Suricata

Éric Leblond / Victor Julien

OISF

March 12, 2013

IDS? IPS?

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

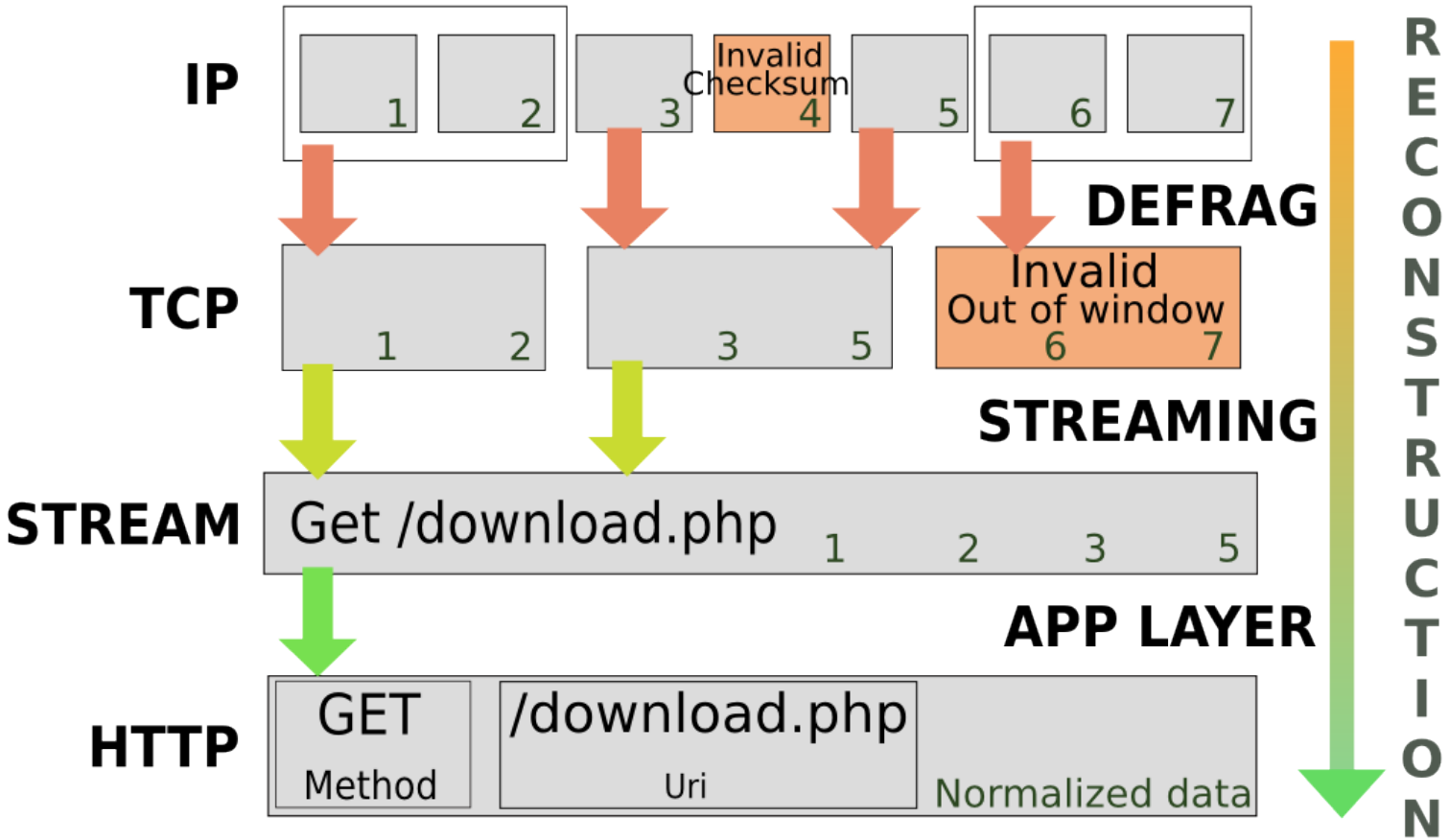
IDS

- (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

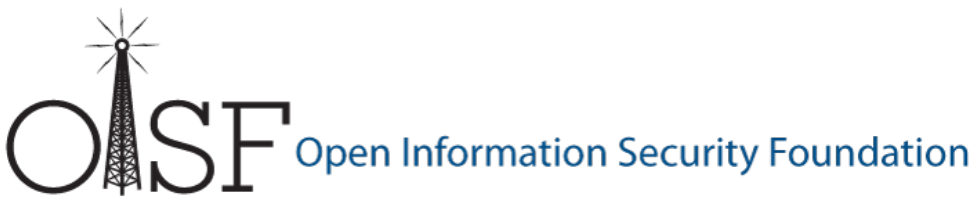
Suricata with dedicated ruleset

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

About OISF

Open Information Security Foundation

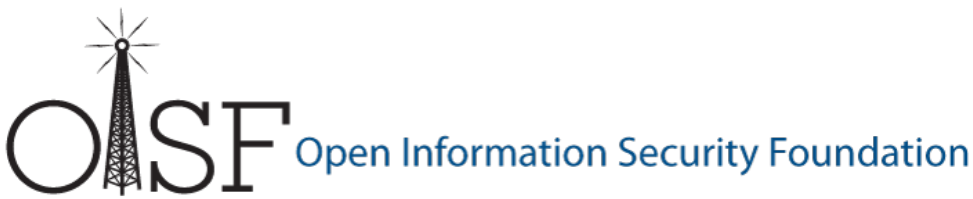
- `http://www.openinfosecfoundation.org`
- Non-profit foundation organized to build a next generation IDS/IPS engine
- Funded by US Governement (DHS, Navy)
- Development of an Open Source IDS/IPS:



About OISF

Open Information Security Foundation

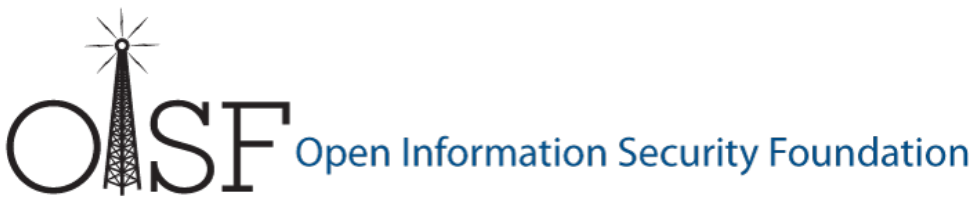
- `http://www.openinfosecfoundation.org`
- Non-profit foundation organized to build a next generation IDS/IPS engine
- Funded by US Governement (DHS, Navy)
- Development of an Open Source IDS/IPS:
  - Paying Developers



About OISF

Open Information Security Foundation

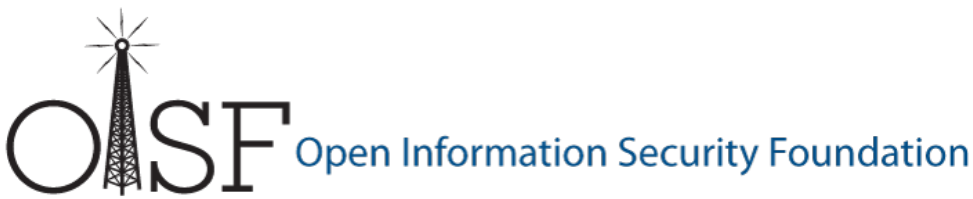
- `http://www.openinfosecfoundation.org`
- Non-profit foundation organized to build a next generation IDS/IPS engine
- Funded by US Governement (DHS, Navy)
- Development of an Open Source IDS/IPS:
  - Paying Developers
  - Financial support of related projects (barnyard2)



About OISF

Open Information Security Foundation

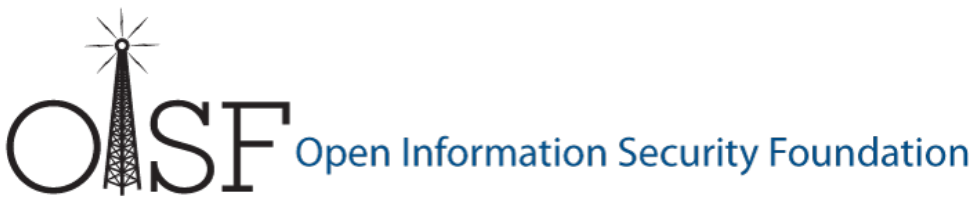
- `http://www.openinfosecfoundation.org`
- Non-profit foundation organized to build a next generation IDS/IPS engine
- Funded by US Governement (DHS, Navy)
- Development of an Open Source IDS/IPS:
  - Paying Developers
  - Financial support of related projects (barnyard2)
  - Board which oversees foundation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

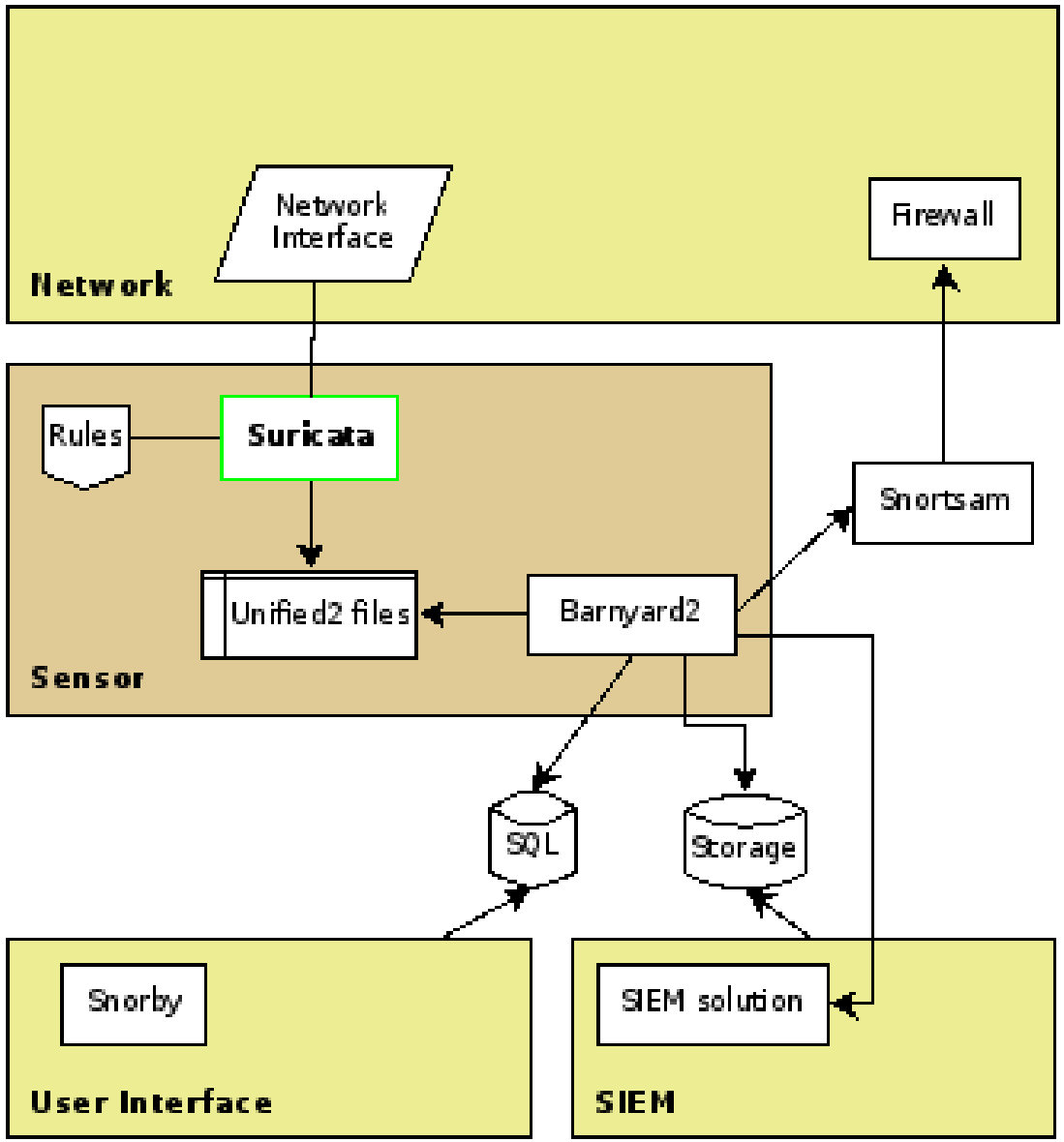
Features

- 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)

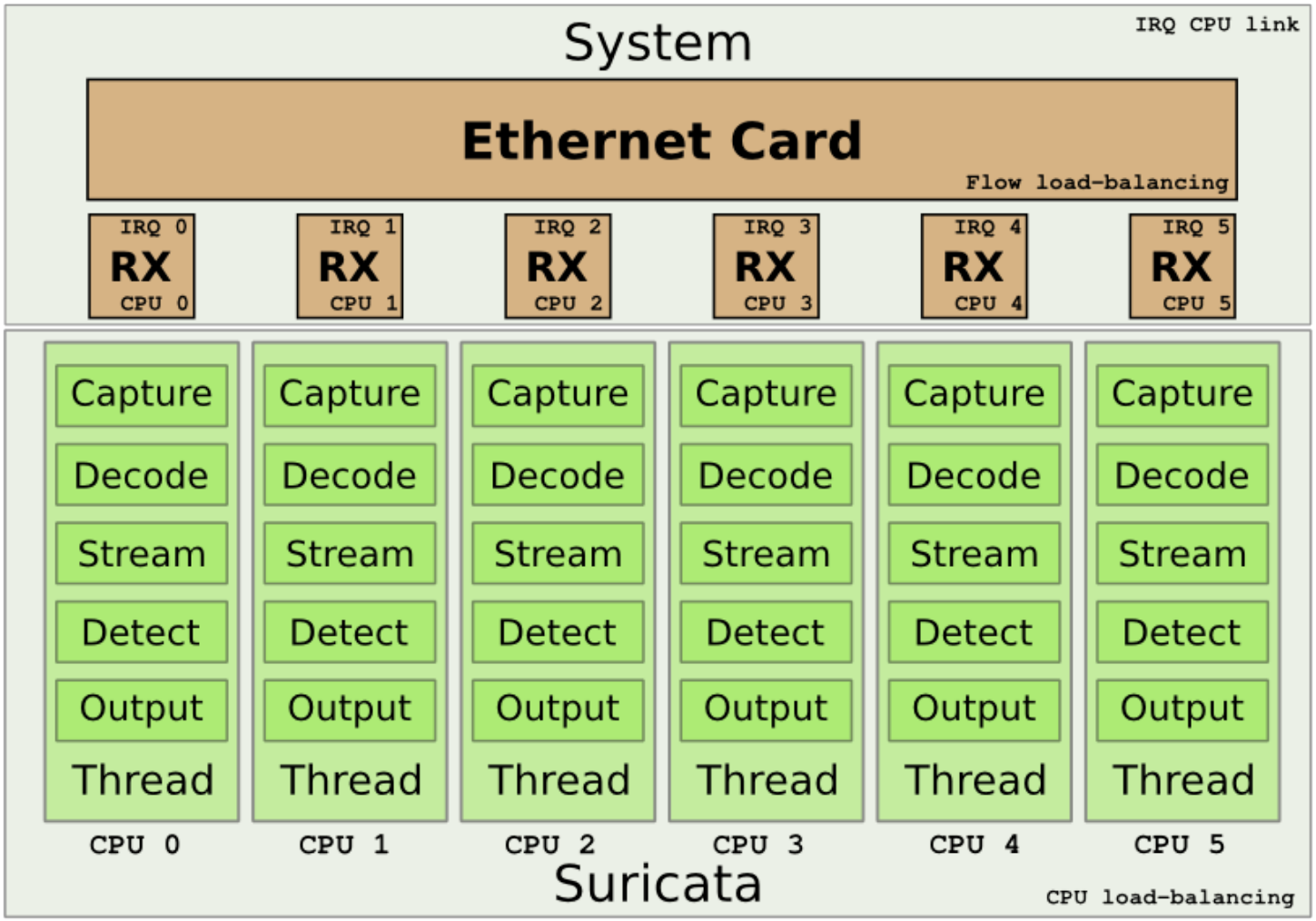
Features

- 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

Luajit rules

- Rule language is really simple
- Some tests are really difficult to write
  - Logic can be obtained via flowbit usage
  - But numerous rules are necessary
- A true language can permit to
  - Simplify some things
  - Realize new things

Experimental rules: `https://github.com/EmergingThreats/et-luajit-scripts`

Declaring a rule

```
alert tcp any any -> any any (msg:"Lua rule"; luajit:test.lua; sid:1;)
```

An example script

```
function init (args)
    local needs = {}
    needs["http.request_line"] = tostring(true)
    return needs
end
-- match if packet and payload both contain HTTP
function match(args)
    a = tostring(args["http.request_line"])
    if #a > 0 then
        if a:find("^POST%s+/.*%.php%s+HTTP/1.0$") then
            return 1
        end
    end
    return 0
end
```

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/`

Stream inline

- High level applicative analysis works on a data stream
- TCP data can be messy
  - Packets loss
  - Packets retransmit
  - Out of order packets
- The $IDPS$ must reconstruct the TCP flow before doing the applicative analysis

Problem

- IDS must be the closer possible to what's received by the target
  - Packet analysis when reception has been proven
  - ACK reception trigger data analysis
- IPS must block the packets before they reached the target
  - The IDS algorithm will block packet *after* they go through
  - An other approach has to be used

IPS as a control point

- IPS is a blocking point
  - It is representative of what goes through
  - It can reconstruct the flows before send them

IPS as a control point

- IPS is a blocking point
  - It is representative of what goes through
  - It can reconstruct the flows before send them
- Suricata implementation
  - Reconstruction of data segments at reception
  - Send reconstructed data to applicative layer analyser
  - Take decision based on data
  - Rewrite packets if necessary
  - Transmit (possibly modified) packets

IPS as a control point

- IPS is a blocking point
  - It is representative of what goes through
  - It can reconstruct the flows before send them
- Suricata implementation
  - Reconstruction of data segments at reception
  - Send reconstructed data to applicative layer analyser
  - Take decision based on data
  - Rewrite packets if necessary
  - Transmit (possibly modified) packets
- Details: `http://www.inliniac.net/blog/2011/01/31/suricata-ips-improvements.html`

Suricata in IPS mode

Using a Linux/Netfilter based IPS

- Use NFQUEUE to send decision to userspace
- All packets of a connection must be seen to Suricata
- The brutal way: iptables -A FORWARD -j NFQUEUE

Suricata in IPS mode

Using a Linux/Netfilter based IPS

- Use NFQUEUE to send decision to userspace
- All packets of a connection must be seen to Suricata
- The brutal way: iptables -A FORWARD -j NFQUEUE

Interaction with the firewall

- NFQUEUE is a terminal target
  - An ACCEPT decision will shortcut the whole ruleset
  - This is the only possible decision but DROP
- The previous method is thus incompatible with the existence of a ruleset.

Living together: the IPS and the firewall case

Classic solution

Use mangle in the PREROUTING or FORWARD chains

- The rule is an isolated table
- Thus no interaction with the rest of the ruleset
- This mean we can do "nothing" in theses mangle chains

Details: `http://home.regit.org/2011/01/building-a-suricata-compliant-ruleset/`

Living together: the IPS and the firewall case

Classic solution

Use mangle in the PREROUTING or FORWARD chains

- The rule is an isolated table
- Thus no interaction with the rest of the ruleset
- This mean we can do "nothing" in theses mangle chains

Alternative solution

- Use advanced functionalities of NFQUEUE
- Simulate a non terminal decision (© Patrick McHardy)

Details: `http://home.regit.org/2011/01/building-a-suricata-compliant-ruleset/`

Alternate decision and packet marking

Alternate decisions

- NF_REPEAT : send the packet back to the start of the table
- NF_QUEUE : send the packet to another queue (chain software using NFQUEUE)

Details: `http://home.regit.org/2011/04/some-new-features-of-ips-mode-in-suricata-1-1beta2/`

Alternate decision and packet marking

Alternate decisions

- NF_REPEAT : send the packet back to the start of the table
- NF_QUEUE : send the packet to another queue (chain software using NFQUEUE)

nfq_set_mark

- New keyword that can be used in signature
- Put a Netfilter mark on the packet if the signature match
- Can be used in every network stack (QoS, routing, Netfilter)

Details: `http://home.regit.org/2011/04/some-new-features-of-ips-mode-in-suricata-1-1beta2/`

WTF: Word Terminator Flow

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

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

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
```

Propagating the mark

```
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
```

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

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;)
```

Watching the clever ones

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

Configuring ulogd

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

```
plugin="/home/eric/builds/ulogd/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

- Open Source Days team for accepting this conference
- All Netfilter developers for their cool work

More information

- Suricata website: `http://www.suricata-ids.org/`
- Victor's blog : `http://www.inliniac.net`
- Eric's blog : `https://home.regit.org`

Contact us

- Eric Leblond: eric@regit.org, @Regiteric on twitter
- Victor Julien: victor@inliniac.net, @inliniac on twitter