Suricata, the Terminator of IDS/IPS world

Éric Leblond

OISF

July 9, 2013

Some word about me

Eric Leblond

- French
- Previously, co-founder and CTO of EdenWall (RIP)
- Now, Contractor
- Suricata IDS/IPS developer
- @Regiteric on Twitter

Some word about me

Eric Leblond

- French
- Previously, co-founder and CTO of EdenWall (RIP)
- Now, Contractor
- Suricata IDS/IPS developer
- @Regiteric on Twitter

regit@netfilter.org

- Netfilter Coreteam Member
- Working on:
  - some kernel stuff
  - libnetfilter_queue and userspace library
  - ulogd2 maintainer

IDS? IPS?

System to uncover malicious/unwanted activity on your network by inspecting the network traffic.

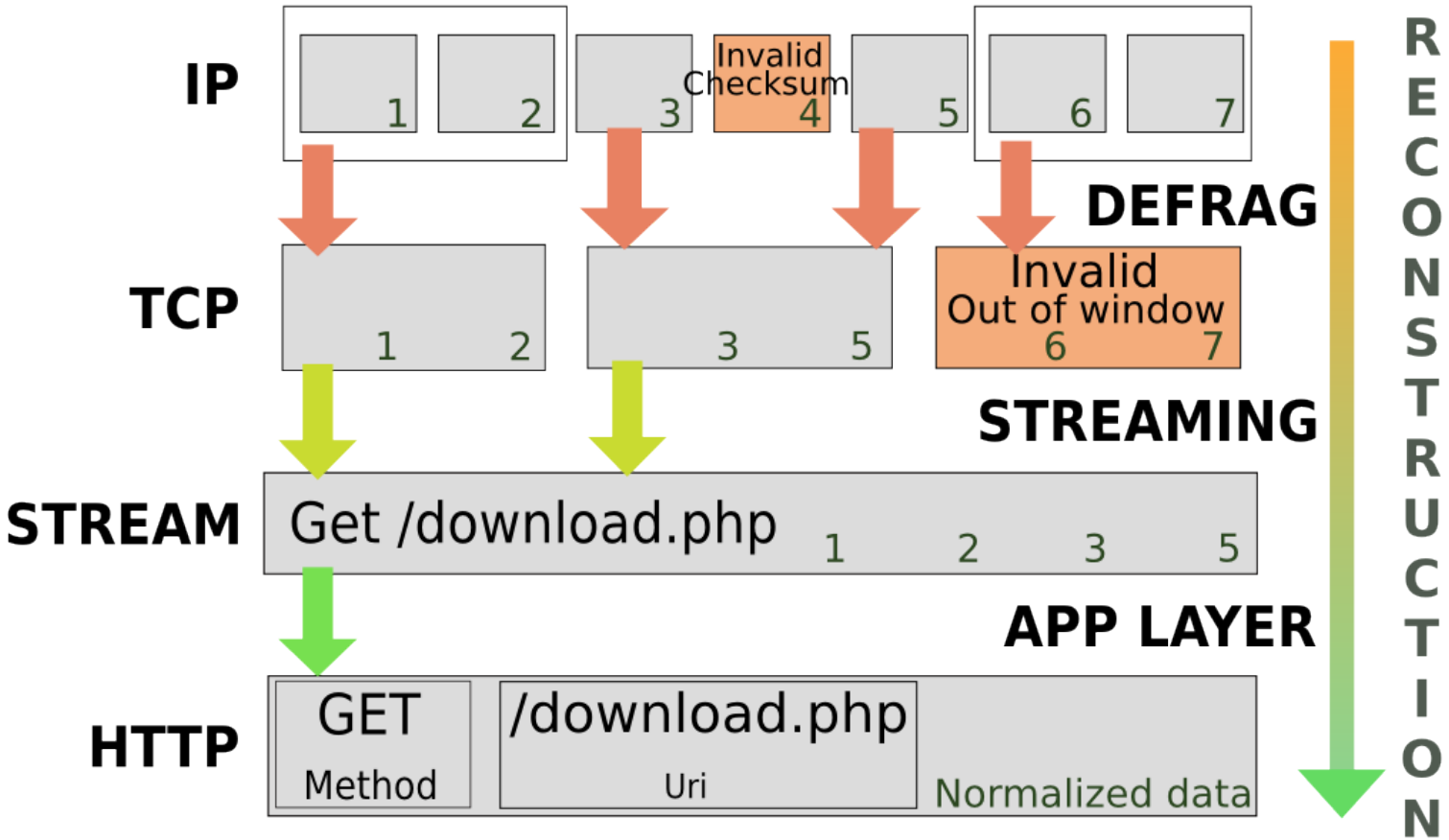
IDS

- (Network) Intrusion *Detection* System
- Passive, it only looks and alerts the admin
- Compare to security camera

IPS

- (Network) Intrusion *Prevention* System
- Active, tries to prevent badness from happening
- Compare to security checkpoint

Suricata reconstruction and normalization



https://home.regit.org/~regit/decomp-en.svg

Similar projects

Bro

- Different technology (capture oriented)
- Statistical study
- Scripting
- Complementary

Snort

- Equivalent
- Compatible
- Competing project

Suricata vs Snort

Suricata

- Driven by a foundation
- Multi-threaded
- Native IPS
- Advanced functions (flowint, libHTP, LuaJIT scripting)
- PF_RING support, CUDA support
- Modern and modular code
- Young but dynamic

Snort

- Developed by Sourcefire
- Multi-process
- IPS support
- SO ruleset (advanced logic + perf but closed)
- No hardware acceleration
- Old code
- 10 years of experience

Independant study:

`http://www.aldeid.com/index.php/Suricata-vs-snort`

Suricata with Snort ruleset

- Not optimised
- Don’t use any advanced features

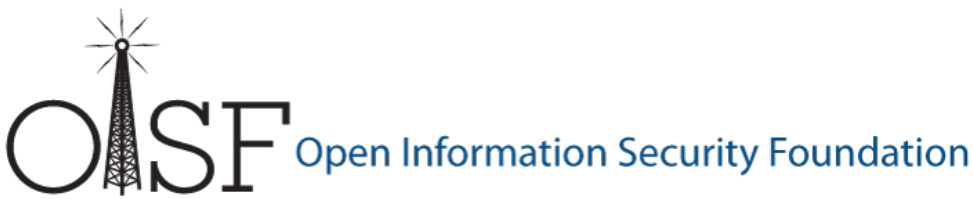
Suricata with dedicated ruleset

- Uses Suricata optimised detection
- Uses Suricata advanced keywords
- Can get one for free from
  `http://www.emergingthreats.net/`

About OISF

Open Information Security Foundation

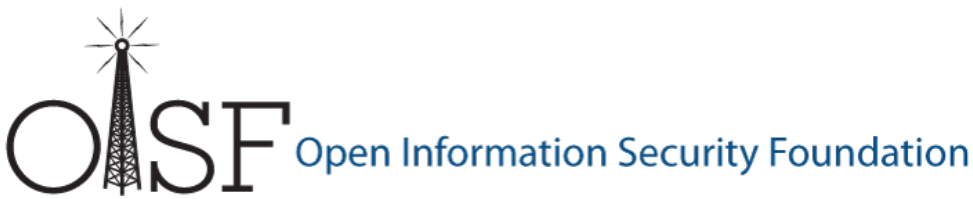
- `http://www.openinfosecfoundation.org`
- Non-profit foundation organized to build a next generation IDS/IPS engine
- Funded by US Governement (DHS, Navy)
- Development of an Open Source IDS/IPS:



About OISF

Open Information Security Foundation

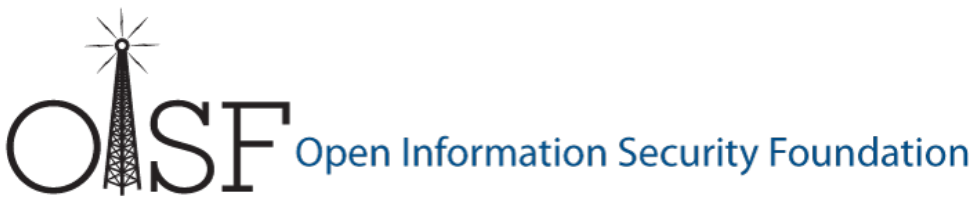
- `http://www.openinfosecfoundation.org`
- Non-profit foundation organized to build a next generation IDS/IPS engine
- Funded by US Governement (DHS, Navy)
- Development of an Open Source IDS/IPS:
  - Paying Developers



About OISF

Open Information Security Foundation

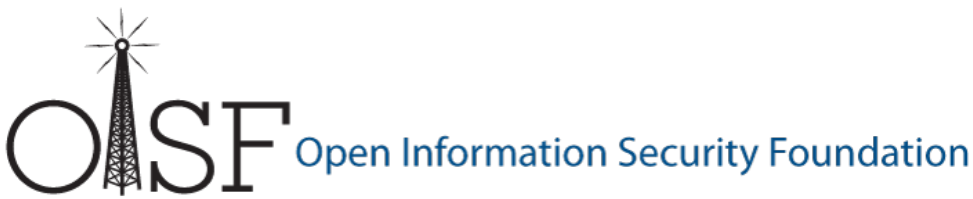
- `http://www.openinfosecfoundation.org`
- Non-profit foundation organized to build a next generation IDS/IPS engine
- Funded by US Governement (DHS, Navy)
- Development of an Open Source IDS/IPS:
  - Paying Developers
  - Financial support of related projects (barnyard2)



About OISF

Open Information Security Foundation

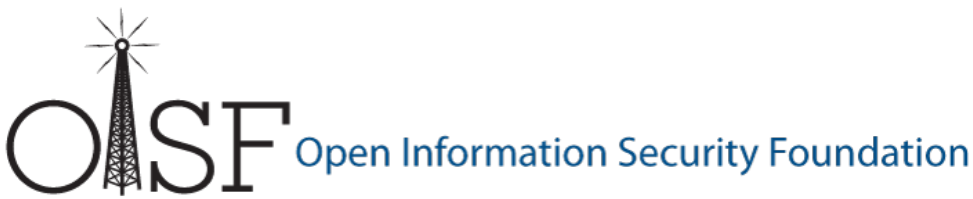
- `http://www.openinfosecfoundation.org`
- Non-profit foundation organized to build a next generation IDS/IPS engine
- Funded by US Governement (DHS, Navy)
- Development of an Open Source IDS/IPS:
  - Paying Developers
  - Financial support of related projects (barnyard2)
  - Board which oversees foundation management



About OISF

Open Information Security Foundation

- `http://www.openinfosecfoundation.org`
- Non-profit foundation organized to build a next generation IDS/IPS engine
- Funded by US Governement (DHS, Navy)
- Development of an Open Source IDS/IPS:
  - Paying Developers
  - Financial support of related projects (barnyard2)
  - Board which oversees foundation management
  - Roadmap is defined in public brainstorm sessions



About OISF

- Consortium members
  - HOST program: Homeland Open Security Technology
  - Platinium level: BAE Systems, nPulse
  - Gold level: Tilera, Endace, Emerging Threats
  - Bronze level: SRC, Everis, NitroSecurity, Myricom, Emulex
  - Technology partner: Napatech, Nvidia

About OISF

- Consortium members
  - HOST program: Homeland Open Security Technology
  - Platinium level: BAE Systems, nPulse
  - Gold level: Tilera, Endace, Emerging Threats
  - Bronze level: SRC, Everis, NitroSecurity, Myricom, Emulex
  - Technology partner: Napatech, Nvidia
- Developers
  - Lead: Victor Julien
  - Core Developers: Anoop Saldanha, Eric Leblond
  - Developers: serveral from consortium members, community.
  - Suricata has been created by about 35 developers so far.

About OISF

- Consortium members
  - HOST program: Homeland Open Security Technology
  - Platinium level: BAE Systems, nPulse
  - Gold level: Tilera, Endace, Emerging Threats
  - Bronze level: SRC, Everis, NitroSecurity, Myricom, Emulex
  - Technology partner: Napatech, Nvidia
- Developers
  - Lead: Victor Julien
  - Core Developers: Anoop Saldanha, Eric Leblond
  - Developers: serveral from consortium members, community.
  - Suricata has been created by about 35 developers so far.
- Board
  - Project leader: Matt Jonkman
  - Richard Bejtlich, Dr. Jose Nazario, Joel Ebrahimi, Marc Norton, Stuart Wilson

Goals

- Bring new technologies to IDS
- Performance: Multi-Threading, Hardware acceleration
- Open source: community driven (GPLv2)
- Support of Linux / *BSD / Mac OSX / Windows

Features

- IPv6 native support

Features

- IPv6 native support
- Multi-threaded

Features

- IPv6 native support
- Multi-threaded
- Native hardware acceleration (PF_RING, Napatech, Endace, Myricom)

Features

- IPv6 native support
- Multi-threaded
- Native hardware acceleration (PF_RING, Napatech, Endace, Myricom)
- Numerous options for performance optimisation

Features

- IPv6 native support
- Multi-threaded
- Native hardware acceleration (PF_RING, Napatech, Endace, Myricom)
- Numerous options for performance optimisation
- Optimized support of IP only tests

Features

- IPv6 native support
- Multi-threaded
- Native hardware acceleration (PF_RING, Napatech, Endace, Myricom)
- Numerous options for performance optimisation
- Optimized support of IP only tests
- IPS is native (inline mode)

Features

- IPv6 native support
- Multi-threaded
- Native hardware acceleration (PF_RING, Napatech, Endace, Myricom)
- Numerous options for performance optimisation
- Optimized support of IP only tests
- IPS is native (inline mode)
- Protocol detection

Features

- IPv6 native support
- Multi-threaded
- Native hardware acceleration (PF_RING, Napatech, Endace, Myricom)
- Numerous options for performance optimisation
- Optimized support of IP only tests
- IPS is native (inline mode)
- Protocol detection
- Advanced HTTP and TLS support

Features

- IPv6 native support
- Multi-threaded
- Native hardware acceleration (PF_RING, Napatech, Endace, Myricom)
- Numerous options for performance optimisation
- Optimized support of IP only tests
- IPS is native (inline mode)
- Protocol detection
- Advanced HTTP and TLS support
- File extraction

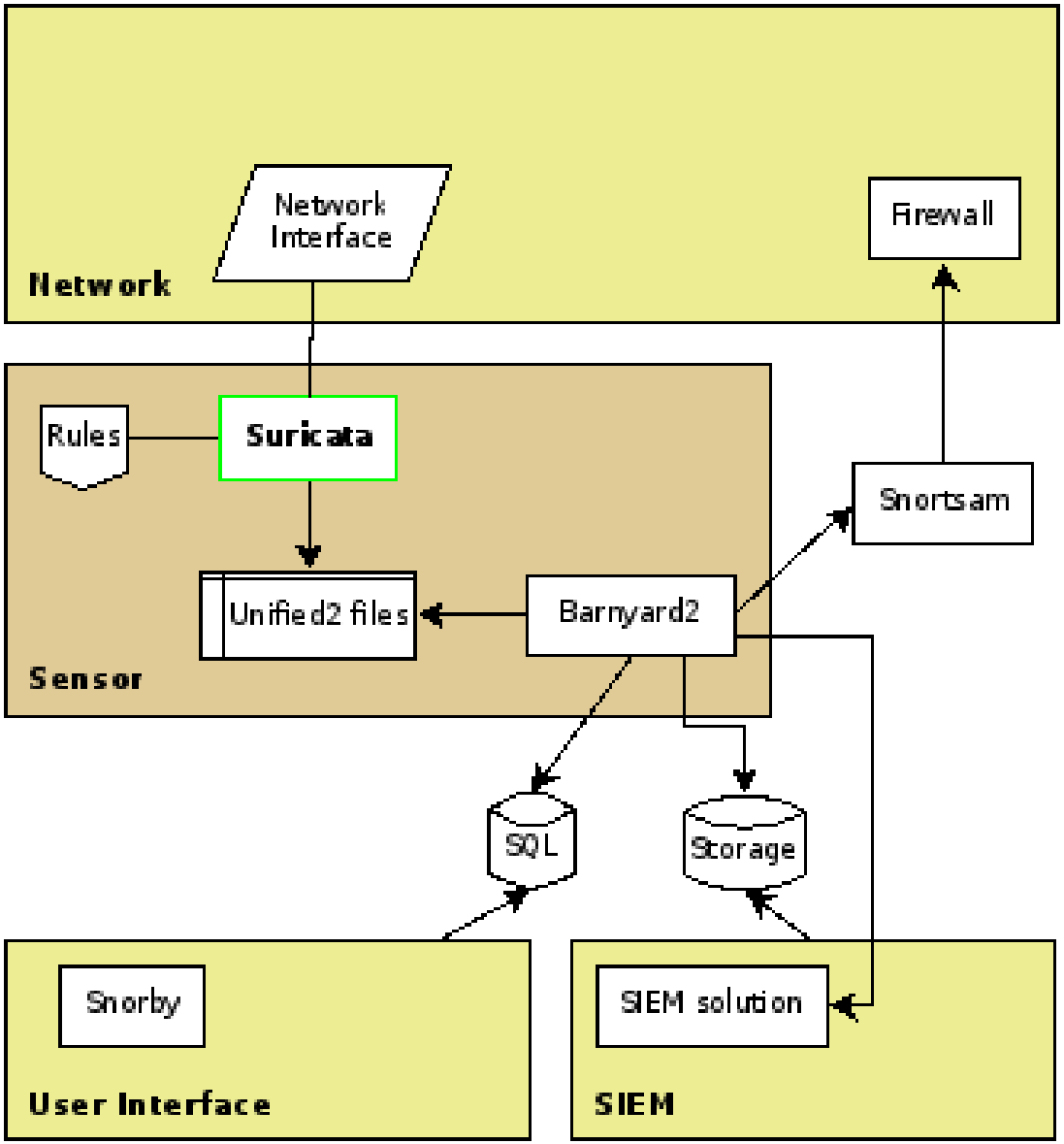
Features

- IPv6 native support
- Multi-threaded
- Native hardware acceleration (PF_RING, Napatech, Endace, Myricom)
- Numerous options for performance optimisation
- Optimized support of IP only tests
- IPS is native (inline mode)
- Protocol detection
- Advanced HTTP and TLS support
- File extraction
- LuaJIT scripting (experimental)

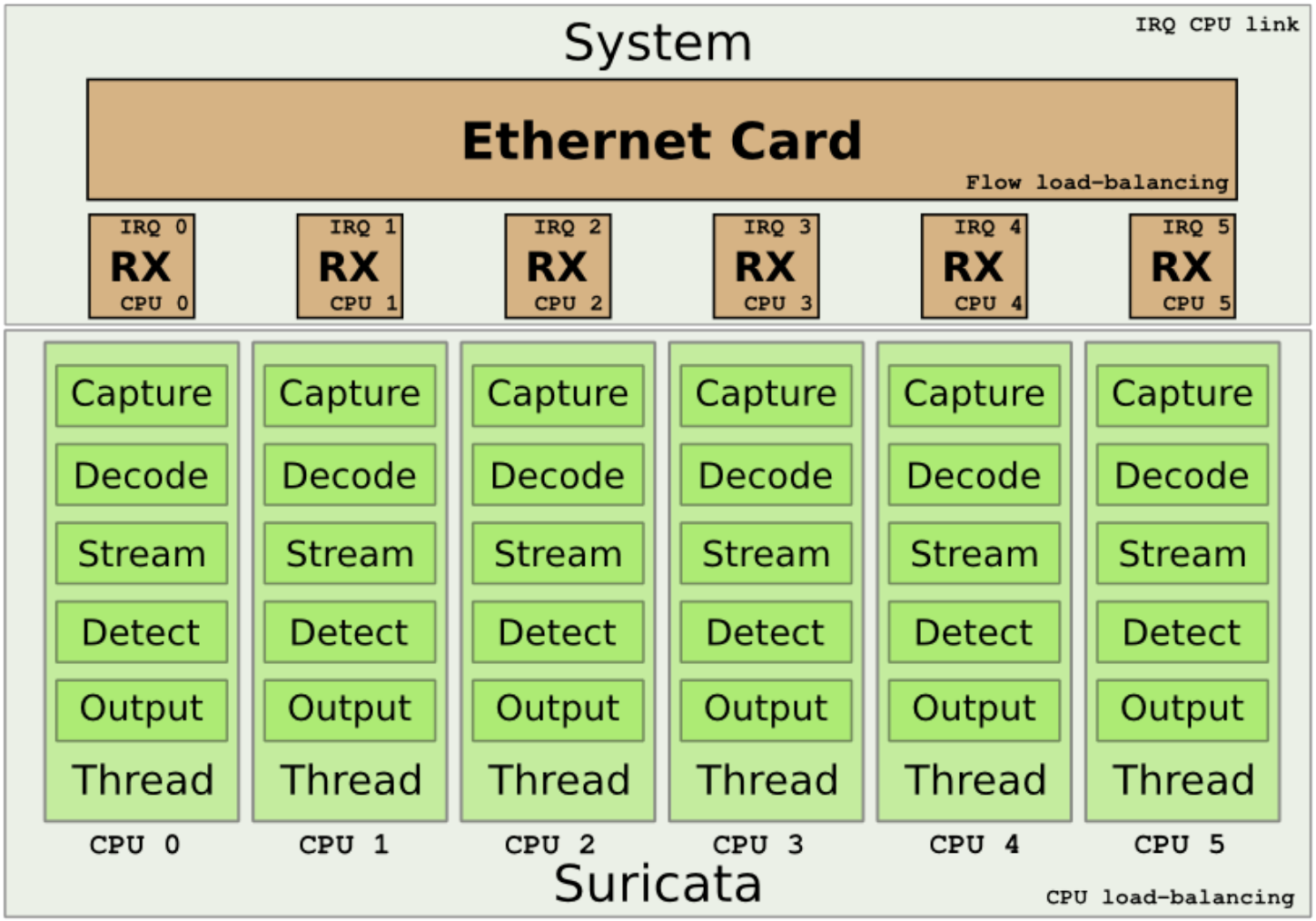
Features

- IPv6 native support
- Multi-threaded
- Native hardware acceleration (PF_RING, Napatech, Endace, Myricom)
- Numerous options for performance optimisation
- Optimized support of IP only tests
- IPS is native (inline mode)
- Protocol detection
- Advanced HTTP and TLS support
- File extraction
- LuaJIT scripting (experimental)
- IP Reputation and GeoIP

Suricata Ecosystem



Example of high performance Suricata setup



Entry modules

IDS

- PCAP
  - live, multi interface
  - offline support
- AF_PACKET
- PF_RING: kernel level, `http://www.ntop.org/PF_RING.html`
- Capture card support: Napatech, Myricom, Endace

Entry modules

IDS

- PCAP
  - live, multi interface
  - offline support
- AF_PACKET
- PF_RING: kernel level, `http://www.ntop.org/PF_RING.html`
- Capture card support: Napatech, Myricom, Endace

IPS

- NFQueue:
  - Linux: multi-queue, advanced support
- AF_PACKET:
  - Linux: bridge
- ipfw :
  - FreeBSD, NetBSD, Mac OSX

Output modules

- Fastlog (simple alerts)
- Unified2 log (full alerts, Barnyard2)
- HTTP log (log in apache-style format)
- TLS log (log certs)
- Pcap log (full packet capture to disk)
- Prelude (IDMEF)
- File log (files transfered over HTTP)

libhtp

- Security oriented HTTP parser
- Written by Ivan Ristić (ModSecurity, IronBee)
- Support of several keywords
  - http_method
  - http_uri & http_raw_uri
  - http_client_body & http_server_body
  - http_header & http_raw_header
  - http_cookie
  - serveral more...
- Able to decode gzip compressed flows

Using HTTP features in signature

Signature example: Chat facebook

```
alert http $HOME_NET any -> $EXTERNAL_NET $HTTP_PORTS \
(
 msg:"ET CHAT Facebook Chat (send message)"; \
 flow:established,to_server; content:"POST"; http_method; \
 content:"/ajax/chat/send.php"; http_uri; content:"facebook.com"; http_header; \
 classtype:policy-violation; reference:url,doc.emergingthreats.net/2010784; \
 reference:url,www.emergingthreats.net/cgi-bin/cvsweb.cgi/sigs/POLICY/POLICY_Facebook_Chat; \
 sid:2010784; rev:4; \
)
```

This signature tests:

- The HTTP method: *POST*
- The page: */ajax/chat/send.php*
- The domain: *facebook.com*

Extraction and inspection of files

- Get files from HTTP downloads and uploads
- Detect information about the file using libmagic
  - Type of file
  - Other details
  - Author (if available)
- A dedicated extension of signature language
- SMTP support coming soon

Dedicated keywords

- *filemagic* : description of content

```
alert http any any -> any any (msg:"windows exec"; \
                               filemagic:"executable for MS Windows"; sid:1; rev:1;)
```

- *filestore* : store file for inspection

```
alert http any any -> any any (msg:"windows exec";
                               filemagic:"executable for MS Windows"; \
                               filestore; sid:1; rev:1;)
```

- *fileext* : file extension

```
alert http any any -> any any (msg:"jpg claimed, but not jpg file"; \
                               fileext:"jpg"; \
                               filemagic:!"JPEG image data"; sid:1; rev:1;)
```

- *filename* : file name

```
alert http any any -> any any (msg:"sensitive file leak";
                               filename:"secret"; sid:1; rev:1;)
```

Examples

- Files sending on a server only accepting PDF

```
alert http $EXTERNAL_NET -> $WEBSERVER any (msg:"suspicious upload"; \
        flow:established,to_server; content:"POST" http_method; \
        content:"/upload.php"; http_uri; \
        filemagic:!"PDF document"; \
        filestore; sid:1; rev:1;)
```

- Private keys in the wild

```
alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"outgoing private key"; \
        filemagic:"RSA private key"; sid:1; rev:1;)
```

Disk storage

- Every file can be stored to disk
- with a metadata file

```
TIME:              10/02/2009-21:34:53.796083
PCAP PKT NUM:      5678
SRC IP:            61.191.61.40
DST IP:            192.168.2.7
PROTO:             6
SRC PORT:          80
DST PORT:          1091
FILENAME:          /ww/aa5.exe
MAGIC:             PE32 executable for MS Windows (GUI)
                   Intel 80386 32-bit
STATE:             CLOSED
SIZE:              30855
```

- Disk usage limit can be set
- Scripts for looking up files / file md5's at Virus Total and others

1. Suricata
   - Ecosystem
   - Goals of the project
   - Features
   - Advanced functionalities

2. IPS
   - IPS basics
   - WTF

3 major modes

Netfilter

- Use libnetfilter_queue and NFQUEUE
- Verdict packet redirected by iptables rules
- Up-to-date support
- Maximum around 5Gb/s

ipfw

- Use divert socket
- Dedicated filtering rules must be added

AF_PACKET

- Use Linux capture
- Ethernet transparent mode
- Experimental

Rules management

The transformation

- Make some rules start with *drop* instead of *alert*
- A selection must be made

Tool usage

- Rules are updated
- A tool is needed to have modifications resist to update
- Pulledpork: `http://code.google.com/p/pulledpork/`
- oinkmaster: `http://oinkmaster.sourceforge.net/`

WTF: Word Termination Feature

Objective

- Fight against Word file transfer
- Because it is Office is heavy like hell
- And you even have to pay for it

Method

- Mark packet when a Word file is transferred
- Limit bandwith with Linux QoS

WTF: Waiting Transfer to Finish

Suricata configuration

The rule

```
alert http any any -> any any ( \
  msg: "Microsoft Word upload"; \
  nfq_set_mark:0x1/0x1; \
  filemagic:"Composite Document File V2 Document"; \
  sid:666 ; rev:1;)
```

Running suricata

```
suricata -q 0 -S word.rules
```

Netfilter configuration (1/2)

Queueing packets

```
iptables -I FORWARD -p tcp --dport 80 -j NFQUEUE
iptables -I FORWARD -p tcp --sport 80 -j NFQUEUE
# iptables -I OUTPUT -p tcp --dport 80 -j NFQUEUE
# iptables -I INPUT -p tcp --sport 80 -j NFQUEUE
```

Analysing packets

- Suricata needs to get all packets
- Get all packets in both way
- NFQUEUE is a terminal target

Netfilter configuration (2/2)

Propagating the mark

- Mark is set on packet
- We want to mark all packet of a connection
- We need to propagate the mark
- CONNMARK target is made for that

Using CONNMARK

```
iptables -A PREROUTING -t mangle -j CONNMARK --restore-mark
iptables -A POSTROUTING -t mangle -j CONNMARK --save-mark
# iptables -A OUTPUT -t mangle -j CONNMARK --restore-mark
```

One slide of QoS

A diffserv implementation

- Controlling how packets are sent
  - Reordering the queue
  - Introducing delay
  - Dropping packets
- Different algorithm available
  - Queueless: fifo, prio
  - With queue: cbq, htb, ...

HTB example

- Split bandwith in different part
- Assign to part
  - Minimum guarantee bandwith
  - Maximum bandwith
  - Priority

Linux QoS configuration

Setting up QoS tree

```
tc qdisc add dev eth0 root \
        handle 1: htb default 0
tc class add dev eth0 parent 1: \
        classid 1:1 htb \
        rate 1kbps ceil 1kbps
```

Sending marked packets to their fate

```
tc filter add dev eth0 parent 1: \
        protocol ip prio 1 \
        handle 1 fw flowid 1:1
```

Evasion technique

What would you test to avoid this

- Change file extension
- Send compressed file

Filename extension change

- Most likely to happen
- Easy to spot in the IDS

Detecting evasion technique

Detecting the evasion

```
alert http any any -> any any ( \
  msg:"Tricky Microsoft Word upload"; \
  nfq_set_mark:0x2/0x2; \
  fileext:!"doc"; \
  filemagic:"Composite Document File V2 Document"; \
  filestore; \
  sid:667; rev:1;)
```

Being nice with clever people

```
tc class add dev eth0 parent 1: classid 1:2 htb \
rate 10kbps ceil 10kbps
tc filter add dev eth0 parent 1: protocol ip \
prio 1 handle 2 fw flowid 1:2
```

Watching the clever ones (1/2)

Watching the clever one from behind a PRISM

- Getting the most information possible about the clevers
- Storing in a pcap file all their trafic for a certain amount of time

Watching the clever ones (1/2)

Watching the clever one from behind a PRISM

- Getting the most information possible about the clevers
- Storing in a pcap file all their trafic for a certain amount of time

Difficulty

- We've got a mark on the connection and we want to keep all trafic
- We need a method to pass from connection to IP

Watching the clever ones (1/2)

Watching the clever one from behind a PRISM

- Getting the most information possible about the clevers
- Storing in a pcap file all their trafic for a certain amount of time

Difficulty

- We've got a mark on the connection and we want to keep all trafic
- We need a method to pass from connection to IP

A possible method: ipset + ulogd

- ipset allows set handling
- set can be list of IPs with timeout
- it is possible to update a set from an iptables rules
- we can populate a set
- log all packets from the set to a pcap file with ulogd

One slide of Ipset

Efficient set handling

- Allow fast and efficient update of ruleset
- Define set that can match and be update fast
- Different type of set
  - bitmap:ip
  - hash:net
  - hash:ip,port,ip
  - ...

Component

- ipset: command line utility to maintain the set
- set match: do matching on the set
- SET target: update set if rule match

Watching the clever ones (2/2)

Using ipset to mark packets

```
ipset create cheaters hash:ip timeout 3600
iptables -A POSTROUTING -t mangle -m mark \
    --mark 0x2/0x2 \
    -j SET --add-set cheaters src --exists
```

Logging marked packets

```
iptables -A PREROUTING -t raw \
    -m set --match-set cheaters src,dst \
    -j NFLOG --nflog-group 1
```

Ulogd to keep the trace

Ulogd2

- Netfilter logging daemon
- Inputs: NFLOG, NFCT, NFACCT, ...
- Outputs: syslog, file, DB, pcap, ...

Ulogd to keep the trace

Ulogd2

- Netfilter logging daemon
- Inputs: NFLOG, NFCT, NFACCT, ...
- Outputs: syslog, file, DB, pcap, ...

Configuring ulogd

- Ulogd will log packets to a pcap file
- We need to activate a stack in ulogd.conf:

```
plugin="/usr/local/lib/ulogd/ulogd_output_PCAP.so"
stack=log2:NFLOG,base1:BASE,pcap1:PCAP
```

Ulogd to keep the trace

Ulogd2

- Netfilter logging daemon
- Inputs: NFLOG, NFCT, NFACCT, ...
- Outputs: syslog, file, DB, pcap, ...

Configuring ulogd

- Ulogd will log packets to a pcap file
- We need to activate a stack in ulogd.conf:

```
plugin="/usr/local/lib/ulogd/ulogd_output_PCAP.so"
stack=log2:NFLOG,base1:BASE,pcap1:PCAP
```

Starting ulogd

```
ulogd -c ulogd.conf
```

Questions

Do you have any questions?

Thanks to

- RMLL team for accepting this conference
- All Netfilter developers for their cool work

More information

- Suricata website: `http://www.suricata-ids.org/`
- Netfilter: `http://www.netfilter.org/`
- Ipset: `http://ipset.netfilter.org/`
- Regit's blog : `https://home.regit.org`

Contact us

- Eric Leblond: eric@regit.org, @Regiteric on twitter
- OISF-users and OISF-devel mailing lists