Welcome to

10. Forensics 1: Auditing and Intrusion Detection

KEA Kompetence Computer Systems Security 2019

Henrik Lund Kramshøj hlk@zencurity.com @kramse

Slides are available as PDF, kramse@Github

10-forensics-auditing-intrusions.tex in the repo security-courses

Plan for today

Subjects

- Auditing and logging
- Volatility and file systems
- Intrusion Detection
- Host and Networks Based Intrusion Detection (HIDS/NIDS)
- Network Security Monitoring

Exercises

- Centralized syslogging and example system
- Open a file system dump

Reading Summary

Bishop chapter 25: Auditing

Bishop chapter 26: Intrusion Detection

And at least 27.4 - or chapter 27 too

Download and browse the ENISA papers listed under Computer Forensics in the reading list

Because tuesday was cancelled

The following pages will be removed from the curriculum:

Bishop chapter 17: Information Flow

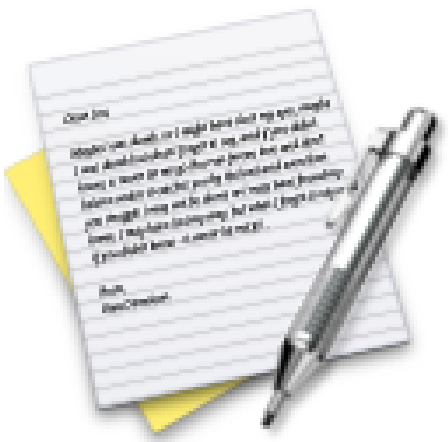
Bishop chapter 18: Confinement Problem

Capsicum: practical capabilities for UNIX

Removing ROP Gadgets from OpenBSD

but we will do the exercise Virtual Machine Escapes

Exercise



Now lets do the exercise

Virtual Machine Escapes 20min

which is number **20** in the exercise PDF.

Auditing and logging

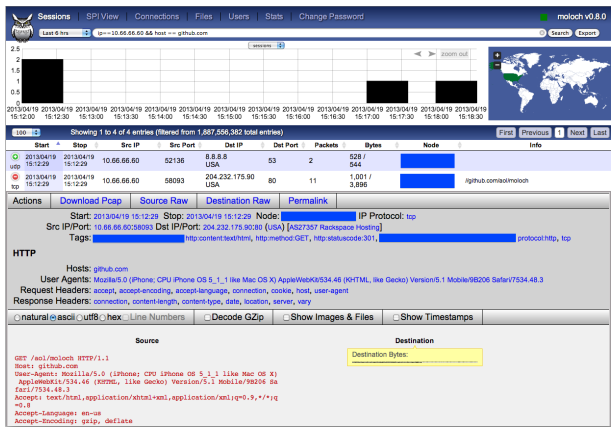
Volatility and file systems

Intrusion Detection

Host and Networks Based Intrusion Detection (HIDS/NIDS)

Network Security Monitoring

Logging today



Log analysis is required today - and we have many logs

Gather logs, parse logs, explain logs - fix stuff

Search your logs with the Elastic stack

Show sample logs from Suricata, Sudo, SSH, ... what we have

How to get started

How to get started searching for security events?

Collect basic data from your devices and networks

- Netflow data from routers
- Session data from firewalls
- Logging from applications: email, web, proxy systems

Centralize!

Process data

- Top 10: interesting due to high frequency, occurs often, brute-force attacks
- *ignore*
- Bottom 10: least-frequent messages are interesting

Centralized syslog

Logfiler er en nødvendighed for at have et transaktionsspor

Logfiler giver mulighed for statistik

Logfiler er desuden nødvendige for at fejlfinde

Det kan være relevant at sammenholde logfiler fra:

- routere
- firewalls
- webservere
- intrusion detection systemer
- adgangskontrolsystemer
- ...

Husk - tiden er vigtig! Network Time Protocol (NTP) anbefales

Husk at logfilerne typisk kan slettes af en angriber - hvis denne får kontrol med systemet

syslog

syslog er system loggen på UNIX og den er effektiv

- man kan definere hvad man vil se og hvor man vil have det dirigeret hen
- man kan samle det i en fil eller opdele alt efter programmer og andre kriterier
- man kan ligeledes bruge named pipes - dvs filer i filsystemet som tunneller fra chroot'ed services til syslog i det centrale system!
- man kan nemt sende data til andre systemer

Man bør lave en centraliseret løsning

syslogd.conf eksempel

```
*.err;kern.debug;auth.notice;authpriv.none;mail.crit    /dev/console
*.notice;auth,authpriv,cron,ftp,kern,lpr,mail,user.none /var/log/messages
kern.debug;user.info;syslog.info                        /var/log/messages
auth.info                                               /var/log/authlog
authpriv.debug                                          /var/log/secure
...
# Uncomment to log to a central host named "loghost".
#*.notice;auth,authpriv,cron,ftp,kern,lpr,mail,user.none        @loghost
#kern.debug,user.info,syslog.info                               @loghost
#auth.info,authpriv.debug,daemon.info                           @loghost
```

Andre syslogs syslog-ng

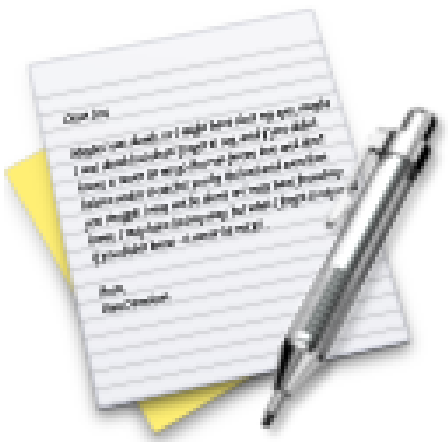
- der findes andre syslog systemer eksempelvis syslog-ng
- konfigureres gennem `/etc/syslog-ng/syslog-ng.conf`

```
options {
        long_hostnames(off);
        sync(0);
        stats(43200);
};

source src  unix-stream("/dev/log"); internal(); pipe("/proc/kmsg"); ;
destination messages  file("/var/log/messages"); ;
destination console_all  file("/dev/console"); ;
log  source(src); destination(messages); ;
log  source(src); destination(console_all); ;
```

Kan eksempelvis TCP og garanteret aflevering af beskeder

Exercise



Now lets do the exercise

Centralized syslog 15min

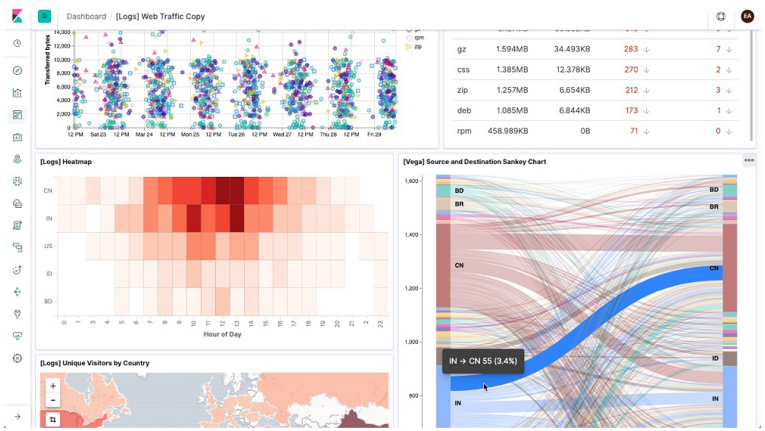
which is number 21 in the exercise PDF.

Web server access log

```
root# tail -f access_log
::1 - - [19/Feb/2004:09:05:33 +0100] "GET /images/IPv6ready.png
HTTP/1.1" 304 0
::1 - - [19/Feb/2004:09:05:33 +0100] "GET /images/valid-html401.png
HTTP/1.1" 304 0
::1 - - [19/Feb/2004:09:05:33 +0100] "GET /images/snowflake1.png
HTTP/1.1" 304 0
::1 - - [19/Feb/2004:09:05:33 +0100] "GET /~hlk/security6.net/images/logo-1.png
HTTP/1.1" 304 0
2001:1448:81:beef:20a:95ff:fef5:34df - - [19/Feb/2004:09:57:35 +0100]
"GET / HTTP/1.1" 200 1456
2001:1448:81:beef:20a:95ff:fef5:34df - - [19/Feb/2004:09:57:35 +0100]
"GET /apache_pb.gif HTTP/1.1" 200 2326
2001:1448:81:beef:20a:95ff:fef5:34df - - [19/Feb/2004:09:57:36 +0100]
"GET /favicon.ico HTTP/1.1" 404 209
2001:1448:81:beef:20a:95ff:fef5:34df - - [19/Feb/2004:09:57:36 +0100]
"GET /favicon.ico HTTP/1.1" 404 209
```

Web server logs are pretty standardized, common log format.

Big Data tools: Elasticsearch



Elasticsearch is an open source distributed, RESTful search and analytics engine capable of solving a growing number of use cases.

`https://www.elastic.co`

We are all Devops now, even security people!

Ansible configuration management

```
- apt: name= item  state=latest
  with_items:
        - unzip
        - elasticsearch
        - logstash
        - redis-server
        - nginx
- lineinfile: "dest=/etc/elasticsearch/elasticsearch.yml state=present
  regexp='script.disable_dynamic: true' line='script.disable_dynamic: true'"
- lineinfile: "dest=/etc/elasticsearch/elasticsearch.yml state=present
  regexp='network.host: localhost' line='network.host: localhost'"
- name: Move elasticsearch data into /data
  command: creates=/data/elasticsearch mv /var/lib/elasticsearch /data/
- name: Make link to /data/elasticsearch
  file: state=link src=/data/elasticsearch path=/var/lib/elasticsearch
```

only requires SSH+python `http://www.ansible.com`

Kibana

Highly recommended for a lot of data visualisation

Non-programmers can create, save, and share dashboards

Source: `https://www.elastic.co/products/kibana`

Logstash pipeline

> Logstash is an open source, server-side data processing pipeline that ingests data from a multitude of sources simultaneously, transforms it, and then sends it to your favorite "stash." (Ours is Elasticsearch, naturally.)
> `https://www.elastic.co/products/logstash`

```
input { stdin { } }
output {
  elasticsearch { host => localhost }
  stdout { codec => rubydebug }
}
```

- Logstash receives via **input**
- Processes with **filters** - grok
- Forward events with **output**

Logstash as SNMPtrap and syslog server

```
input {
  snmptrap {
    host => "0.0.0.0"
    type => "snmptrap"
    port => 1062
    community => "xxxxx"   }
  tcp {
    port => 5000
    type => syslog  }
  udp {
    port => 5000
    type => syslog  }
}
```

- We run logstash on port 5000 - but use IPtables port forwarding
- Have you even configured SNMP traps?
- Maybe you have a device sending SNMP traps right now ...

IPtables forwarding

```
*nat
:PREROUTING ACCEPT [0:0]
# redirect all incoming requests on port 514 to port 5000
-A PREROUTING -p tcp --dport 514 -j REDIRECT --to-port 5000
-A PREROUTING -p udp --dport 514 -j REDIRECT --to-port 5000
-A PREROUTING -p udp --dport 162 -j REDIRECT --to-port 1062
COMMIT
```

Inserted near beginning of /etc/ufw/before.rules on Ubuntu

Remember defense in depth, dont run a privileged Java VM process as root ☺

Grok expresssions

```
filter {
  if [type] == "syslog" {
    grok {
      match => { "message" => "%{SYSLOGTIMESTAMP:syslog_timestamp}
      %{SYSLOGHOST:syslog_hostname} %{DATA:syslog_program}
      (?:\[%{POSINT:syslog_pid}\])?: %{GREEDYDATA:syslog_message}" }
      add_field => [ "received_at", "%{@timestamp}" ]
      add_field => [ "received_from", "%{host}" ]
    }
    syslog_pri { }
    date {
      match => [ "syslog_timestamp", "MMM  d HH:mm:ss", "MMM dd HH:mm:ss" ]
    }
  }
}
```

- Logstash filter expressions grok can normalize and split data into fields

Source: Config snippet from recommended link

`http://logstash.net/docs/1.4.1/tutorials/getting-started-with-logstash`

Grok expresssions, sample from my archive

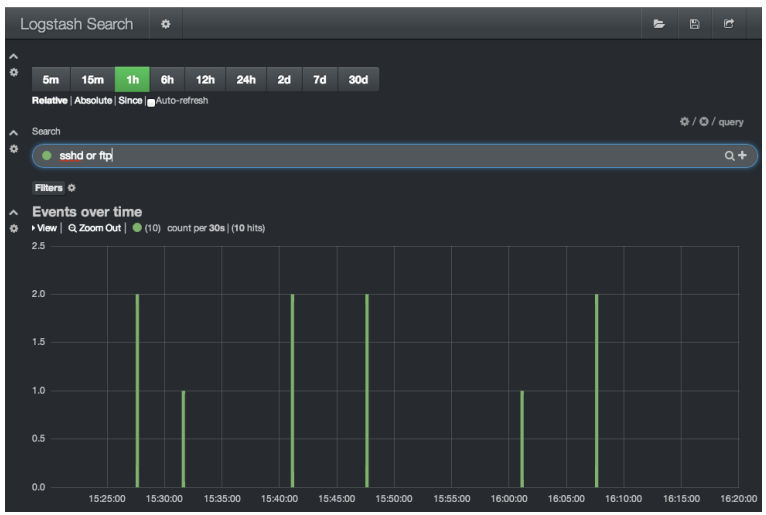
```
filter {
# decode some SSHD
if [syslog_program] == "sshd" {
  grok {
# May 20 10:27:08 odn1-nsm-01 sshd[4554]: Accepted publickey for hlk from
10.50.11.17 port 50365 ssh2: DSA 9e:fd:3b:3d:fc:11:0e:b9:bd:22:71:a9:36:d8:06:c7

match => { "message" => "%{SYSLOGTIMESTAMP:timestamp} %{HOSTNAME:host_target}
sshd\[%{BASE10NUM}\]: Accepted publickey for %{USERNAME:username} from
  %{IP:src_ip} port %{BASE10NUM:port} ssh2" }

# "May 20 10:27:08 odn1-nsm-01 sshd[4554]: pam_unix(sshd:session):
session opened for user hlk by (uid=0)"
match => { "message" => "%{SYSLOGTIMESTAMP:timestamp} %{HOSTNAME:host_target}
sshd\[%{BASE10NUM}\]: pam_unix\(sshd:session\): session opened for user
%{USERNAME:username}" }
```

- Logstash filter expressions grok can normalize and split data into fields

View data efficiently



View data by digging into it easily - must be fast

Logstash and Kibana are just examples, but use indexing to make it fast!

Other popular examples include Graylog and Grafana

Suricata with Dashboards

Picture from Twitter
`https://twitter.com/nullthreat/status/445969209840128000`

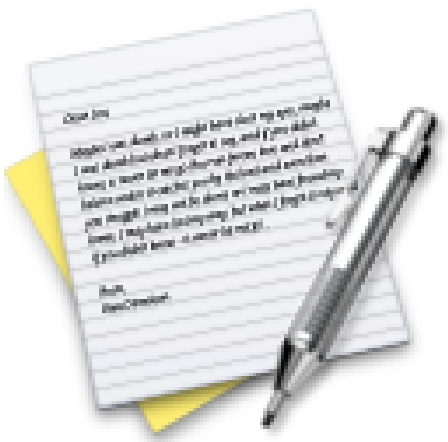
`http://suricata-ids.org/`

IP reputation

Zeek documentation Intel framework
`https://docs.zeek.org/en/stable/frameworks/intel.html`

Suricata reputation support
`https://suricata.readthedocs.io/en/suricata-4.0.5/reputation/index.html`

Exercise



Now lets do the exercise

Create Kibana Dashboard 15min

which is number 22 in the exercise PDF.

Collect Network Evidence from the network

Network Flows

Cisco standard NetFlow version 5 defines a flow as a unidirectional sequence of packets that all share the following 7 values:

- Ingress interface (SNMP ifIndex)
- Source IP address
- Destination IP address
- IP protocol
- Source port for UDP or TCP, 0 for other protocols
- Destination port for UDP or TCP, type and code for ICMP, or 0 for other protocols
- IP Type of Service

Source: `https://en.wikipedia.org/wiki/NetFlow`

Netflow

Netflow is getting more important, more data share the same links

Accounting is important

Detecting DoS/DDoS and problems is essential

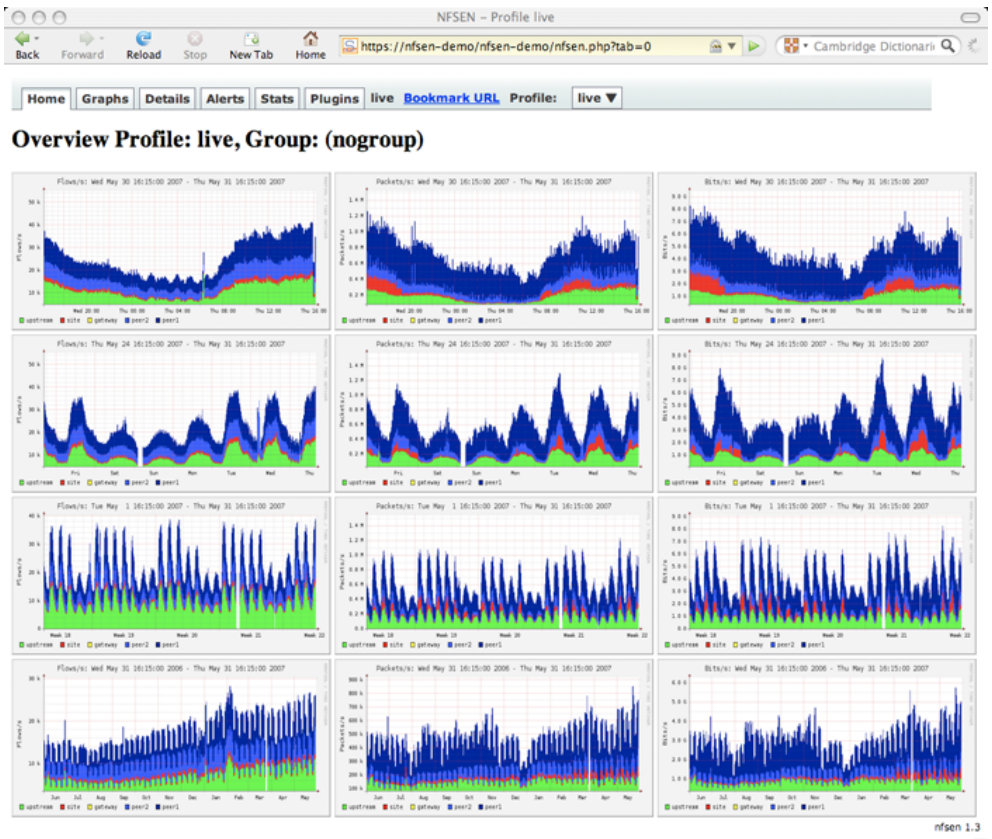
Netflow sampling is vital information - 123Mbit, but what kind of traffic

NFSen is an old but free application `http://nfsen.sourceforge.net/`

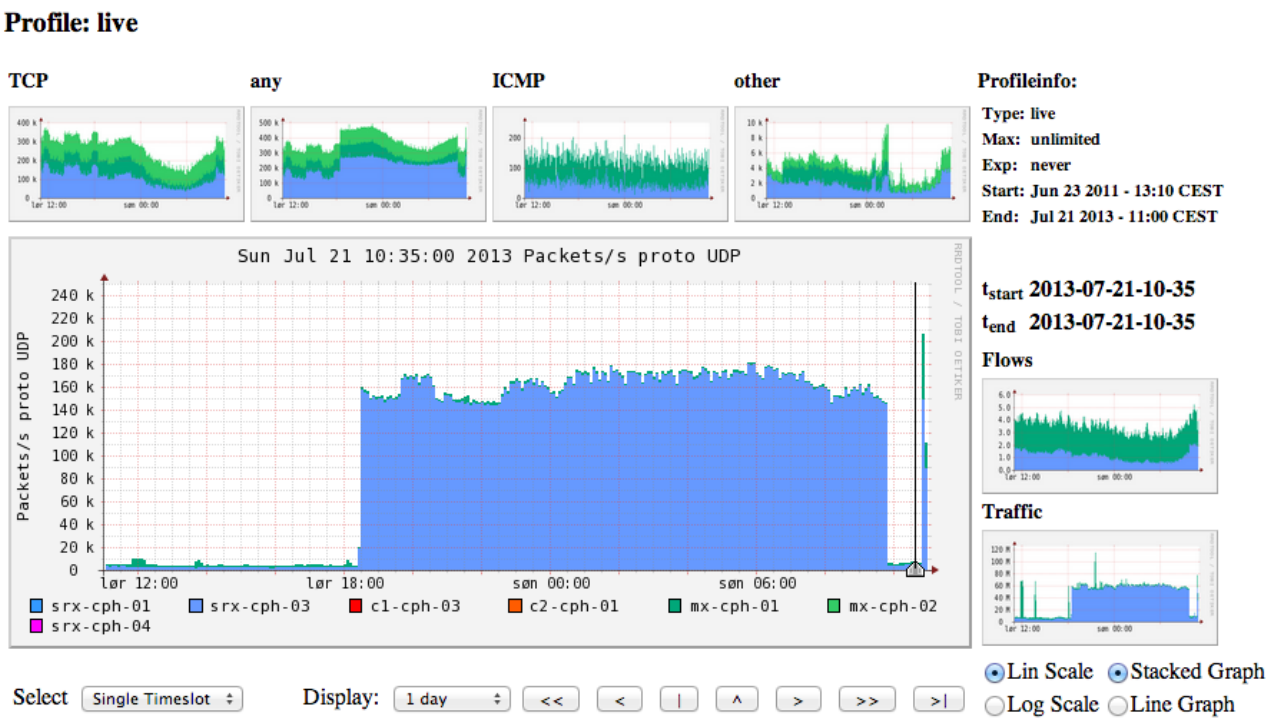
Currently also investigating sFlow - hopefully more fine grained

sFlow, short for "sampled flow", is an industry standard for packet export at Layer 2 of the OSI model,
`https://en.wikipedia.org/wiki/SFlow`

Netflow using NFSen

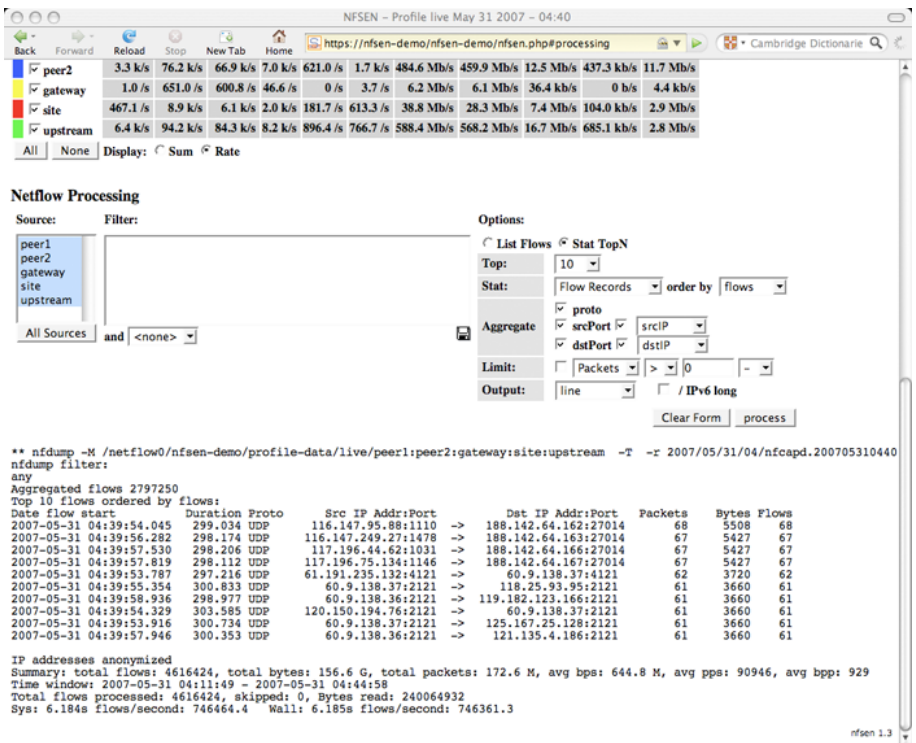


Netflow NFSen



An extra 100k packets per second from this netflow source (source is a router)

Netflow processing from the web interface



Bringing the power of the command line forward

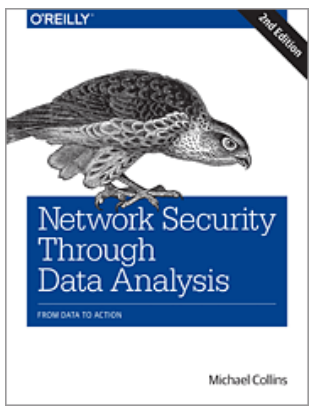
Next steps

Always improving things:

Suricata IDS `http://www.openinfosecfoundation.org/`
More graphs, with **automatic identification** of IPs under attack
Identification of **short sessions without data** - spoofed addresses
Alerting from **existing** devices
Dashboards with key measurements

Conclusion: Combine tools!

Network Security Through Data Analysis



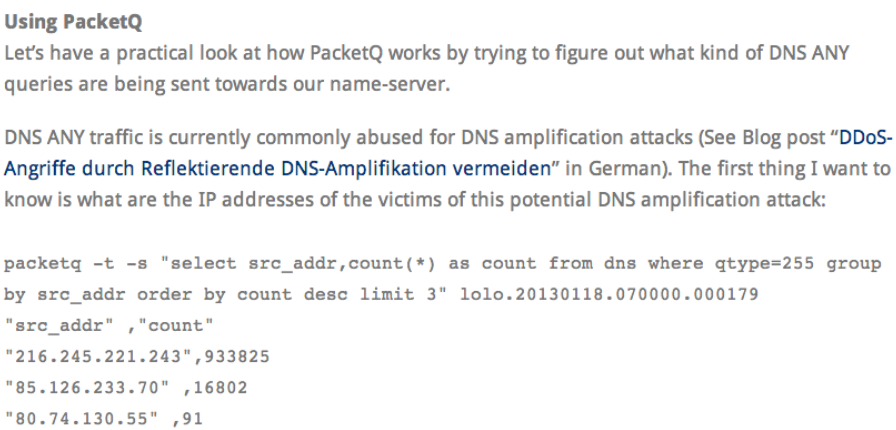
Low page count, but high value! Recommended.

Network Security through Data Analysis, 2nd Edition By Michael S Collins Publisher: O'Reilly Media 2015-05-01: Second release, 348 Pages

New Release Date: August 2017

And we have the ANSM book

Network tools - more examples



Using PacketQ

Let's have a practical look at how PacketQ works by trying to figure out what kind of DNS ANY queries are being sent towards our name-server.

DNS ANY traffic is currently commonly abused for DNS amplification attacks (See Blog post "DDoS-Angriffe durch Reflektierende DNS-Amplifikation vermeiden" in German). The first thing I want to know is what are the IP addresses of the victims of this potential DNS amplification attack:

```
packetq -t -s "select src_addr,count(*) as count from dns where qtype=255 group
by src_addr order by count desc limit 3" lolo.20130118.070000.000179
"src_addr" ,"count"
"216.245.221.243",933825
"85.126.233.70" ,16802
"80.74.130.55" ,91
```

- DNS: DSC and PacketQ `https://github.com/DNS-OARC/PacketQ`
- Packetbeat `https://www.elastic.co/products/beats/packetbeat`
- `http://securityblog.switch.ch/2013/01/22/using-packetq/`
- `http://jpmens.net/2013/05/27/server-agnostic-logging-of-dns-queries-responses/`

Network Forensics ENISA

The European Union Agency for Network and Information Security (ENISA) is a centre of expertise for cyber security in Europe.

ENISA is contributing to a high level of network and information security (NIS) within the European Union, by developing and promoting a culture of NIS in society to assist in the proper functioning of the internal market.

`https://www.enisa.europa.eu/`

ENISA has published a number of network forensics documents which are free to use, so these are our basics.

Forensic analysis

> Network forensics is a sub-branch of digital forensics relating to the monitoring and analysis of computer network traffic for the purposes of information gathering, legal evidence, or intrusion detection 5 .

Systems used to collect network data for forensics use usually come in three forms:

- Packet capture: All packets passing through a certain traffic point are captured and written to storage
- Intrusion detection systems
- Network flow sensors

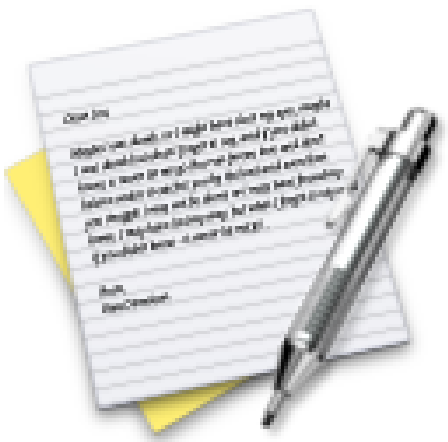
The acronym OSCAR 8 stands for: Obtain information, Strategize, Collect evidence, Analyse, Report

Source: Forensic analysis Network Incident Response Handbook, Document for teachers 1.0 DECEMBER 2016, ENISA `EXE2_Forensic_analysis_II-Handbook.pdf`

ENISA exercises

- We will use these as examples:
- ENISA Presenting, correlating and filtering various feeds Handbook, Document for teachers
  `https://www.enisa.europa.eu/topics/trainings-for-cybersecurity-specialists/online-training-m`
  `documents/presenting-correlating-and-filtering-various-feeds-handbook`
- ENISA Forensic analysis, Network Incident Response
  `https://www.enisa.europa.eu/topics/trainings-for-cybersecurity-specialists/online-training-m`
  `documents/2016-resources/exe2_forensic_analysis_ii-handbook`
- ENISA Network Forensics, Handbook, Document for teachers
  `https://www.enisa.europa.eu/topics/trainings-for-cybersecurity-specialists/online-training-m`
  `documents/network-forensics-handbook`

Exercise

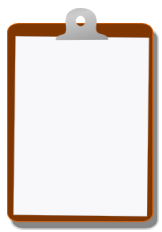


Now lets do the exercise

File System Forensics

which is number 23 in the exercise PDF.

For Next Time



Think about the subjects from this time, write down questions

Check the plan for chapters to read in the books
Most days have less than 100 pages, but some days may have more!

Visit web sites and download papers if needed

Retry the exercises to get more confident using the tools