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 (e.g., Suricata)
  - Statistical anomaly-based detection (e.g., Random Cut Forest)
  - Stateful protocol analysis detection (e.g., Zeek)
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™)
<table>
<thead>
<tr>
<th>Strengths</th>
<th>Weaknesses</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Analyzes both link-layer and application-layer behavior</td>
<td>• Session metadata only (not full payload)</td>
</tr>
<tr>
<td>• Content extraction</td>
<td>• Setup and configuration can be complicated</td>
</tr>
<tr>
<td>• Behavioral analysis</td>
<td>• Produces flat textual log files which can be unwieldy for in-depth analysis</td>
</tr>
<tr>
<td>• Session correlation</td>
<td></td>
</tr>
<tr>
<td>• Can add support for uncommon protocols through scripts/plugins</td>
<td></td>
</tr>
</tbody>
</table>
Zeek Log Files

- Network Protocols
- Files
- Detection
- Network Observations
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, ...

Detection (cont.)

• **weird** – Unexpected network-level activity
  • > 150 weirdness indicators across many protocols

• **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
<table>
<thead>
<tr>
<th>Strengths</th>
<th>Weaknesses</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Large scale index packet capture and search tool</td>
<td>• No OT protocol support</td>
</tr>
<tr>
<td>• Packet analysis engine with support for many common IT protocols</td>
<td>• Adding new protocol parsers requires C programming</td>
</tr>
<tr>
<td>• Web interface for browsing, searching, analysis and PCAP carving for exporting</td>
<td></td>
</tr>
<tr>
<td>• PCAP payloads (not just session header/metadata) are viewable and searchable</td>
<td></td>
</tr>
</tbody>
</table>
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, Suricata IDS, 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.
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)
GENISYS
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 / Virtual Network Computing (RFB/VNC)
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

PCAP

Suricata

Zeek

Filebeat

Logstash

OpenSearch

OpenSearch Dashboards

Analyst

Arkime Viewer

Arkime

capture

Filebeat

Carved File Scan
hash- or signature-based

ClamAV

yara

CAPA

VIRUSTOTAL
Traffic is collected passively by the Hedgehog sensor device.

Metadata is securely forwarded to Malcolm.

Logs are enriched and stored in OpenSearch.

Machine learning algorithms identify anomalies.

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

Traffic is visualized in OpenSearch Dashboards and Arkime Viewer.

- Malcolm can authenticate users from its own list or via Active Directory / LDAP.
- Alerts are sent over email, webhooks, Slack or Amazon Chime.
- 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.
- Traffic is collected passively by the Hedgehog sensor device.
- Zeek, Arkime Capture and Suricata 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.
- 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.
- 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 may be triggered by exceeded thresholds, anomalies detected, custom queries, etc.
- 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.

Data Pipeline
https://github.com/idaholab/Malcolm
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
- Walkthroughs on YouTube: search “Malcolm Network Traffic Analysis”
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
Importing Traffic Captures for Analysis

- Specify tags for search and filter
- Enable Suricata and/or Zeek analysis and file extraction
  - Or configure as global defaults
- Upload PCAP files or archived Zeek logs
  - pcapng not supported yet
- https://localhost/upload
Data Tagging and Enrichment

• Logstash enriches Zeek and Suricata log metadata
  • 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 Arkime sessions, Zeek logs and Arkime alerts 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 and Suricata alerts
• 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
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
Zeek 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.
Suricata Alerts

- Protocol-aware Suricata signatures generate alerts for suspect traffic
- Use the default Emerging Threats Open ruleset or custom signatures from other sources

Network Logs

General
Overview
Security Overview
ICS/IoT Security Overview
Severity
Connections
Actions and Results
Files
Executables
Software
Zeek: Intelligence
Zeek: Notices
Zeek: Ward
Signatures
Suricata Alerts
+ Arkime

Common Protocols
- DC/DC
- DHCP
- DNS
- FTP / TFTP
- HTTP
- RPC
- Kerberos
- LDAP
- MQTT
- MySQL
- NETBIOS
- NTP
- OSPF
- QUID
- RADIUS
- RDP
- SNMP
- Samba
- SMTP
- SSH
- SNMP
- SSL
- SSH / IP
- Certificates
- STUN
- Syslog
- TDS / TDS
- RPC / TDS
- SQL
- Telnet / rlogin / rsh
- Tunnels

ICS/IoT Protocols
- BACnet
- BEAP
- DNP3
- EtherCAT
- EtherNet/IP
- GENIUS
- Modbus
- OPC-UA binary
- PROFINET
- SMTP
- Best Guess

Alerts - Log Count

1,758

Alerts - Log Count Over Time

Alerts - Tags

Suspicous_POST_body
Exploit_Kit
Dshield
SSL
LDAP
FTP
TOR
Blackhole
DCERPC
SMB
QUIC
SMTP
DNS
HTTP
SpamBot
Spyware_User_Agent

Alerts - Target

<table>
<thead>
<tr>
<th>Target Name</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Windows_XP_Vista_7.8.30_Server_22.64_Bit</td>
<td>352</td>
</tr>
<tr>
<td>DNS Server</td>
<td>260</td>
</tr>
<tr>
<td>Client_Endpoint</td>
<td>165</td>
</tr>
<tr>
<td>Any</td>
<td>98</td>
</tr>
<tr>
<td>SMB_CLIENT</td>
<td>29</td>
</tr>
</tbody>
</table>

Alerts - Name

<table>
<thead>
<tr>
<th>Name</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>ET POLICY HTTP traffic on port 443 (POST)</td>
<td>977</td>
</tr>
<tr>
<td>ET HUNTING Suspicious NULL DNS Request</td>
<td>222</td>
</tr>
<tr>
<td>ET JASS Hash - (Abuse.ch) Possible Adware</td>
<td>76</td>
</tr>
<tr>
<td>ET POLICY Command Shell Activity Using Cramesic Environmental Variable Over SMB - Very Likely Lateral Movement</td>
<td>40</td>
</tr>
</tbody>
</table>
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
  - GENISYS
  - Modbus
  - OPCUA Binary
  - PROFINET
  - S7comm
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
Custom Visualizations

• Create new visualizations from scratch or based on existing charts or dashboards
## Search Syntax Comparison

<table>
<thead>
<tr>
<th></th>
<th>Arkime</th>
<th>Dashboards (Lucene)</th>
<th>Dashboards (DQL)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Field exists</strong></td>
<td>event.dataset == EXISTS!</td>
<td><em>exists</em>:event.dataset</td>
<td>event.dataset:*</td>
</tr>
<tr>
<td><strong>Field does not exist</strong></td>
<td>event.dataset != EXISTS!</td>
<td>NOT <em>exists</em>:event.dataset</td>
<td>NOT event.dataset:*</td>
</tr>
<tr>
<td><strong>Field matches a value</strong></td>
<td>port.dst == 22</td>
<td>destination.port:22</td>
<td>destination.port:22</td>
</tr>
<tr>
<td><strong>Field does not match a value</strong></td>
<td>port.dst != 22</td>
<td>NOT destination.port:22</td>
<td>NOT destination.port:22</td>
</tr>
<tr>
<td><strong>Field matches at least one of a list of values</strong></td>
<td>tags == [external_source, external_destination]</td>
<td>tags:(external_source OR external_destination)</td>
<td>tags:(external_source or external_destination)</td>
</tr>
<tr>
<td><strong>Field range (inclusive)</strong></td>
<td>http.statuscode &gt;= 200 &amp; http.statuscode &lt;= 300</td>
<td>http.statuscode:[200 TO 300]</td>
<td>http.statuscode &gt;= 200 and http.statuscode &lt;= 300</td>
</tr>
<tr>
<td>Field range (exclusive)</td>
<td>Arkime</td>
<td>Dashboards (Lucene)</td>
<td>Dashboards (DQL)</td>
</tr>
<tr>
<td>-------------------------</td>
<td>--------</td>
<td>---------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>Field range (mixed exclusivity)</td>
<td>$\text{http.statuscode} &gt; 200 \land \text{http.statuscode} &lt; 300$</td>
<td>$\text{http.statuscode}:[200 \text{ TO } 300}$</td>
<td>$\text{http.statuscode} &gt; 200 \text{ and } \text{http.statuscode} &lt; 300$</td>
</tr>
<tr>
<td>Match all search terms (AND)</td>
<td>$(\text{tags} = [\text{external_source, external_destination}]) \land (\text{http.statuscode} = 401)$</td>
<td>$\text{tags}:(\text{external_source OR external_destination}) \land \text{http.statuscode}:401$</td>
<td>$\text{tags}:(\text{external_source or external_destination}) \text{ and } \text{http.statuscode}:401$</td>
</tr>
<tr>
<td>Match any search terms (OR)</td>
<td>$(\text{zeek_ftp.password} = \text{EXISTS!}) \lor (\text{zeek_http.password} = \text{EXISTS!}) \lor (\text{zeek.user} = \text{&quot;anonymous&quot;})$</td>
<td>$\text{_exists:_zeek_ftp.password OR _exists:_zeek_http.password OR zeek.user:&quot;anonymous&quot;}$</td>
<td>$\text{zeek_ftp.password:* or zeek_http.password:* or zeek.user:&quot;anonymous&quot;}$</td>
</tr>
</tbody>
</table>
## Search Syntax Comparison (cont.)

<table>
<thead>
<tr>
<th></th>
<th>Arkime</th>
<th>Dashboards (Lucene)</th>
<th>Dashboards (DQL)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Global string search (anywhere in the document)</strong></td>
<td>all Arkime search expressions are field-based</td>
<td>microsoft</td>
<td>microsoft</td>
</tr>
<tr>
<td><strong>Wildcards</strong></td>
<td>host.dns == &quot;<em>micro?oft</em>&quot; (? for single character, * for any characters)</td>
<td>dns.host:<em>micro?oft</em></td>
<td>dns.host:<em>micro</em>ft*</td>
</tr>
<tr>
<td></td>
<td>(? for single character, * for any characters)</td>
<td>(for single character, * for any characters)</td>
<td>(for any characters)</td>
</tr>
<tr>
<td><strong>Regex</strong></td>
<td>host.http == /.*www./f.<em>k./com.</em>/</td>
<td>zeek_http.host: /.*www./f.<em>k./com.</em>/</td>
<td>Dashboards Query Language does not currently support regex</td>
</tr>
<tr>
<td><strong>IPv4 values</strong></td>
<td>ip == 0.0.0.0/0</td>
<td>source.ip:&quot;0.0.0.0/0&quot; OR destination.ip:&quot;0.0.0.0/0&quot;</td>
<td>source.ip:&quot;0.0.0.0/0&quot; OR destination.ip:&quot;0.0.0.0/0&quot;</td>
</tr>
<tr>
<td><strong>IPv6 values</strong></td>
<td>(ip.src == EXISTS!</td>
<td></td>
<td>ip.dst == EXISTS!) &amp;&amp; (ip != 0.0.0.0/0)</td>
</tr>
<tr>
<td></td>
<td>Arkime</td>
<td>Dashboards (Lucene)</td>
<td>Dashboards (DQL)</td>
</tr>
<tr>
<td>-------------------------</td>
<td>--------</td>
<td>--------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>GeoIP information available</td>
<td>country == EXISTS!</td>
<td><em>exists</em>:destination.geo OR <em>exists</em>:source.geo</td>
<td>destination.geo:* or source.geo:*</td>
</tr>
<tr>
<td>Log type</td>
<td>event.dataset == notice</td>
<td>event.dataset:notice</td>
<td>event.dataset:notice</td>
</tr>
<tr>
<td>IP CIDR Subnets</td>
<td>ip.src == 172.16.0.0/12</td>
<td>source.ip:&quot;172.16.0.0/12&quot;</td>
<td>source.ip:&quot;172.16.0.0/12&quot;</td>
</tr>
<tr>
<td>Search time frame</td>
<td>Use Arkime time bounding controls under the search bar</td>
<td>Use Dashboards time range controls in the upper right-hand corner</td>
<td>Use Dashboards time range controls in the upper right-hand corner</td>
</tr>
<tr>
<td>GeoIP information available</td>
<td>country == EXISTS!</td>
<td><em>exists</em>:destination.geo OR <em>exists</em>:source.geo</td>
<td>destination.geo:* or source.geo:*</td>
</tr>
</tbody>
</table>
• Front end for both enriched Zeek logs, Suricata alerts and Arkime sessions
  • Malcolm’s custom Arkime Zeek data source adds full support for Zeek logs to Arkime, including ICS protocols
• Filter by data source (Zeek, Suricata or Arkime); or, view 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
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
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)
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 ≤ 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
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 among Zeek, Suricata and Arkime
  • Learning the strengths of each will help you more effectively find the good stuff
Correlate Zeek or Suricata Logs and Packet Payloads

- Correlate Zeek or Suricata logs and Arkime sessions using common fields
  - communityId fingerprints flows to bridge data sources
  - rootId/event.id filters logs for the same session
  - Filter community ID OR’ed with event.id to see all Arkime sessions and Zeek or Suricata logs for the same traffic

```plaintext
communityId == "1:r7tGG//fXP1P0+BXH3zXETCtEFI=" || event.id == "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 (2000+)
• Filter on event.dataset (e.g., conn to see conn.log)
• Filter on protocol regardless of data source (e.g., protocol:http in Dashboards and protocols == http in Arkime)
• Use tags
Thank you!

Visit Malcolm on GitHub to read the docs, make suggestions, report issues and star 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.