomaly  
Detection



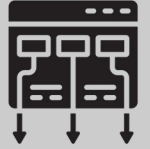
Alerting



Visualization



Payload  
Analysis



Framework



beats



OpenSearch



OpenSearch  
Anomaly  
Detection  
Plugin



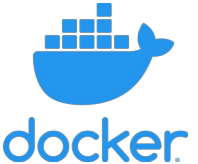
OpenSearch  
Alerting  
Plugin



OpenSearch  
Dashboards



CyberChef



Arkime



ClamAV



netsniff-ng



CAPA



logstash



TCPDUMP



VIRUSTOTAL



Arkime



Arkime  
session PCAP  
export to  
WIRESHARK



NGINX



# Supported Protocols

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

Internet layer

Border Gateway Protocol (BGP)

**Building Automation and Control (BACnet)**

**Bristol Standard Asynchronous Protocol (BSAP)**

Distributed Computing Environment / Remote Procedure Calls  
(DCE/RPC)

Dynamic Host Configuration Protocol (DHCP)

**Distributed Network Protocol 3 (DNP3)**

Domain Name System (DNS)

**EtherCAT**

**EtherNet/IP / Common Industrial Protocol (CIP)**

FTP (File Transfer Protocol)

Google Quick UDP Internet Connections (gQUIC)

Hypertext Transfer Protocol (HTTP)

IPsec

Internet Relay Chat (IRC)

Lightweight Directory Access Protocol (LDAP)

Kerberos

**Modbus**

MQ Telemetry Transport (MQTT)

MySQL

NT Lan Manager (NTLM)

Network Time Protocol (NTP)

Oracle

**Open Platform Communications Unified Architecture  
(OPC UA) Binary**

Open Shortest Path First (OSPF)

OpenVPN

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)**

Secure Shell (SSH)

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

Session Initiation Protocol (SIP)

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

Simple Mail Transfer Protocol (SMTP)

Simple Network Management Protocol (SNMP)

SOCKS

STUN (Session Traversal Utilities for NAT)

Syslog

Tabular Data Stream (TDS)

Telnet / remote shell (rsh) / remote login (rlogin)

TFTP (Trivial File Transfer Protocol)

WireGuard

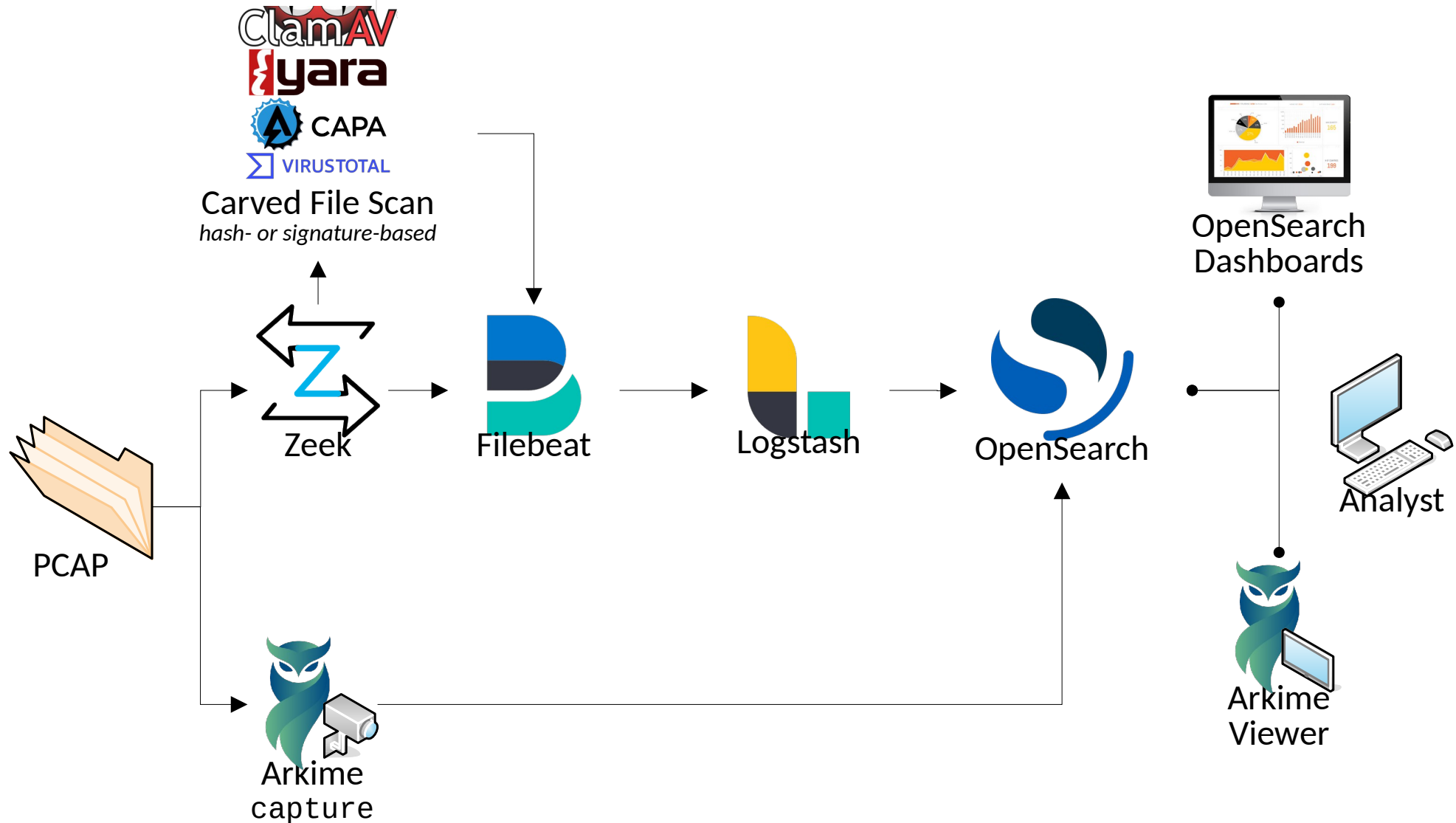
various tunnel protocols (e.g., GTP, GRE, Teredo, AYIYA, IP-in-IP, etc.)

*\* Industrial control systems protocols indicated with **bold***

# Malcolm

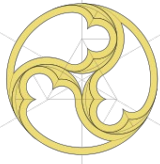
## Data Pipeline

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





# Malcolm



## Data Pipeline

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

Traffic is collected passively by the Hedgehog sensor device

- Zeek and Arkime Capture generate metadata about network communications
- Full PCAP may be stored locally on the sensor
- Files transfers are detected and the files scanned for threats
- PCAP may also be uploaded to or captured by Malcolm without requiring a dedicated sensor

Metadata is securely forwarded to Malcolm

- All communications between the sensor and aggregator are TLS-encrypted
- Sensor data including resource utilization, syslog, audit logs, temperatures and more may also be forwarded

Logs are enriched and stored in OpenSearch

- Lookups are performed for GeoIP, ASN, MAC-to-vendor, community ID, domain name entropy, etc.
- Network events normalized across protocols and data sources
- Best-guess techniques applied for identifying obscure ICS traffic
- Enriched metadata may be forwarded to higher-tiered Malcolm instance

Machine learning algorithms identify anomalies

- Default detectors are provided for action and result, flow size and types of transferred files
- Custom detectors may be created for any aspect of any supported protocol

Alerts are sent over email, webhooks, Slack or Amazon Chime

- Alerts may be triggered by exceeded thresholds, anomalies detected, custom queries, etc.

Traffic is visualized in OpenSearch Dashboards and Arkime Viewer

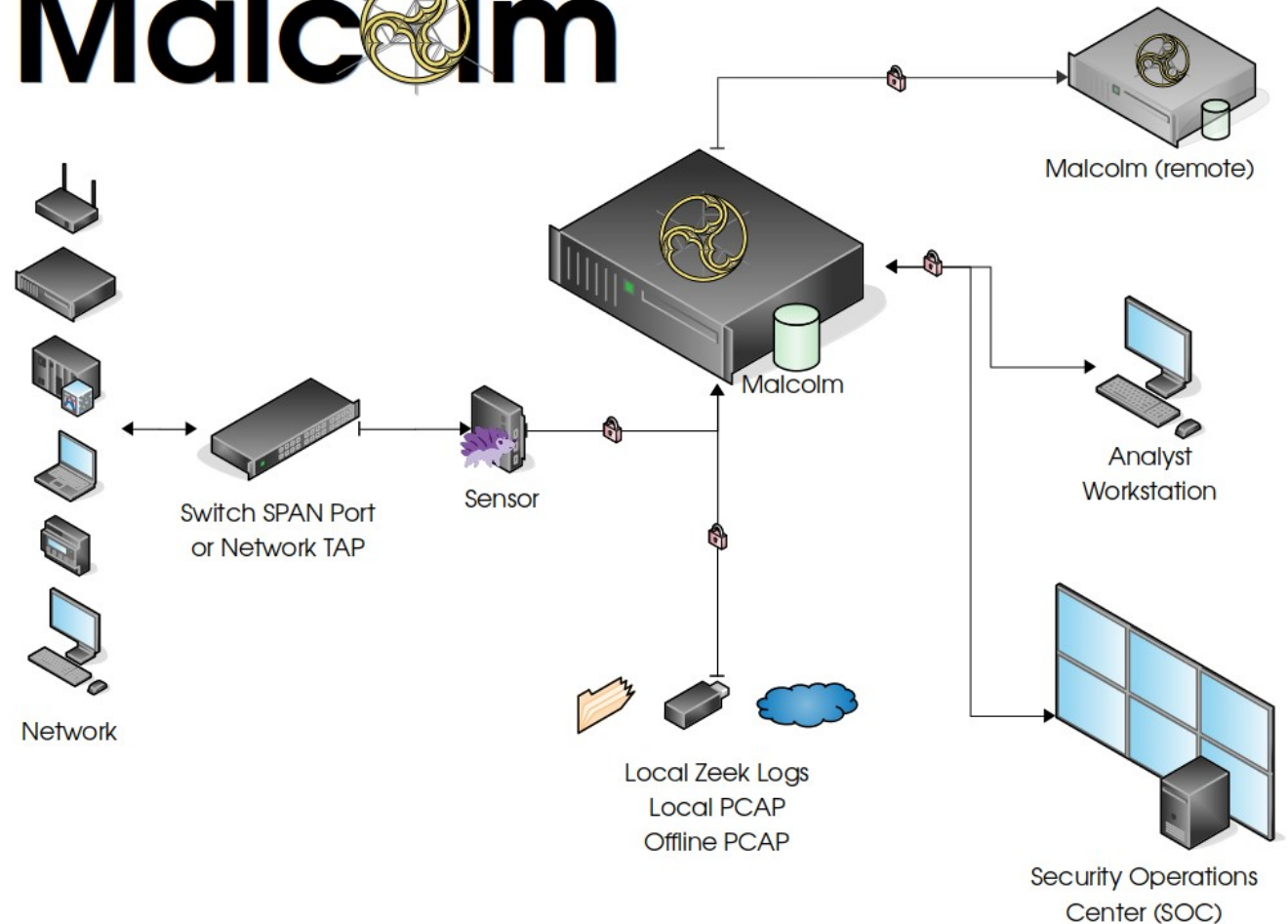
- Dozens of custom dashboards are provided for all supported protocols
- PCAP payloads are retrieved from sensor automatically on demand
- Custom visualizations may be created via drag-and-drop interface
- Malcolm can authenticate users from its own list or via Active Directory / LDAP

# Configuring and Running Malcolm

- Runs natively in Docker or in a Virtual Machine
- 16+GB RAM, 4+ cores, “enough” disk for PCAP and logs suggested
- Documentation and source code on GitHub: [github.com/idaholab/Malcolm](https://github.com/idaholab/Malcolm)
- Walkthroughs on [YouTube](#): search “Malcolm Network Traffic Analysis”



## Malcolm








# Identifying Network Hosts and Subnets

- Assign custom names to network hosts and subnets prior to PCAP import
- Allows identification of cross-segment traffic and name-based search and filter
- Define in text file(s) or via web interface
- <https://localhost/name-map-ui>



The screenshot shows a web interface for mapping network addresses to names. It features a table with columns for Address, Name, and Tag. Each row has edit and delete icons. At the bottom, there are input fields for Address, Name, and Tag (optional), along with a save button.

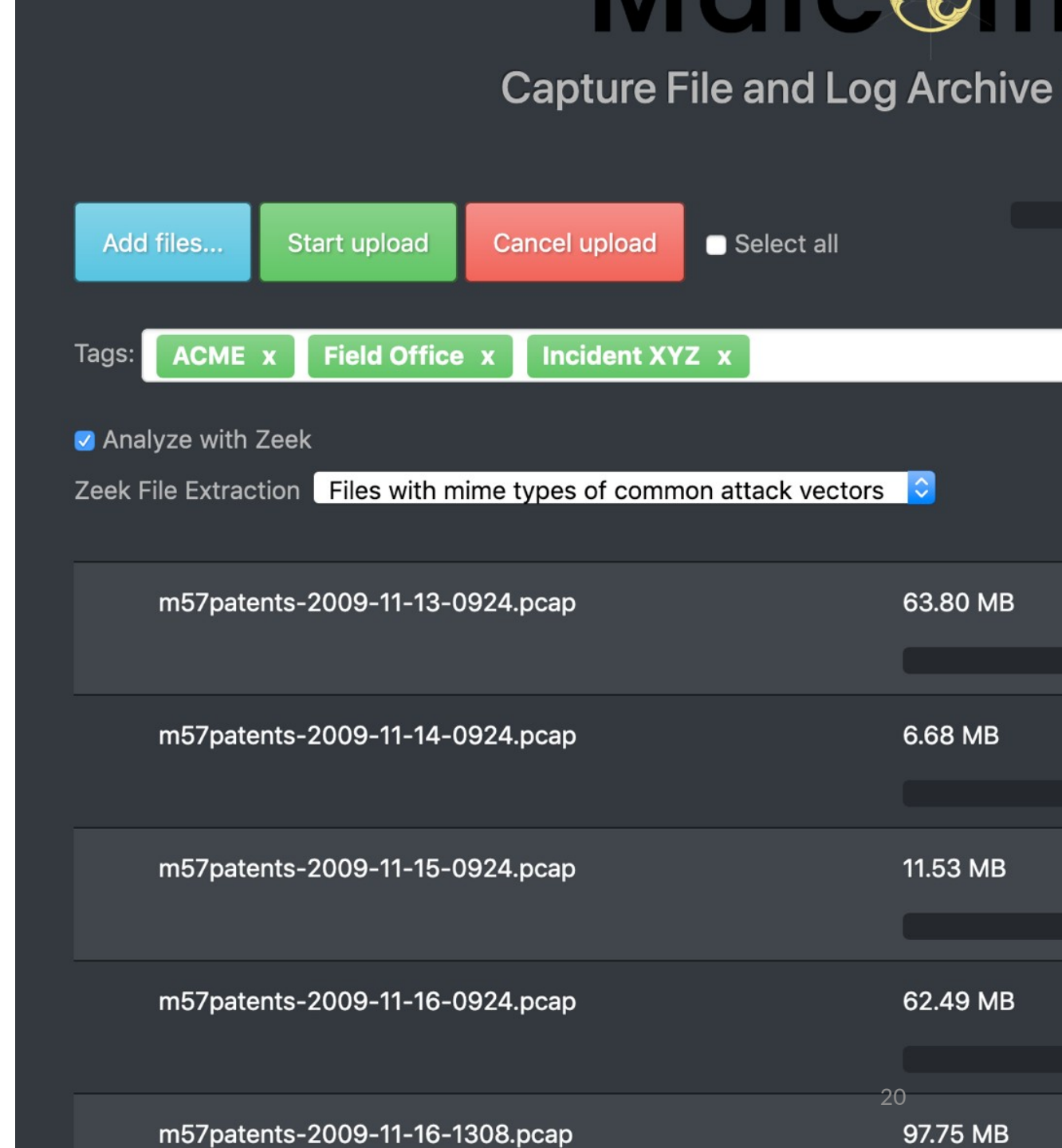
	Address	Name	Tag		
	06:46:0b:a6:16:bf	serial-host.intranet.lan	testbed		
ent	10.0.0.0/8	corporate			
	127.0.0.1	localhost			
	127.0.1.1	localhost			
ent	172.16.0.0/12	virtualized	testbed		
	192.168.10.10	office-laptop.intranet.lan			
ent	192.168.40.0/24	corporate			
ent	192.168.50.0/24	corporate			
ent	192.168.100.0/24	control			
ent	192.168.200.0/24	dmz			
	:::1	localhost			

Search mappings

▼ Address Name Tag (optional) 

# Importing Traffic Captures for Analysis

- Specify tags for search and filter
- Enable Zeek analysis and file extraction
  - Or configure as global default
- Upload PCAP files or archived Zeek logs
  - pcapng not supported yet
- <https://localhost/upload>



# Data Tagging and 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)
- `tags` field
  - Populated for both Arkime sessions and Zeek logs with tags provided on upload and words extracted from PCAP filenames
  - `internal_source`,  
`internal_destination`,  
`external_source`,  
`external_destination`,  
`cross_segment`



- Front end for Zeek logs
- Prebuilt visualizations for all protocols Malcolm parses
- WYSIWYG editors to create custom visualizations and dashboards
- Drill down from high-level trends to specific items of interest
- <https://localhost/dashboards>

Malcolm

Dashboard / Security Overview

### Zeek Logs

#### General

- [Overview](#)
- [Security Overview](#)
- [ICS/IoT Security Overview](#)
- [Severity](#)
- [Connections](#)
- [Actions and Results](#)
- [Files](#)
- [Executables](#)
- [Software](#)
- [Notices](#)
- [Weird](#)
- [Signatures](#)
- [Intel Feeds](#)
- [Arkime](#)

#### Common Protocols

DCE/RPC ● DHCP ● DNS ● FTP / TFTP ● HTTP ● IRC ● Kerberos ● LDAP ● MQTT ● MySQL ● NTLM ● NTP ● OSPF ● QUIC ● RADIUS ● RDP ● RFB ● SIP ● SMB ● SMTP ● SNMP ● SSH ● SSL / X.509 Certificates ● STUN ● Syslog ● TDS / TDS RPC / TDS SQL ● Telnet / rlogin / rsh ● Tunnels

#### ICS/IoT Protocols

### Notices by Category

Notice Category	Count
SSL::Invalid_Server_Cert	50
ATTACK::Execution	27
ATTACK::Lateral_Movement	6
EternalSafety::EternalSynergy	5
ATTACK::Lateral_Movement_Multiple_Attempts	4
Signatures::Sensitive_Signature	4
ATTACK::Lateral_Movement_Extracted_File	2
EternalSafety::EternalChampion	2
EternalSafety::ViolationNtRename	2
EternalSafety::ViolationTx2Cmd	2
ATTACK::Discovery	1
EternalSafety::DoublePulsar	1
FTP::Bruteforcing	1
Ripple20::Treck_TCP_observed	1

Export: [Raw](#) [Formatted](#)

### Outdated/Insecure Application Protocols

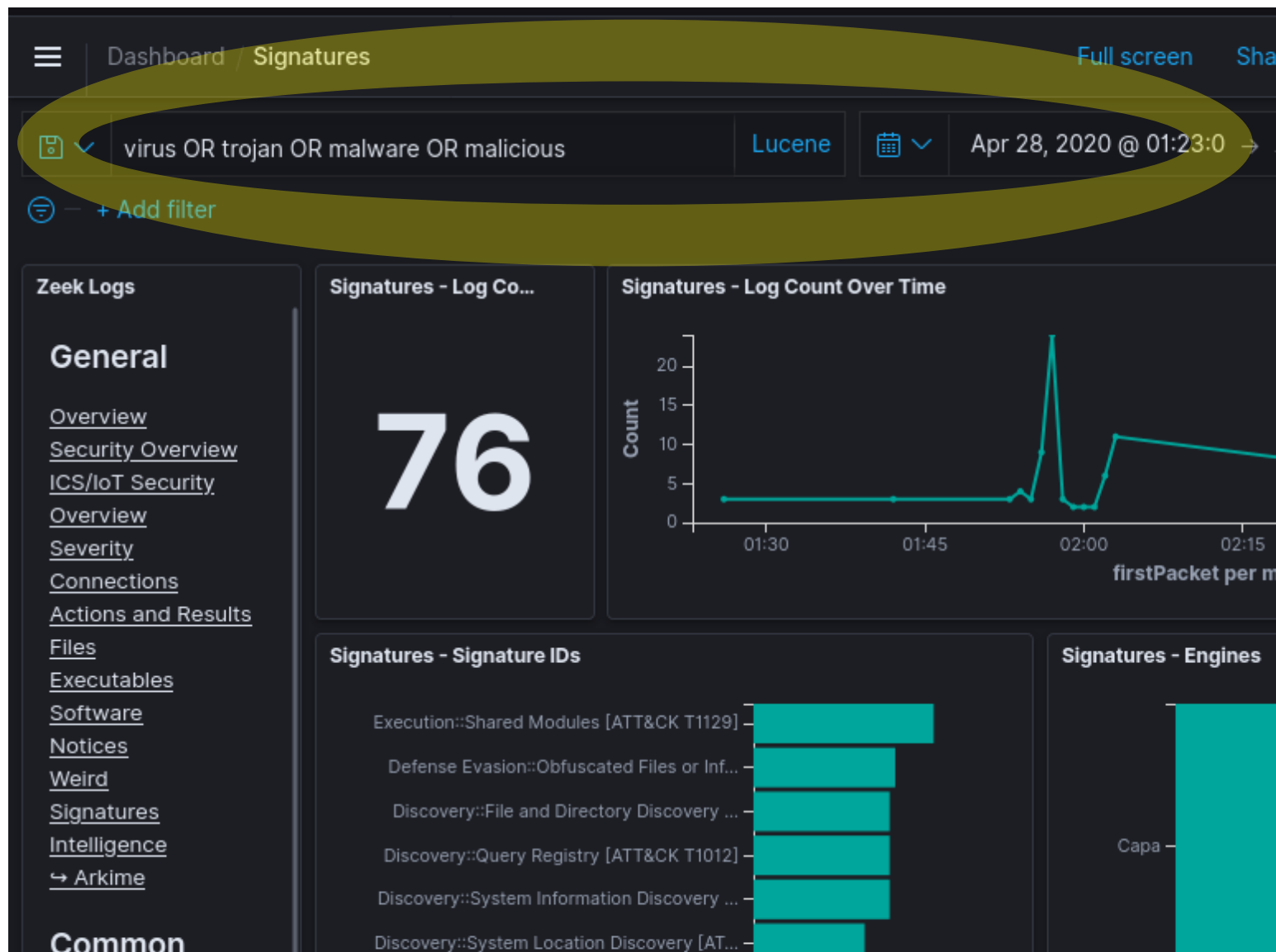
Application Protocol	Protocol Version	Count
ftp	-	1,063
smb	1	535
tftp	-	64
ntp	3	42
tls	TLSv10	38

### Connections by Destination Country



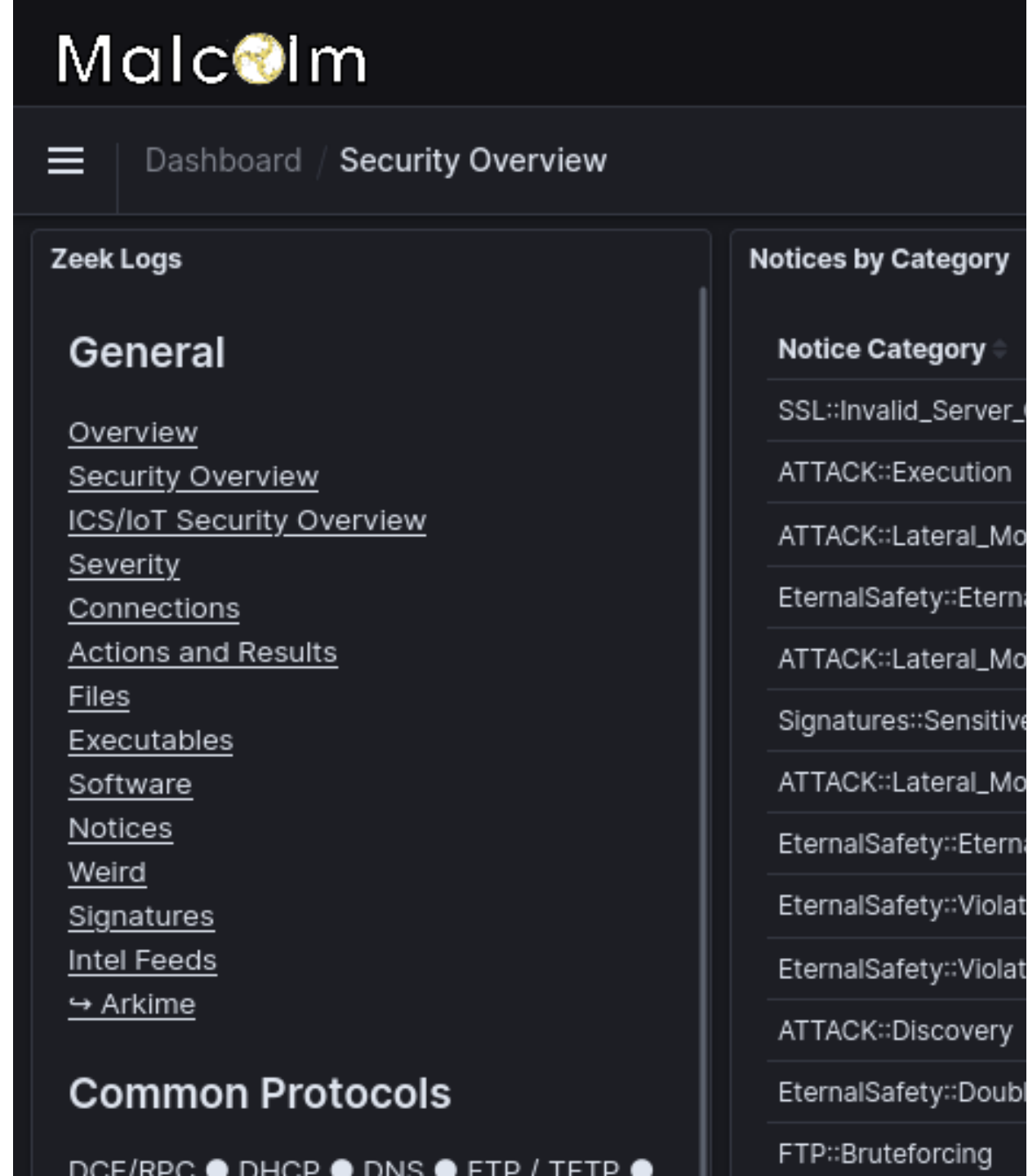
# Dashboards Filters and Search

- Time filter: define search time frame
- Query bar: write queries in Lucene syntax or DQL (Dashboards Query Language)
- Filter bar: define filters using a UI
  - Pin filters as you move across dashboards
- Save queries and filters for reuse



# Overview Dashboards

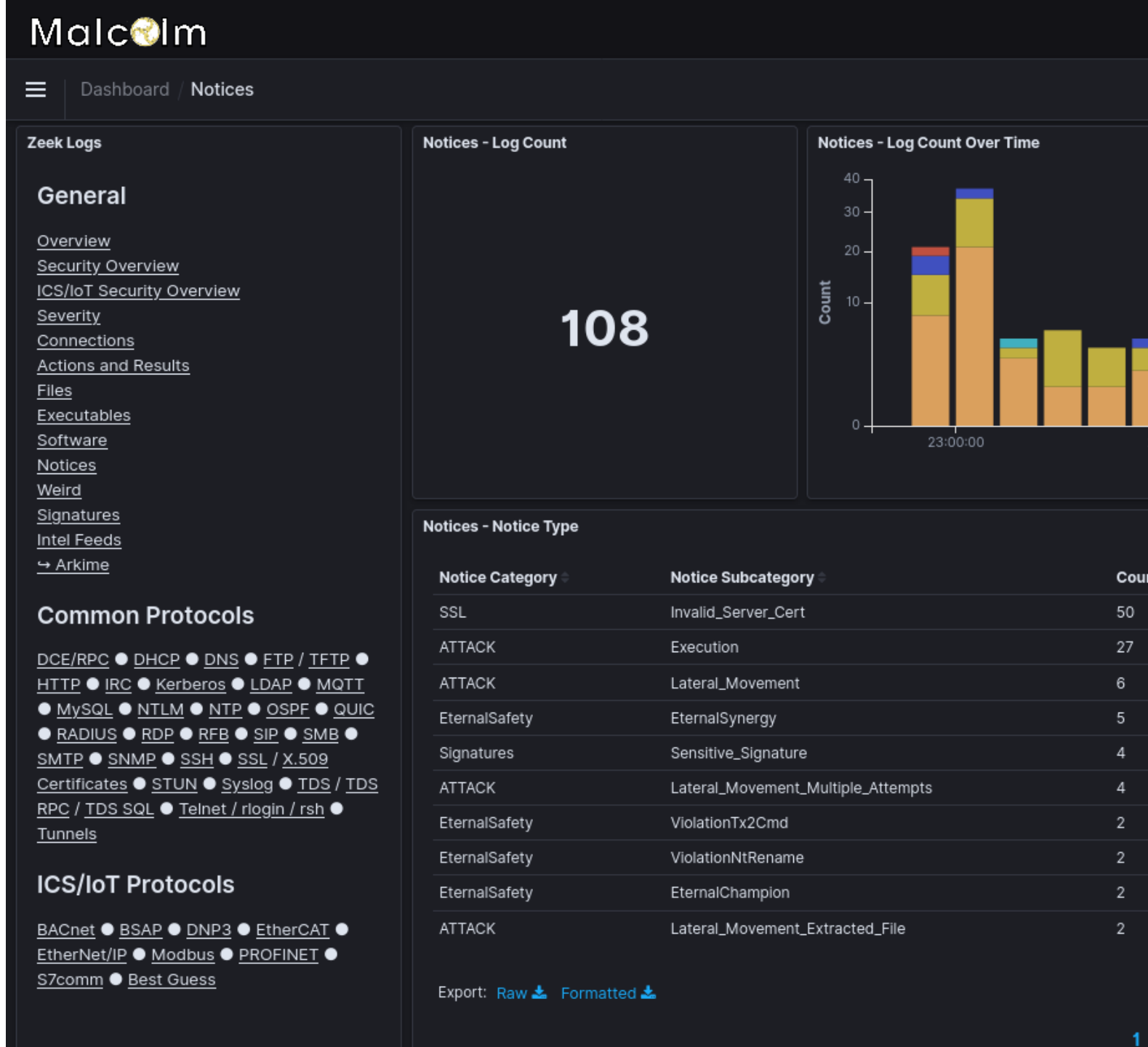
- High-level view of trends, sessions and events
- Populated from logs across all protocols
- Good jumping-off place for investigation





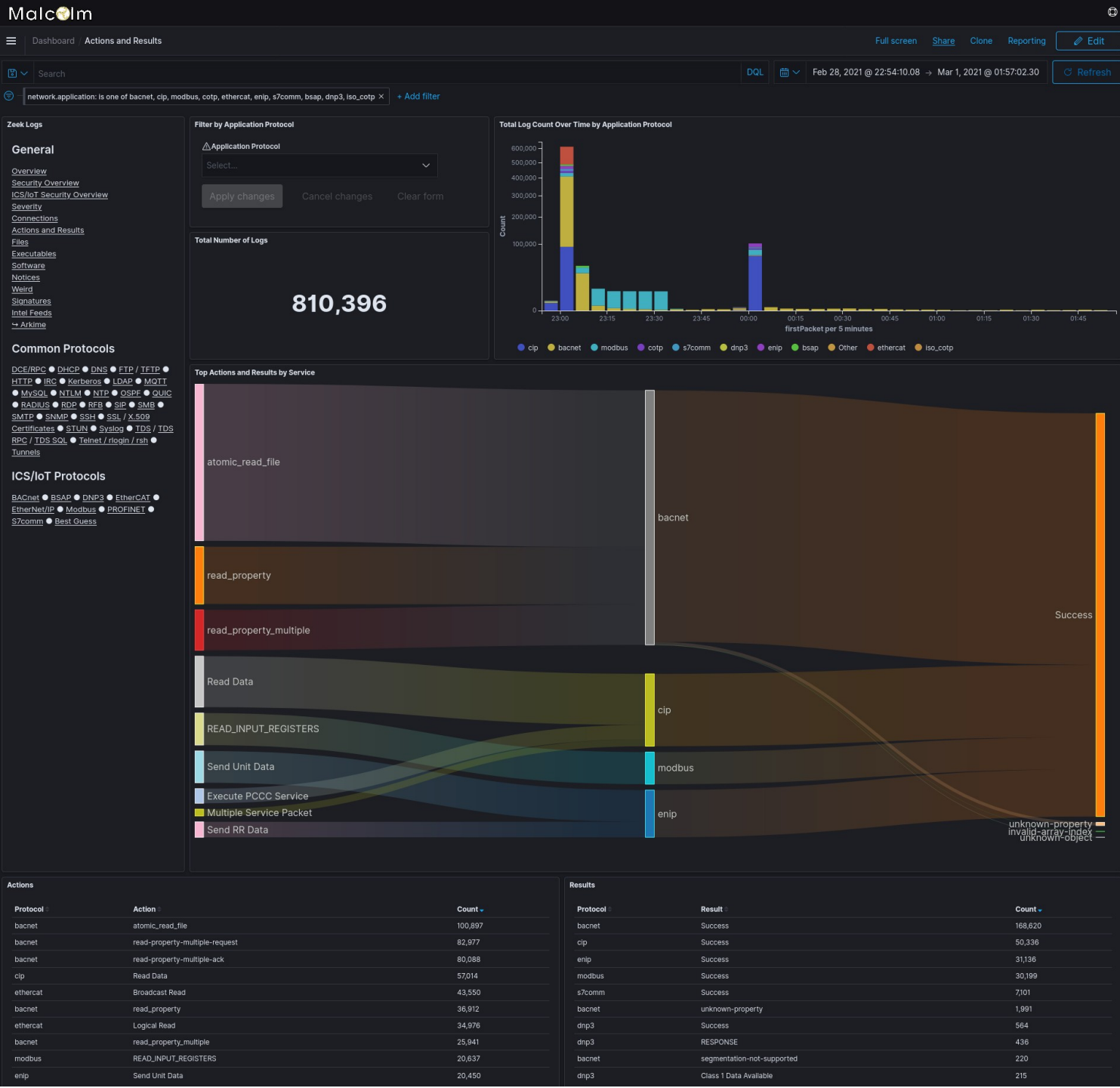
# Notices

- Zeek notices are things that are odd or potentially bad
- In addition to Zeek's defaults, Malcolm raises notices for recent critical vulnerabilities and attack techniques



# Security & ICS/IoT Security Overviews





# 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

# Protocol Dashboards

- Highlight application-specific fields of interest
- Grouped by common IT protocols and ICS/IoT protocols
- ICS protocols
  - BACnet
  - BSAP
  - DNP3
  - EtherCAT
  - EtherNet/IP
  - Modbus
  - OPCUA Binary
  - PROFINET
  - S7comm

[Intelligence](#)

[↔ Arkime](#)

## Common Protocols

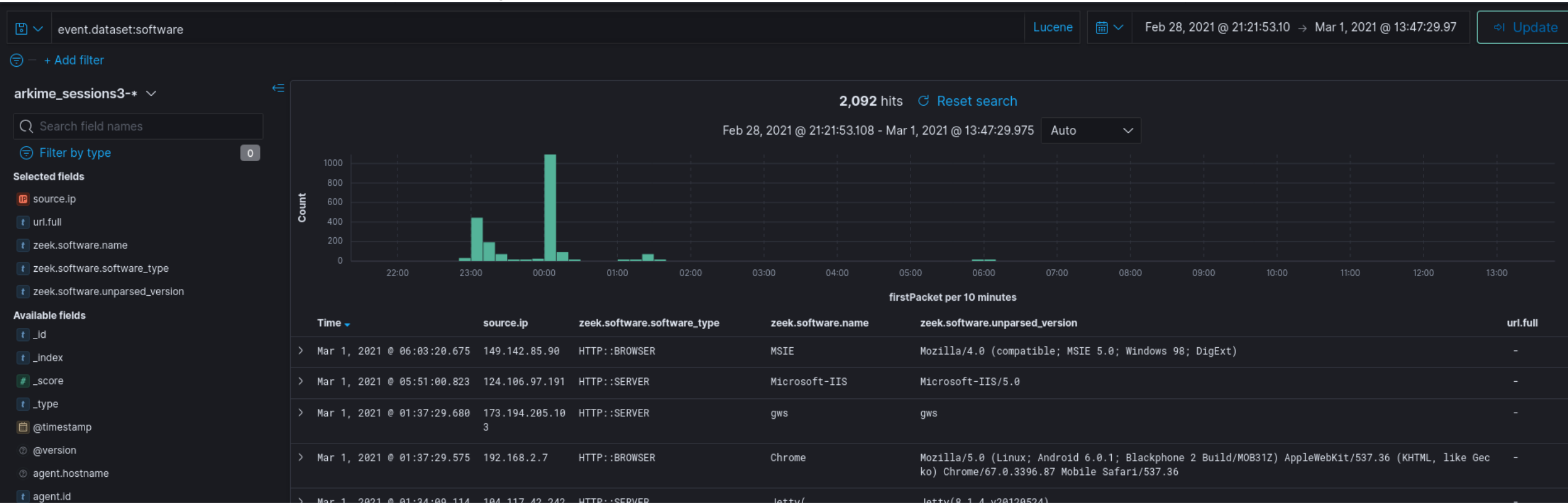
[DCE/RPC](#) ● [DHCP](#) ● [DNS](#) ● [FTP](#) / [TFTP](#) ●  
[HTTP](#) ● [IRC](#) ● [Kerberos](#) ● [LDAP](#) ● [MQTT](#)  
● [MySQL](#) ● [NTLM](#) ● [NTP](#) ● [OSPF](#) ● [QUIC](#)  
● [RADIUS](#) ● [RDP](#) ● [RFB](#) ● [SIP](#) ● [SMB](#) ●  
[SMTP](#) ● [SNMP](#) ● [SSH](#) ● [SSL](#) / [X.509](#)  
[Certificates](#) ● [STUN](#) ● [Syslog](#) ● [TDS](#) / [TDS](#)  
[RPC](#) / [TDS SQL](#) ● [Telnet](#) / [rlogin](#) / [rsh](#) ●  
[Tunnels](#)

## ICS/IoT Protocols

[BACnet](#) ● [BSAP](#) ● [DNP3](#) ● [EtherCAT](#) ●  
[EtherNet/IP](#) ● [Modbus](#) ● [OPCUA Binary](#) ●  
[PROFINET](#) ● [S7comm](#) ● [Best Guess](#)

# Discover

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



## New Visualization

Q Filter



Area



Controls



Coordinate  
Map



Data Table



Gantt Chart



Gauge



Goal



Heat Map



Horizontal Bar



Line



Markdown



Metric



Pie



Region Map



Sankey  
Diagram



TSVB



Tag Cloud



Timeline



Vega



Vertical Bar

# Custom Visualizations

- Create new visualizations from scratch or based on existing charts or dashboards

# Search Syntax Comparison

	Arkime	Dashboards (Lucene)	Dashboards (DQL)
Field exists	<code>zeek.logType == EXISTS!</code>	<code>_exists_:zeek.logType</code>	<code>zeek.logType:*</code>
Field does not exist	<code>zeek.logType != EXISTS!</code>	<code>NOT _exists_:zeek.logType</code>	<code>NOT zeek.logType:*</code>
Field matches a value	<code>port.dst == 22</code>	<code>dstPort:22</code>	<code>dstPort:22</code>
Field does not match a value	<code>port.dst != 22</code>	<code>NOT dstPort:22</code>	<code>NOT dstPort:22</code>
Field matches at least one of a list of values	<code>tags == [external_source, external_destination]</code>	<code>tags:(external_source OR external_destination)</code>	<code>tags:(external_source or external_destination)</code>
Field range (inclusive)	<code>http.statuscode &gt;= 200 &amp;&amp; http.statuscode &lt;= 300</code>	<code>http.statuscode:[200 TO 300]</code>	<code>http.statuscode &gt;= 200 and http.statuscode &lt;= 300</code>

# Search Syntax Comparison (cont.)

	Arkime	Dashboards (Lucene)	Dashboards (DQL)
Field range (exclusive)	<code>http.statuscode &gt; 200 &amp;&amp; http.statuscode &lt; 300</code>	<code>http.statuscode:{200 TO 300}</code>	<code>http.statuscode &gt; 200 and http.statuscode &lt; 300</code>
Field range (mixed exclusivity)	<code>http.statuscode &gt;= 200 &amp;&amp; http.statuscode &lt; 300</code>	<code>http.statuscode:[200 TO 300}</code>	<code>http.statuscode &gt;= 200 and http.statuscode &lt; 300</code>
Match all search terms (AND)	<code>(tags == [external_source, external_destination]) &amp;&amp; (http.statuscode == 401)</code>	<code>tags:(external_source OR external_destination) AND http.statuscode:401</code>	<code>tags:(external_source or external_destination) and http.statuscode:401</code>
Match any search terms (OR)	<code>(zeek_ftp.password == EXISTS!)    (zeek_http.password == EXISTS!)    (zeek.user == "anonymous")</code>	<code>_exists_:zeek_ftp.password OR _exists_:zeek_http.password OR zeek.user:"anonymous"</code>	<code>zeek_ftp.password:* or zeek_http.password:* or zeek.user:"anonymous"</code>



# Search Syntax Comparison (cont.)

	Arkime	Dashboards (Lucene)	Dashboards (DQL)
Global string search (anywhere in the document)	all Arkime search expressions are field-based	microsoft	microsoft
Wildcards	host.dns == "*micro?oft*" ( ? for single character, * for any characters)	dns.host:*micro?oft* ( ? for single character, * for any characters)	dns.host:*micro*ft* ( * for any characters)
Regex	host.http == /. *www\.f.*k\.com.* /	zeek_http.host: /. *www\.f.*k\.com.* /	Dashboards Query Language does not currently support regex
IPv4 values	ip == 0.0.0.0/0	srcIp:"0.0.0.0/0" OR dstIp:"0.0.0.0/0"	srcIp:"0.0.0.0/0" OR dstIp:"0.0.0.0/0"
IPv6 values	(ip.src == EXISTS!    ip.dst == EXISTS!) && (ip != 0.0.0.0/0)	(_exists_:srcIp AND NOT srcIp:"0.0.0.0/0") OR (_exists_:dstIp AND NOT dstIp:"0.0.0.0/0")	(srcIp:* and not srcIp:"0.0.0.0/0") or (dstIp:* and not dstIp:"0.0.0.0/0")

# Search Syntax Comparison (cont.)

	Arkime	Dashboards (Lucene)	Dashboards (DQL)
GeoIP information available	<code>country == EXISTS!</code>	<code>_exists_:zeek.destination_geo OR _exists_:zeek.source_geo</code>	<code>zeek.destination_geo:* or zeek.source_geo:*</code>
Zeek log type	<code>zeek.logType == notice</code>	<code>zeek.logType:notice</code>	<code>zeek.logType:notice</code>
IP CIDR Subnets	<code>ip.src == 172.16.0.0/12</code>	<code>srcIp:"172.16.0.0/12"</code>	<code>srcIp:"172.16.0.0/12"</code>
Search time frame	Use Arkime time bounding controls under the search bar	Use Dashboards time range controls in the upper right-hand corner	Use Dashboards time range controls in the upper right-hand corner
GeoIP information available	<code>country == EXISTS!</code>	<code>_exists_:zeek.destination_geo OR _exists_:zeek.source_geo</code>	<code>zeek.destination_geo:* or zeek.source_geo:*</code>

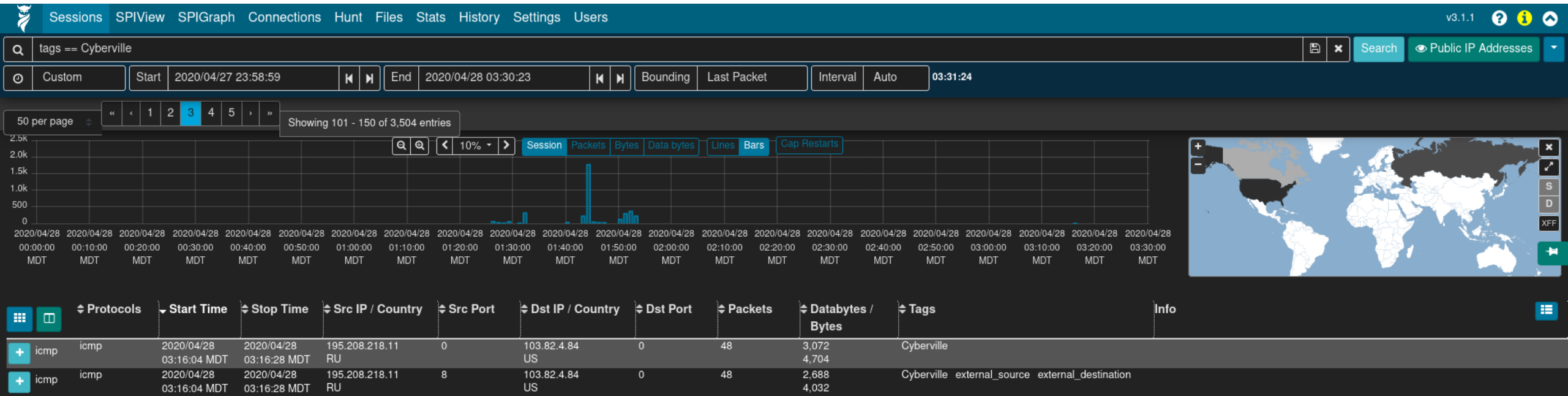
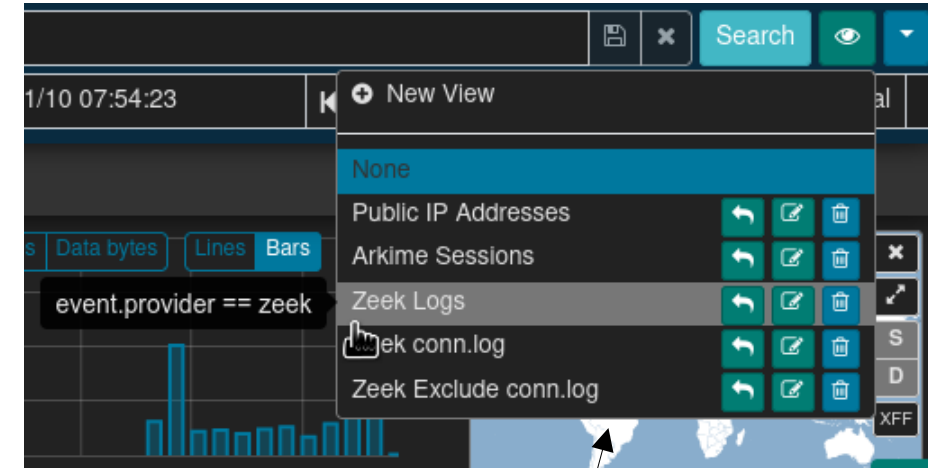


# Arkime

- Front end for **both** enriched Zeek logs and Arkime sessions
  - Malcolm's custom Arkime Zeek data source adds full support for Zeek logs to Arkime, including ICS protocols
- Filter by Zeek logs or Arkime sessions; or, view both together
- “Wireshark at scale”: full PCAP availability for
  - viewing packet payload
  - exporting filtered and joined PCAP sessions
  - running deep-packet searches
- <https://localhost>

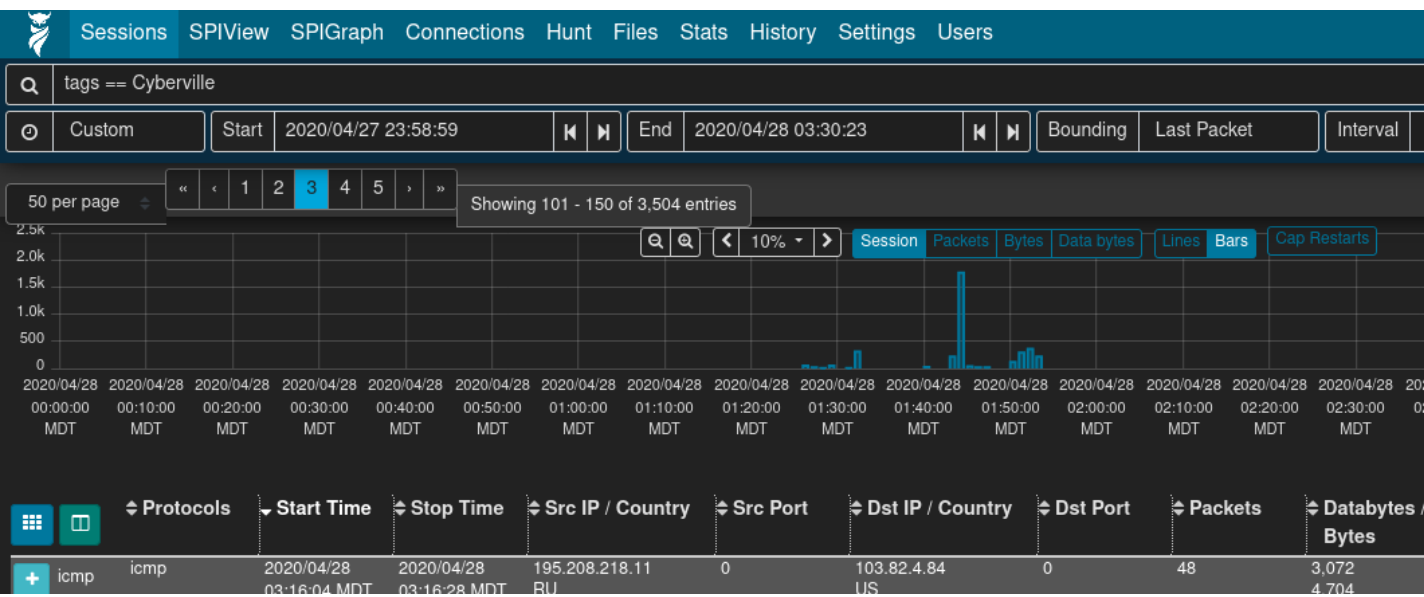
# Arkime Filters and Search

- Time filter: define search time frame
- Map filter: restrict results to geolocation
- Query bar: write queries in Arkime syntax
- Views: overlay previously-specified filters on current search



# Sessions

- Field-level details of sessions/logs matching filters
- Similar to Dashboards' Discover



The screenshot shows the 'Zeek http.log' interface with a search filter 'protocols == http && tags == external\_destination'. The time range is from 2020/11/11 06:23:48 to 2021/05/30 06:00:53. The interface displays detailed session information for a specific entry.

Field	Value
Log Type	http
Malcolm Data Source	zeek
Malcolm Node	filebeat
Originating Host	217.226.31.170
Originating GeoIP Country	Germany
Originating GeoIP City	Bremen
Responding Host	124.106.97.191
Responding GeoIP Country	Philippines
Responding GeoIP City	Santa Elena
Originating Port	4230
Responding Port	80
Related IP	217.226.31.170 124.106.97.191
Protocol	tcp
Service	http
Service Version	1.1
Action	GET
Result	Bad Gateway
Severity	20
Risk Score	20
Severity Tags	External traffic
File Magic	text/html

Additional details shown below the table:

- Pipeline Depth: 1
- Request Method: GET
- URI: /\_vti\_bin/.../winnt/system32/cmd.exe?/c+dir+x:\\c+dir+x:\\c+dir+x:\\
- Version: 1.1

# Packet Payloads

- Displayed for Arkime sessions with full PCAP (i.e., not Zeek logs)
- File carving on the fly
- Download session PCAP
- Examine payload with CyberChef

Source	Destination
GET /PostExploitation/PCAnyPass.exe HTTP/1.1 Accept: text/html, application/xhtml+xml, */* Referer: http://10.10.10.11/PostExploitation/ Accept-Language: en-US User-Agent: Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Trident/5.0) Accept-Encoding: gzip, deflate Host: 10.10.10.11 Connection: Keep-Alive	HTTP/1.0 200 OK Server: SimpleHTTP/0.6 Python/2.7.17 Date: Fri, 17 Apr 2020 19:21:32 GMT Content-type: application/x-msdos-program Content-Length: 49152 Last-Modified: Fri, 16 Apr 2010 19:09:50 GMT  <a href="#">PCAnyPass.exe</a>

Packets200

naturalasciiutf8hex

Show Packets

Line Numbers

Uncompress

Show Image & Files

Show Info

File Bytes:

base64

CyberChef

# Export PCAP

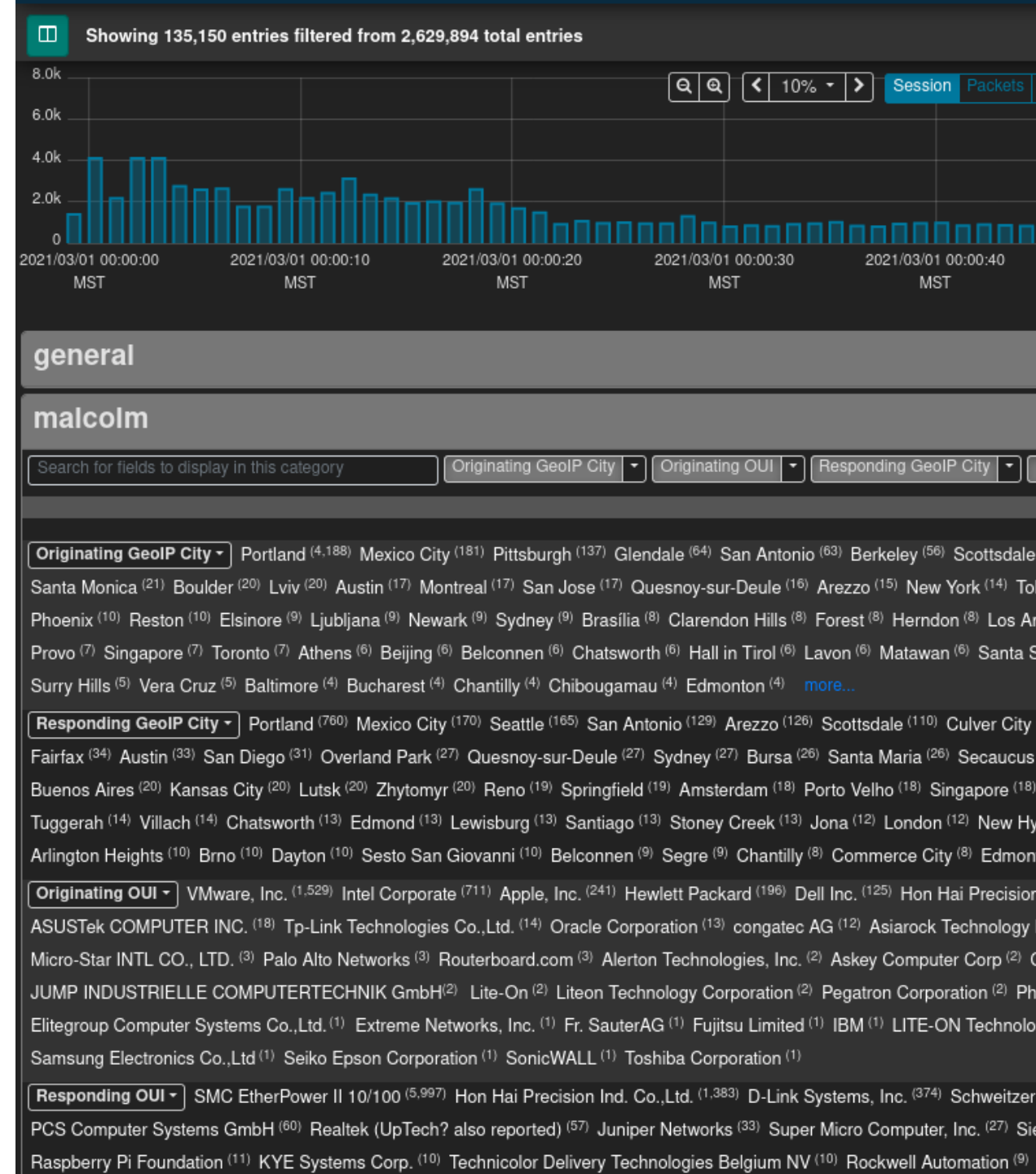
- Creates a new PCAP file from filtered sessions
- Include open, visible or all matching sessions
- Apply “Arkime Sessions” view to sessions first
- Narrow as much as possible prior to exporting (huge PCAP files are a pain)

The screenshot displays the Arkime Sessions web interface. At the top, a navigation bar includes tabs for Sessions, SPIView, SPIGraph, Connections, Hunt, Files, Stats, History, Settings, and Users. A search bar contains the query 'country != US && protocols == http'. Below this, a filter bar shows 'Custom' settings for a time range from 2021/02/28 23:59:11 to 2021/03/01 00:28:26, with a duration of 00:29:15. The 'Include' section is set to 'same time period' and 'linked segments (slow)'. The 'Filename' field is 'US\_HTTP.pcap', and an 'Export PCAP' button is visible. The main area shows a bar chart of session activity over time, with a table below it. The table has columns for Protocols, Start Time, Stop Time, Src IP / Country, Src Port, Dst IP / Country, Dst Port, Packets, Databytes / Bytes, Tags, and Info. The first row shows a session for 'tcp http' on 2021/03/01, with source IP 10.0.52.164 and destination IP 61.8.0.17. A world map is visible in the bottom right corner.

Protocols	Start Time	Stop Time	Src IP / Country	Src Port	Dst IP / Country	Dst Port	Packets	Databytes / Bytes	Tags	Info
tcp http	2021/03/01	2021/03/01	10.0.52.164	2550	61.8.0.17	80	7,195	5,160,414	HTTP out-of-order-dst	URI mirror.pacific.net.au/openoffice/stable/2.0.0/OOo_2.0.0_Win32Intel_install.exe

# SPIView

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





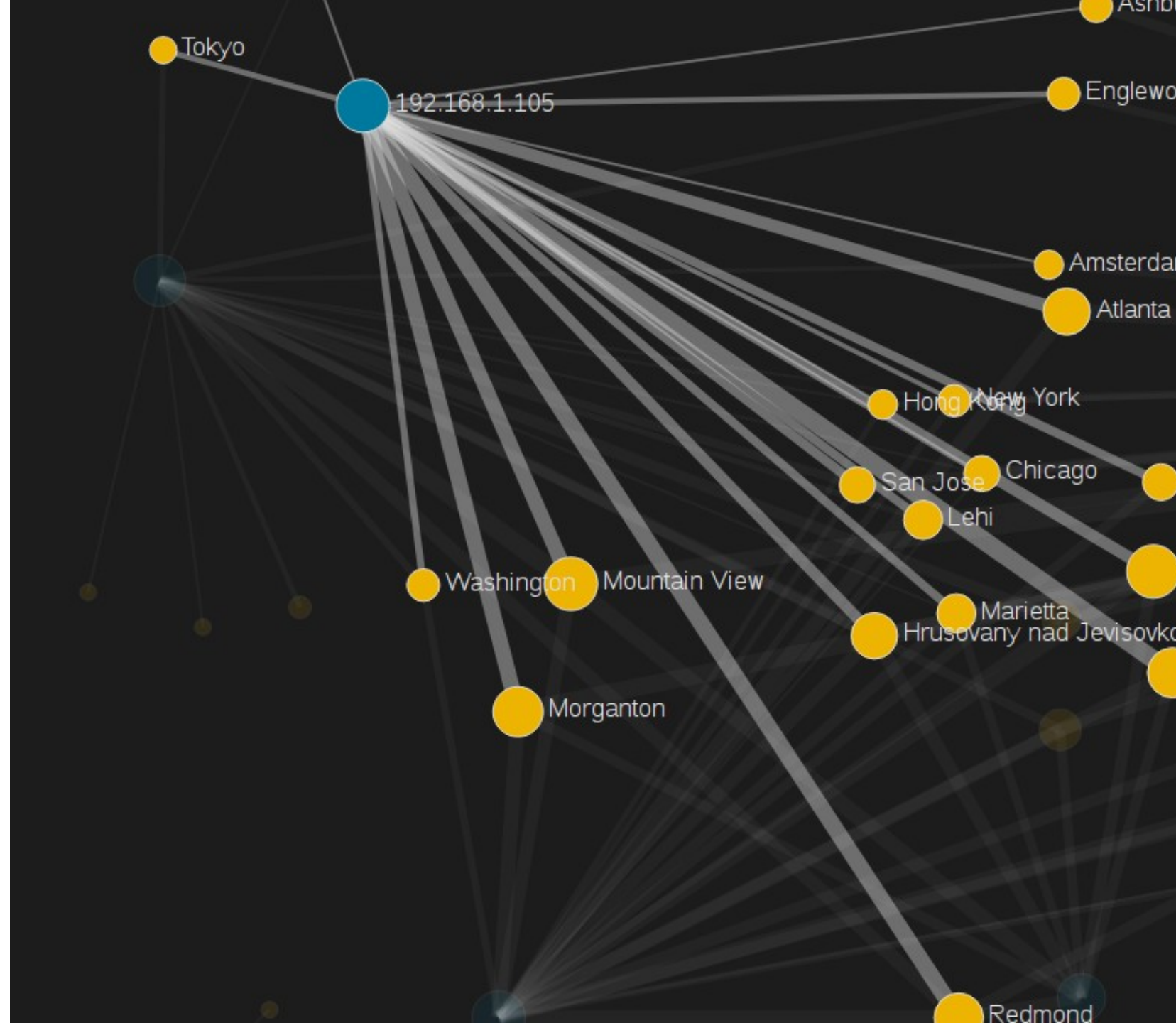
# SPIGraph

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



# Connections

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



# Packet Search (“Hunt”)

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

The screenshot displays the Arkime Hunt interface. At the top, a navigation bar includes links for Sessions, SPIView, SPIGraph, Connections, Hunt (active), Files, Stats, History, Settings, and Users. The version is v3.1.1. Below the navigation bar is a search bar with the query 'protocols == http'. To the right of the search bar are buttons for 'Search' and 'Arkime Sessions'. Below the search bar is a filter bar with 'All (careful)' selected, and time range selectors for 'Start' (1969/12/31 17:00:00) and 'End' (2021/12/06 12:10:02). Below the filter bar is a message: 'Creating a new packet search job will search the packets of 2,906 sessions.' To the right of this message is a button 'Create a packet search job'. Below the message is a section titled 'Hunt Job History'. This section has a search bar for 'Search your packet search job history' and a '50 per page' dropdown. Below the search bar is a table with the following columns: Status, Matches, Name, User, Search text, Notify, Created, and ID. The table contains one entry with the following data: Status: 100%, Matches: 141, Name: HTTP with password, User: (empty), Search text: password (ascii), Notify: (empty), Created: 2021/12/06 12:12:27 MST, ID: s5YpkX0BTA40FhD4X7dA. Below the table is a section titled 'This hunt is finished' with the following details: Found 141 sessions matching password (ascii) of 2,908 sessions searched. Created: 2021/12/06 12:12:27 MST. Last Updated: 2021/12/06 12:12:32 MST. Examining 500 raw source and destination packets per session. The sessions query expression was: protocols == http. The sessions query view was: Arkime Sessions. The sessions query time range was from 1969/12/31 17:00:00 MST to 2021/12/06 12:10:02 MST.

Sessions SPIView SPIGraph Connections **Hunt** Files Stats History Settings Users v3.1.1

Q protocols == http Search Arkime Sessions

All (careful) Start 1969/12/31 17:00:00 End 2021/12/06 12:10:02 Bounding Last Packet

Creating a new packet search job will search the packets of 2,906 sessions. Create a packet search job

### Hunt Job History

Q Search your packet search job history 50 per page Showing 1 - 1 of 1 entries

Status	Matches	Name	User	Search text	Notify	Created	ID
100%	141	HTTP with password		password (ascii)		2021/12/06 12:12:27 MST	s5YpkX0BTA40FhD4X7dA

✓ This hunt is **finished**

Found 141 sessions matching **password** (ascii) of 2,908 sessions searched

Created: 2021/12/06 12:12:27 MST

Last Updated: 2021/12/06 12:12:32 MST

Examining 500 raw source and destination packets per session

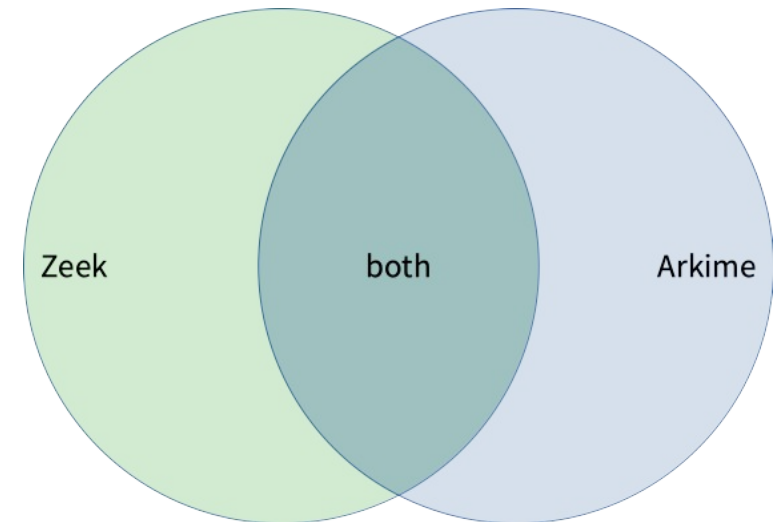
The sessions query expression was: **protocols == http**

The sessions query view was: **Arkime Sessions**

The sessions query time range was from 1969/12/31 17:00:00 MST to 2021/12/06 12:10:02 MST

# Data Source Correlation

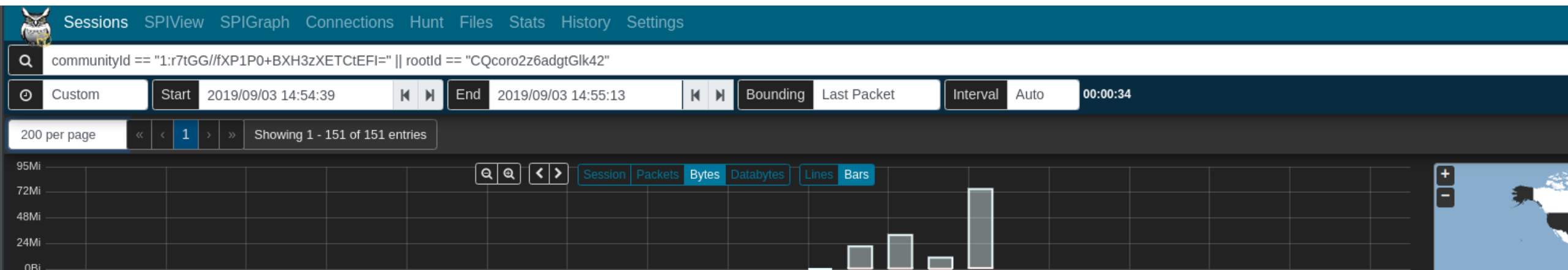
- Search syntax is different between Arkime and Dashboards (and in some cases, so are field names)
  - See search syntax comparison table, Malcolm and Arkime docs
- Despite considerable overlap, there are differences in protocol parser support between Zeek and Arkime
  - Learning the strengths of each will help you more effectively find the good stuff



# Correlate Zeek Logs and Packet Payloads

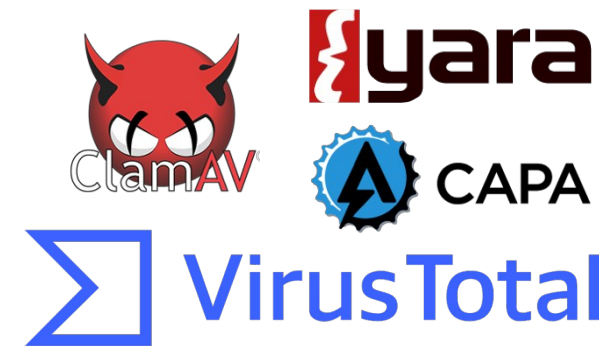
- Correlate Zeek logs and Arkime sessions using common fields
- `communityId` fingerprints flows in both and can bridge the two
- `rootId / zeek.uid` filters Zeek logs for the same session
- Filter community ID OR'ed with Zeek UID to see all Arkime sessions and Zeek logs for the same traffic

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



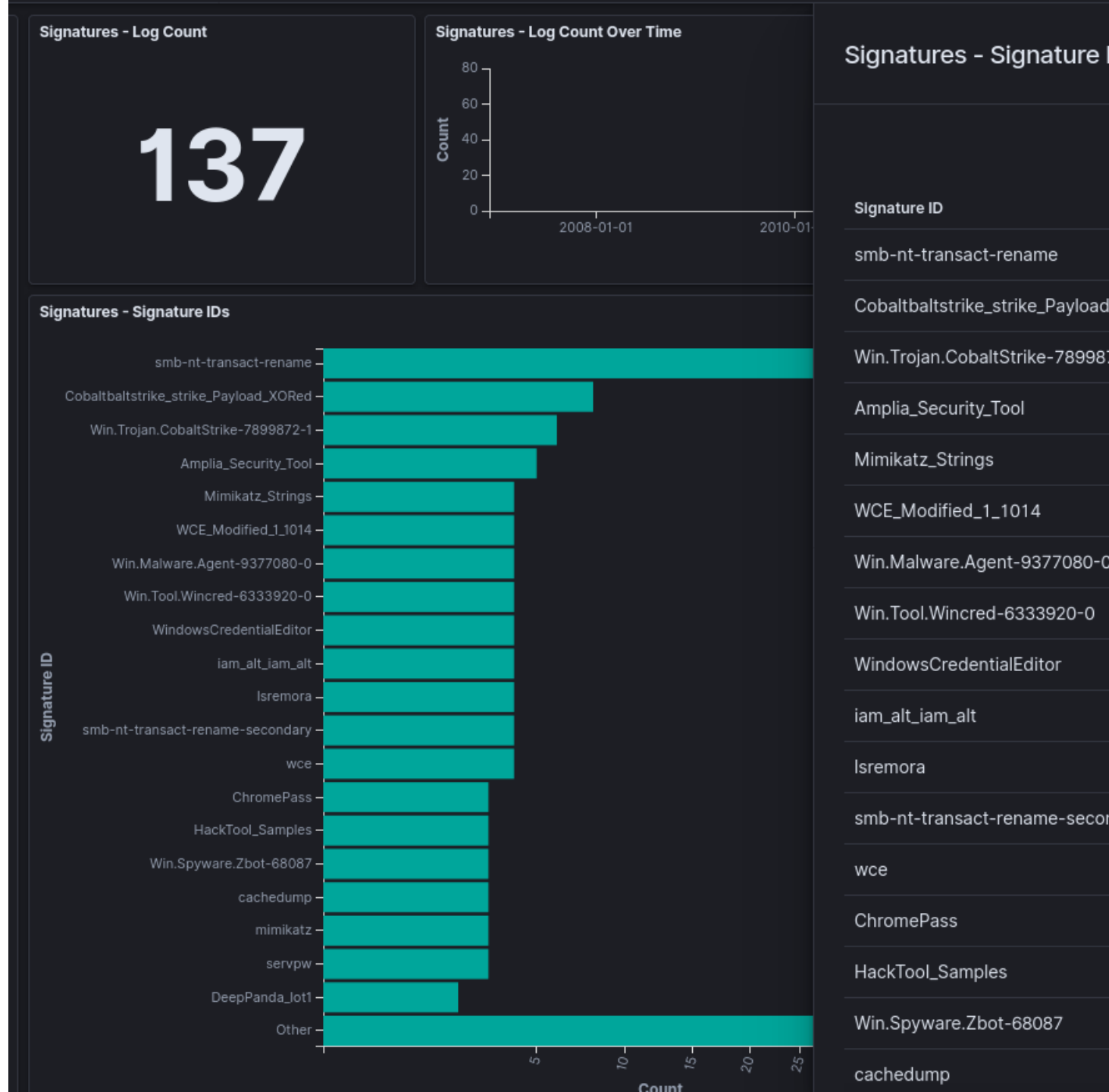
# File Analysis

- Zeek can “carve” file transfers from common protocols
- Malcolm can examine carved files and flag hits
  - ClamAV – open source antivirus engine
  - YARA – pattern matching swiss army knife
  - Capa – portable executable capabilities analyzer
  - VirusTotal – online database of file hashes
    - requires API token and internet connection
- Triggering files can be saved to `zeek-logs/extract_files` under Malcolm directory for further analysis
  - Be careful! Carved files may contain live malware!



# Signatures

- Signatures dashboard in Dashboards shows scanned file hits
- Use `zeek.fuid` field in *Signatures - Logs* table to pivot to connection UID (`zeek.uid`) and other logs with pertinent session details



# 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 Arkime and Zeek regardless of data source (e.g., `protocol:http` in Dashboards and `protocols == http` in Arkime)
- Use tags



# Malcolm



## Thank you!

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

Malcolm is Copyright © 2022 Battelle Energy Alliance, LLC, and is developed and released as open-source software through the cooperation of the Cybersecurity and Infrastructure Security Agency of the US Department of Homeland Security.