tshark + ELK: Network Traffic Monitoring and Analysis

Project

https://www.h21lab.com/tools/tshark-elasticsearch

https://github.com/H21lab/tsharkVM

About me

Martin Kacer

www.h21lab.com

Credits and special thanks

Special thanks to people who helped with the Wireshark development or otherwise contributed to this work:

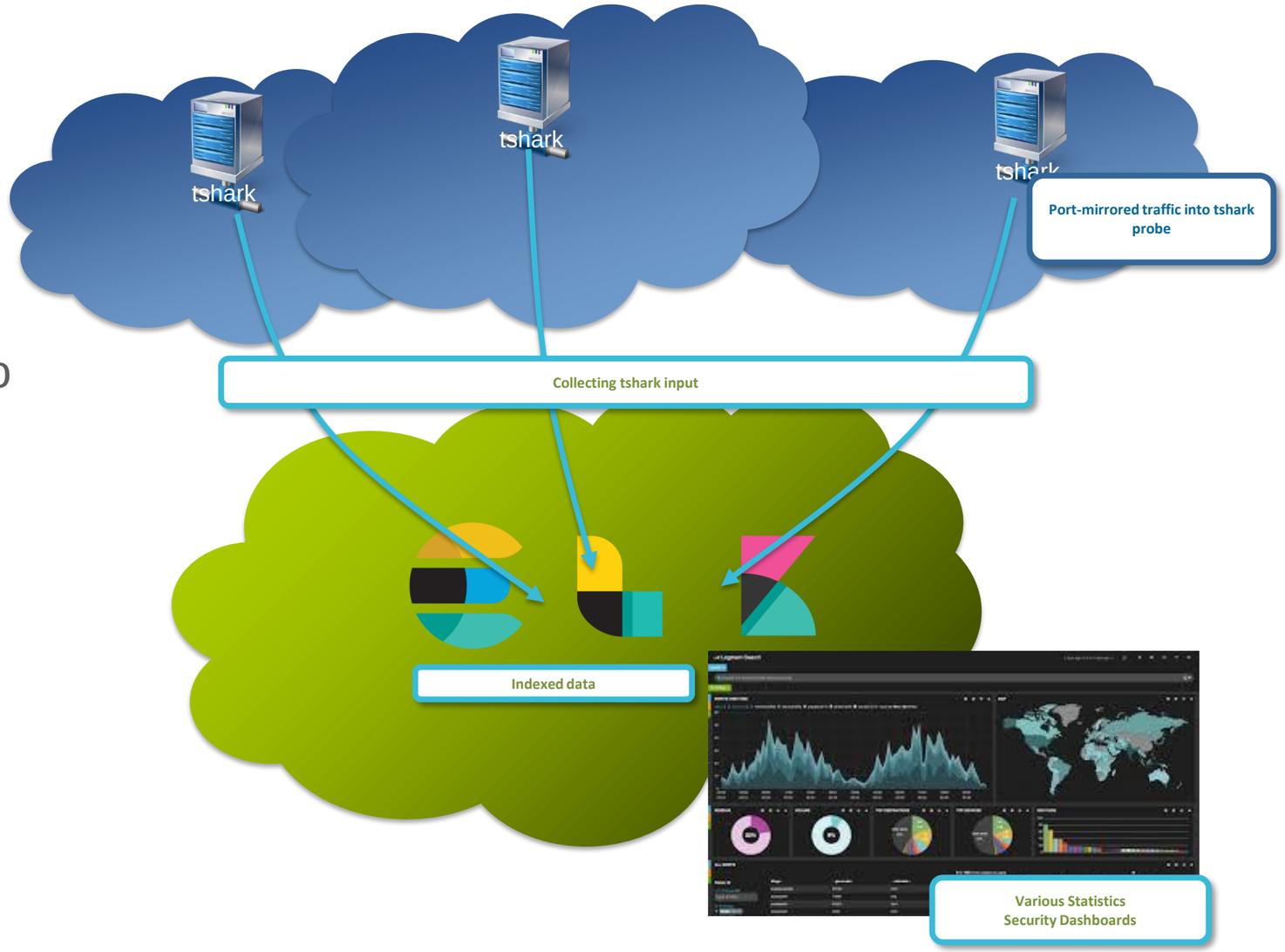
- Anders Broman
- Alexis La Goutte
- Christoph Wurm
- Dario Lombardo
- Vic Hargrave

Overview

TShark is a terminal oriented version of Wireshark

tshark can be used in this way as monitoring probe to push the data into Elasticsearch cluster which enables:

- Indexing of the selected protocol data
- Free monitoring tool (for all protocols which wireshark support)
- Visualizations and dashboards in Kibana
- Possible further analytic and correlation



tshark json format

man tshark

-T ek|fields|json|jsonraw|pdml|ps|psml|tabs|text
Set the format of the output when viewing decoded packet data. The options are one of:

ek Newline delimited JSON format for bulk import into Elasticsearch. It can be used with -j or -J including the JSON filter or with -x to include raw hex-encoded packet data. If -P is specified it will print the packet summary only, with both -P and -V it will print the packet summary and packet details. If neither -P or -V are used it will print the packet details only. Example of usage to import data into Elasticsearch:

```
tshark -T ek -j "http tcp ip" -P -V -x -r file.pcap > file.json
curl -H "Content-Type: application/x-ndjson" -XPOST http://elasticsearch:9200/_bulk --data-binary
"@file.json"
```

Elastic requires a mapping file to be loaded as template for packets-* index in order to convert wireshark types to elastic types. This file can be auto-generated with the command "tshark -G elastic-mapping". Since the mapping file can be huge, protocols can be selected by using the option --elastic-mapping-filter:

```
tshark -G elastic-mapping --elastic-mapping-filter ip,udp,dns
```

tshark json format

fields The values of fields specified with the -e option, in a form specified by the -E option. For example,

```
tshark -T fields -E separator=, -E quote=d
```

would generate comma-separated values (CSV) output suitable for importing into your favorite spreadsheet program.

json JSON file format. It can be used with -j or -J including the JSON filter or with -x option to include raw hex-encoded packet data. Example of usage:

```
tshark -T json -r file.pcap
tshark -T json -j "http tcp ip" -x -r file.pcap
```

jsonraw JSON file format including only raw hex-encoded packet data. It can be used with -j including or -J the JSON filter option. Example of usage:

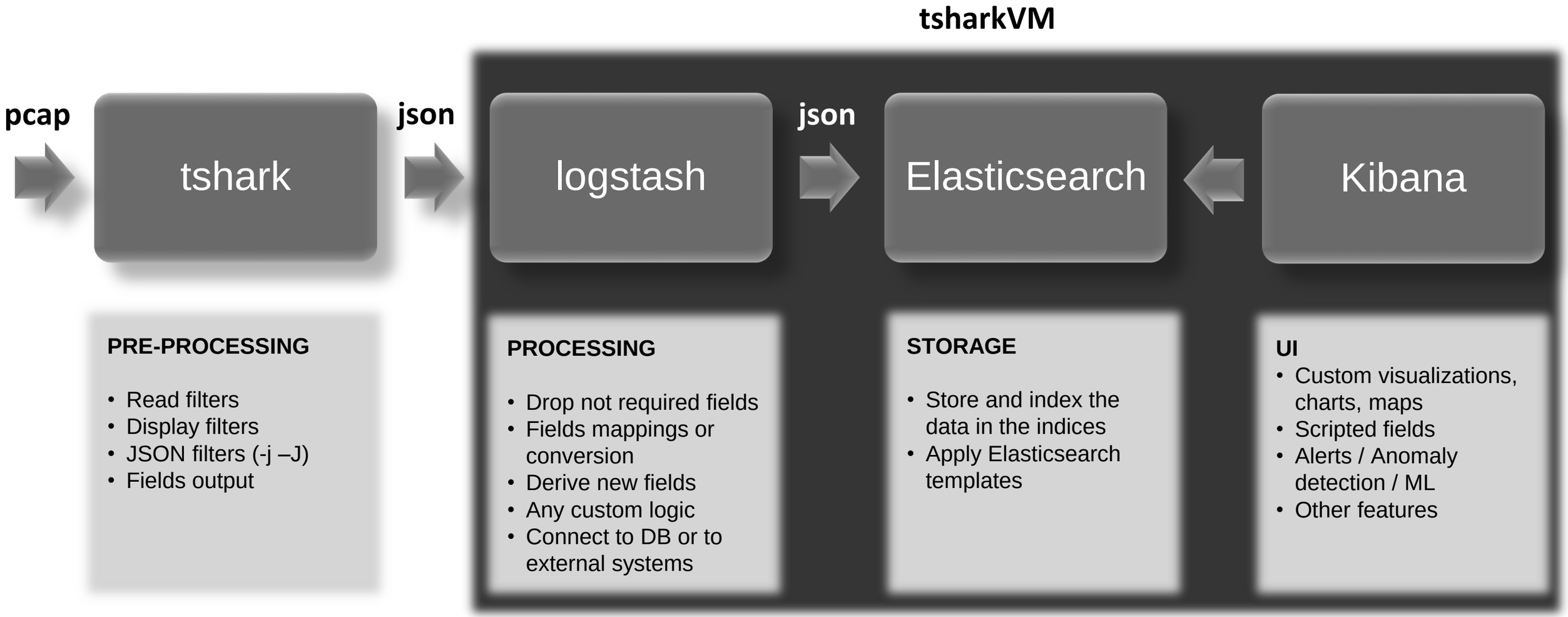
```
tshark -T jsonraw -r file.pcap
tshark -T jsonraw -j "http tcp ip" -x -r file.pcap
```

tshark json format

```
-j  <protocol match filter>
Protocol match filter used for ek|json|jsonraw|pdml output file types.  Parent node containing
multiple child nodes is only included, if the name is found in the filter.

Example: tshark -j "ip ip.flags text"

-J  <protocol match filter>
Protocol top level filter used for ek|json|jsonraw|pdml output file types.  Parent node containing
multiple child nodes is included with all children.

Example: tshark -J "http tcp"

...

--no-duplicate-keys      If -T json is specified, merge duplicate keys in an object into a single
key with as value a json array containing all values
```

NOTE: The -T ek is de-duplicated by default in the latest wireshark code. This is required for Elasticsearch 6.0 and higher due the strict duplicate checking. The use of switch --no-duplicate-keys should be used based on the json parsers. Without this switch, the json generates also duplicated values. This has been described in wireshark bug 12958.

tsharkVM overview



tsharkVM how-to

Clone source code

```
git clone https://github.com/H21lab/tsharkVM.git
```

Build tshark VM

```
sudo apt update
sudo apt install tshark virtualbox vagrant
bash ./build.sh
```

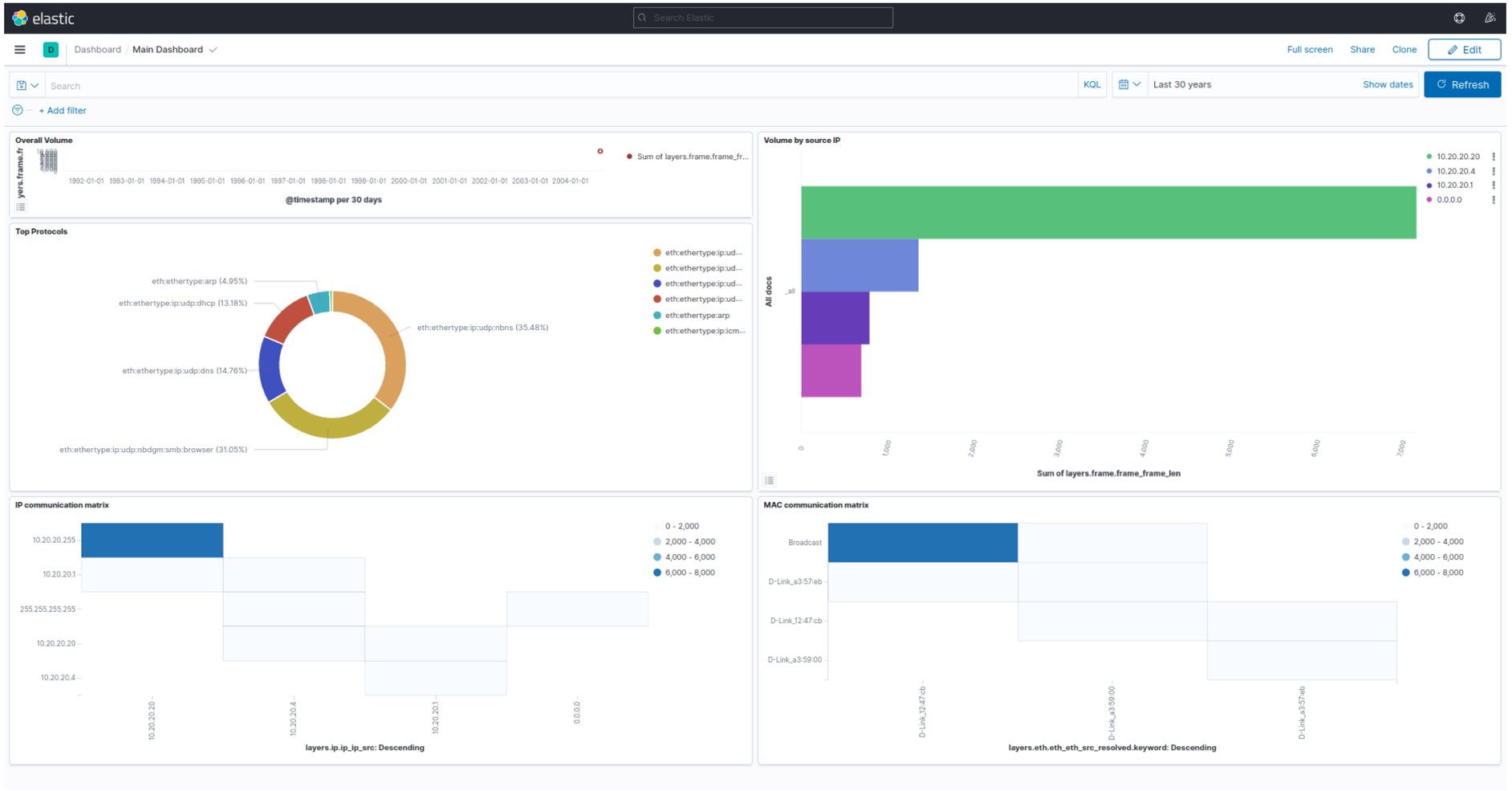
Upload pcaps to VM

```
# copy your pcaps into ./Trace
# run following script
bash upload_pcaps.sh

# or use tshark directly towards 127.0.0.1 17570/tcp
tshark -r trace.pcapng -x -T ek > /dev/tcp/localhost/17570
```

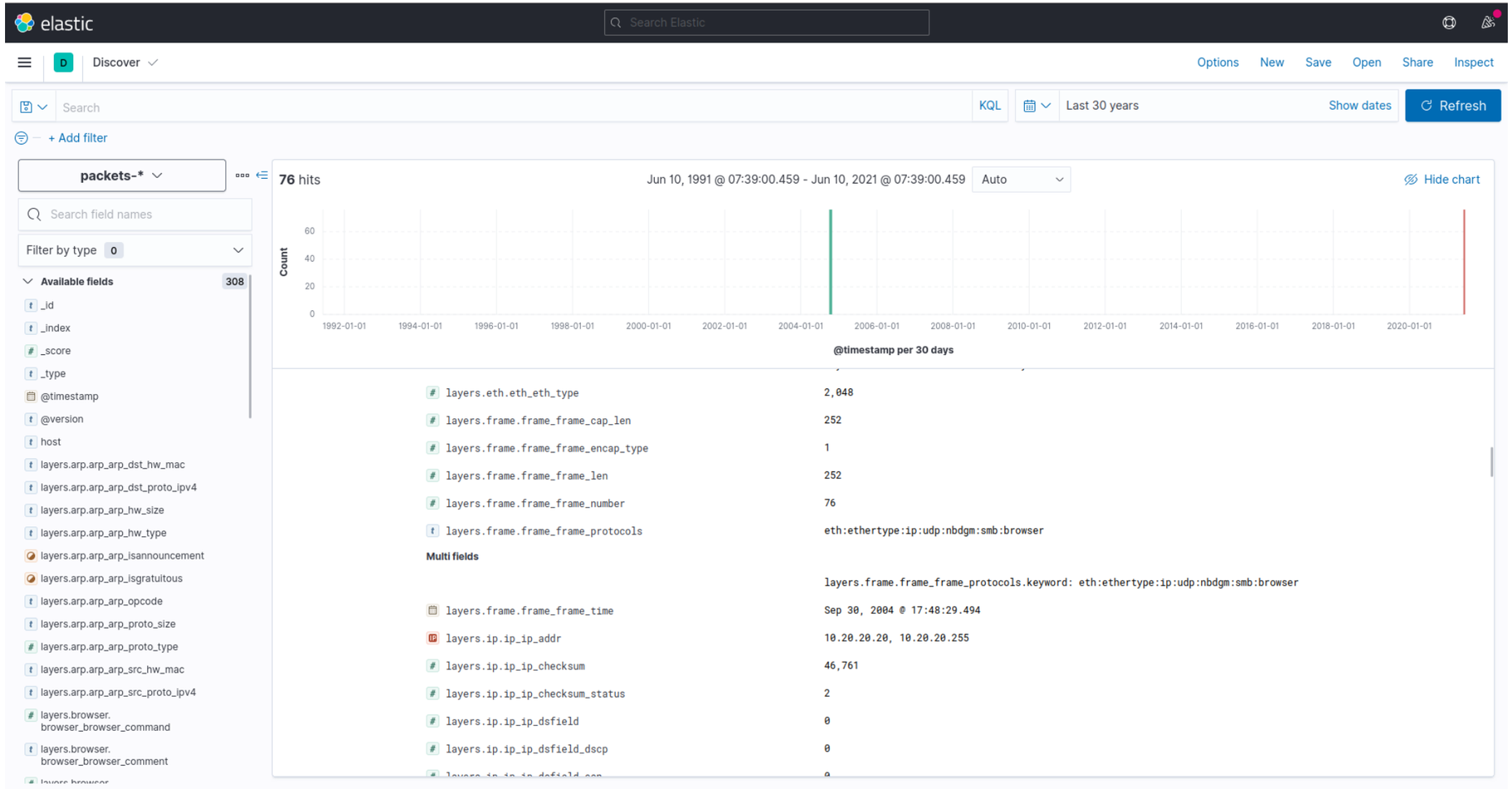
Open Kibana with browser

```
firefox http://127.0.0.1:15601/app/kibana#/dashboards
```

Kibana UI


elastic
Search Elastic
Dashboard / Main Dashboard
Full screen
Share
Clone
Edit
Search
KQL
Last 30 years
Show dates
Refresh
+ Add filter
Overall Volume
Sum of layers.frame.frame_fr...
@timestamp per 30 days
Top Protocols
eth:ethertype:arp (4.95%)
eth:ethertype:ip:udp:dhcp (13.18%)
eth:ethertype:ip:udp:nbns (35.48%)
eth:ethertype:ip:udp:dns (14.76%)
eth:ethertype:ip:udp:nbdgm:smb:browser (31.05%)
Volume by source IP
10.20.20.20
10.20.20.4
10.20.20.1
0.0.0.0
All docs
Sum of layers.frame.frame_frame_len
IP communication matrix
10.20.20.255
10.20.20.1
255.255.255.255
10.20.20.20
10.20.20.4
0 - 2,000
2,000 - 4,000
4,000 - 6,000
6,000 - 8,000
layers.ip.ip_ip_src: Descending
MAC communication matrix
Broadcast
D-Link_a3:57:eb
D-Link_12:47:cb
D-Link_a3:59:00
layers.eth.eth_eth_src_resolved.keyword: Descending

Kibana UI


elastic
Search Elastic
Discover
Options
New
Save
Open
Share
Inspect
Search
KQL
Last 30 years
Show dates
Refresh
+ Add filter
packets-*
76 hits
Jun 10, 1991 @ 07:39:00.459 - Jun 10, 2021 @ 07:39:00.459
Auto
Hide chart
Search field names
Filter by type 0
Available fields 308
_id
_index
_score
_type
@timestamp
@version
host
layers.arp.arp_arp_dst_hw_mac
layers.arp.arp_arp_dst_proto_ipv4
layers.arp.arp_arp_hw_size
layers.arp.arp_arp_hw_type
layers.arp.arp_arp_isannouncement
layers.arp.arp_arp_isgratuitous
layers.arp.arp_arp_opcode
layers.arp.arp_arp_proto_size
layers.arp.arp_arp_proto_type
layers.arp.arp_arp_src_hw_mac
layers.arp.arp_arp_src_proto_ipv4
layers.browser.browser_browser_command
layers.browser.browser_browser_comment
Count
60
40
20
0
1992-01-01
1994-01-01
1996-01-01
1998-01-01
2000-01-01
2002-01-01
2004-01-01
2006-01-01
2008-01-01
2010-01-01
2012-01-01
2014-01-01
2016-01-01
2018-01-01
2020-01-01
@timestamp per 30 days
layers.eth.eth_eth_type 2,048
layers.frame.frame_frame_cap_len 252
layers.frame.frame_frame_encap_type 1
layers.frame.frame_frame_len 252
layers.frame.frame_frame_number 76
layers.frame.frame_frame_protocols eth:ethertype:ip:udp:nbdgm:smb:browser
Multi fields
layers.frame.frame_frame_protocols.keyword: eth:ethertype:ip:udp:nbdgm:smb:browser
layers.frame.frame_frame_time Sep 30, 2004 @ 17:48:29.494
layers.ip.ip_ip_addr 10.20.20.20, 10.20.20.255
layers.ip.ip_ip_checksum 46,761
layers.ip.ip_ip_checksum_status 2
layers.ip.ip_ip_dsfield 0
layers.ip.ip_ip_dsfield_dscp 0

Further data processing

Custom processing:

- Logstash ruby filter
- Standalone application

Kibana capabilities:

- Analytics
- Alerts
- Anomaly detections
- Machine learning
- Others

Lessons learned

- Multiple tshark instances can run in parallel
- Better to process smaller pcaps (this is reducing impact if the tshark process crash)
- Indexing of all fields is useful only for small traffic samples (e.g. Threat Hunting to analyze small suspicious captures)
- For continues processing of traffic, the cost-efficient approach is the indexing of required fields only
- frame_raw (hex dump of raw packet) can be also stored in Elasticsearch to allow later packet reconstruction (this is possible for lower volume of traffic)
- tshark currently does not precisely identify the field location in protocol tree. E.g. for SS7 traffic the SCTP dechunking / preprocessing is required first

Real-World use cases examples

Telecom Monitoring

- Continuous processing of signaling
- Indexing of selected fields only
- No session correlation
- Backward pcap reconstruction possible with frame_raw stored in Elasticsearch

Telecom IDS

- Continuous processing of signaling
- Indexing of selected fields
- Logstash ruby plugin with security logic
- Session correlation using in-memory database
- Elasticsearch storing security alerts

Threat Hunting

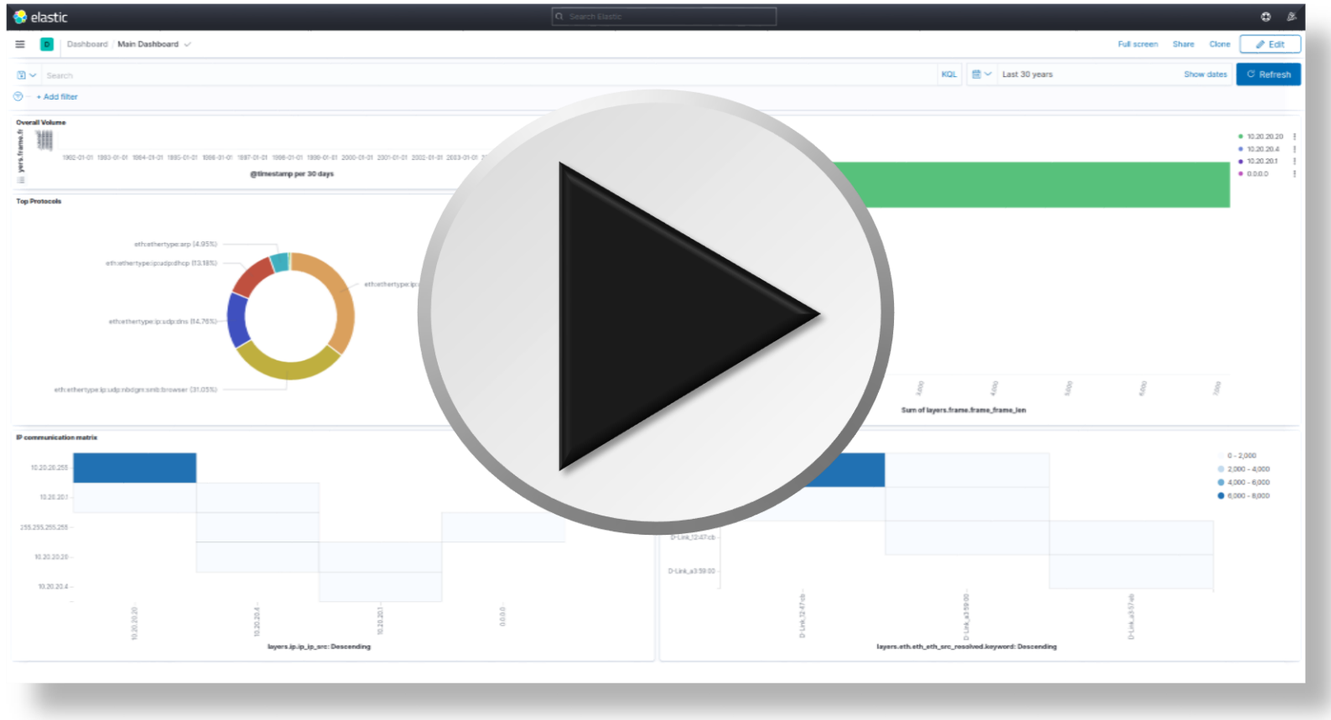
- Continuous processing of small suspicious traffic captures
- Indexing of all fields
- Creating high number of indices

Related work

https://www.h21lab.com/tools/json-to-pcap

https://www.h21lab.com/tools/anomaly-detection

Demo


elastic
Dashboard / Main Dashboard
Full screen
Share
Clone
Edit
Search
KQL
Last 30 years
Show dates
Refresh
+ Add filter
Overall Volume
@timestamp per 30 days
Top Protocols
eth:ethertype:arp (4.95%)
eth:ethertype:ip:udp:dhcp (13.18%)
eth:ethertype:ip:udp:dns (14.76%)
eth:ethertype:ip:udp:nbdgm:smb:browser (31.05%)
Sum of layers.frame.frame_frame_len
IP communication matrix
10.20.20.255
10.20.20.1
255.255.255.255
10.20.20.20
10.20.20.4
layers.ip.ip_ip_src: Descending
layers.eth.eth_eth_src_resolved.keyword: Descending
0 - 2,000
2,000 - 4,000
4,000 - 6,000
6,000 - 8,000

Q&A