# 9. Network Intrusion Detection Systems

Simona Buchovecká, Tomáš Čejka

Faculty of Information Technology, CTU in Prague

*simona.buchovecka@fit.cvut.cz, tomas.cejka@fit.cvut.cz*

April 19, 2021

Section 1

# Introduction

# Abbreviations

**IDS** Intrusion Detection System

- focused on suspicious/malicious traffic detection
- host-based / network-based
- signature-based (recognition of bad patterns, malware)

**ADS** Anomaly Detection System

- statistical, machine learning, or other approach to detection
- anomalous traffic — deviation from normal, usually observed
- detection of known or unknown traffic
- "behavioral analysis"

**IPS** Intrusion Prevention System

- identify malicious activity, log information, report it, try to block/stop it
- packet discarding (/blackholing), connection resetting

Section 2

# Particular Detection Systems

# Existing Tools (mainly IDS)

**Packet-based**

- Snort
- Zeek (formerly Bro)
- Suricata
- ...

**Flow-based**

- Flowmon ADS / DDoS Defender
- Stream4Flow
- ntopng
- NfSen (batch processing)
- Analysis Pipeline (SiLK)
- NEMEA
- ...

# Existing Tools (mainly IDS)

**Offline Processing**

- Elasticsearch + Kibana
  elastalert (https://github.com/Yelp/elastalert)
  elastiflow (https://github.com/robcowart/elastiflow)
- Python + Pandas, matplotlib
- ...

# Snort Introduction

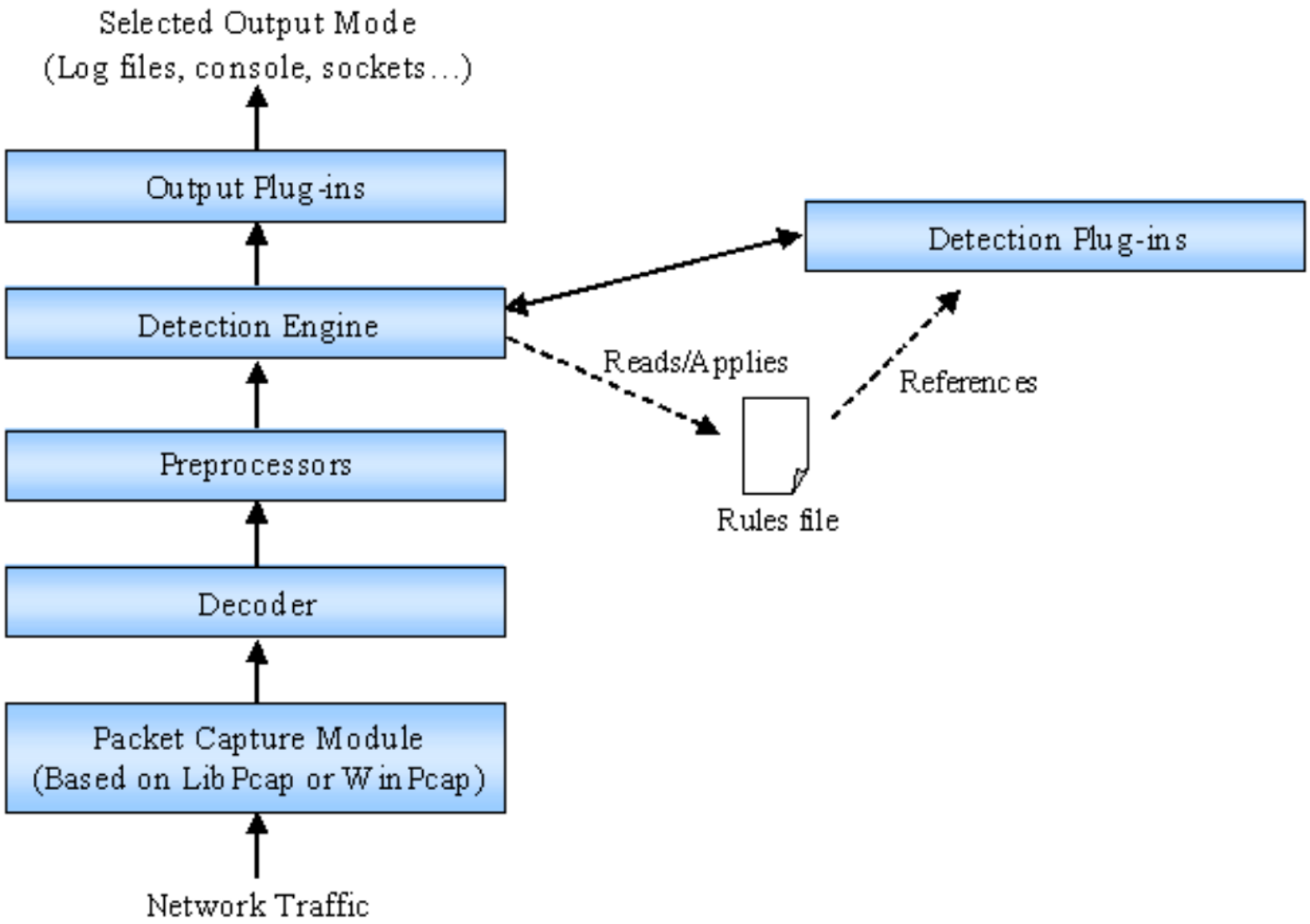
IDS/IPS - Intrusion Detection / Prevention System

- Statefulness: Rule-based detection with thresholds to track the number of times a rule was triggered

Open source, developed by Cisco Systems (formerly SourceFire)
Combines:

- Signature, protocol, and anomaly-based inspection
- But usually it's packet-based
- Sniffer, packet logger, intrusion detection

# Snort Structure & Plugins



# Snort Components #1

Packet Decoder

- Takes packets from various interfaces

Preprocessors

- Arrange or modify packets
- E.g., convert unicode or hex characters in URL to text
- Reassemble fragmented IP packets
- Reassemble TCP segments
- Check for anomalies in packet headers (and issue alerts)
- Port scan processor

# Snort Components #2

Detection Engine

- Detects intrusions in packets
- The detection engine speed is critical
- Uses Snort rules:
  - Rules arranged in a chain
  - If a rule is matched, a defined action is taken
  - Otherwise, the packet is dropped
  - *Actions*: Logging packets, issuing alerts, ...

General principles:

- The first rule that applies generates action/alert
- The highest priority applicable rule generates action/alert

# Snort Components #3

Rules are applied to:

- IP Header
- Transport Layer Header (TCP, UDP, ICMP)
- Application layer header
- Packet payload (legal issues)

Detection engine performance depends on:

- Machine power
- Internal bus speed
- Network load
- Number of rules

# Snort Components #4

Logging and Alerting System

- Logs in `/var/log/snort` by default

Output Plugins

- Log to `/var/log/snort/alerts` or other file, or via syslog facility
- Log to a database (MySQL, Oracle)
- Send e-mails, show web-based alerts
- Send SNMP traps
- Generate XML output
- Modify configuration of routers or firewalls
- Send SMB messages to MS Windows machines

# Snort Rules #1

Intro: https://resources.infosecinstitute.com/snort-rules-workshop-part-one/

General form:

```
action proto src_ip src_port direction dst_ip
dst_port (options)
```

Log 100 packet if ssh exploit is suspected

```
activate tcp any any -> 192.168.1.21 22
(content:"/bin/sh"; activates:1; \
msg:"Possible SSH buffer overflow"; )

dynamic tcp any any -> 192.168.1.21 22
(activated_by:1; count:100;)
```

# Snort Rules #2

Custom action:

```
ruletype redalert
{
    type alert
    output alert_syslog: LOG_AUTH LOG_ALERT
    output database: log, mysql, user=snort
dbname=snort host=localhost
}
```

# Snort Rules #3

Detect Nop:

```
alert tcp any any -> any any (msg:"Possible exploit"; \
    content:"|90|"; offset:40; depth:75; dsize: >6000;)
```

SYN & FIN Sent in one packet:

```
alert any any -> any any (flags: SF,12; \
    msg: "Possible SYN FIN scan";)
```

Note: Do not use escaped quotes

# Zeek (Bro)

## Basic information

- Transforms packets into events
- Events are processed using a script interpreter
- Turing complete Bro scripting language
- Zeek analyzers, in Zeek's event engine, perform application layer decoding, anomaly detection, signature matching, connection analysis

Resources:
`https://www.zeek.org/`

`https://en.wikipedia.org/wiki/Zeek`

# Suricata (1/3)

**Basic information**

- IDS, IPS, Network Security Monitoring (NSM)
- TCP/IP engine (IPv6 support, tunnel decoding, tracking sessions, reassembling)
- protocol parsing (HTTP, SSL, TLS, SMB, ...)
- PCRE support
- Lua scripting

**Rules format**

*action header options*
Example:

```
drop tcp $HOME_NET any -> $EXTERNAL_NET any
(msg:"ET TROJAN Likely Bot Nick in IRC (USA +..)";
flow:established,to_server; flowbits:isset,is_proto_irc;
content:"NICK "; pcre:"/NICK .*USA.*[0-9]{3,}/i";
reference:url,doc.emergingthreats.net/2008124;
classtype:trojan-activity; sid:2008124; rev:2;)
```

# Suricata (3/3)

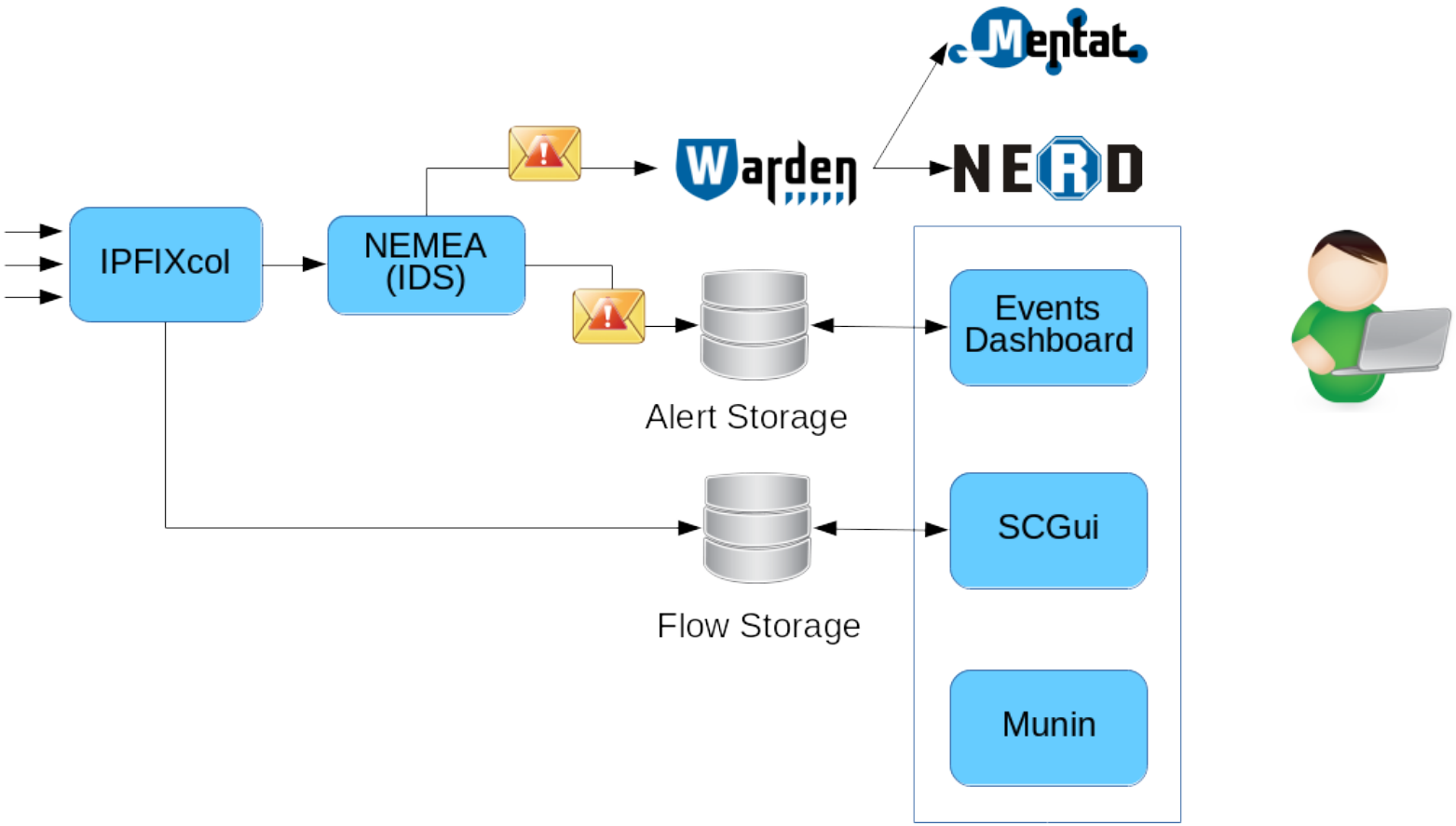

Resources:

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

# Network Measurements Analysis (NEMEA) (1/2)

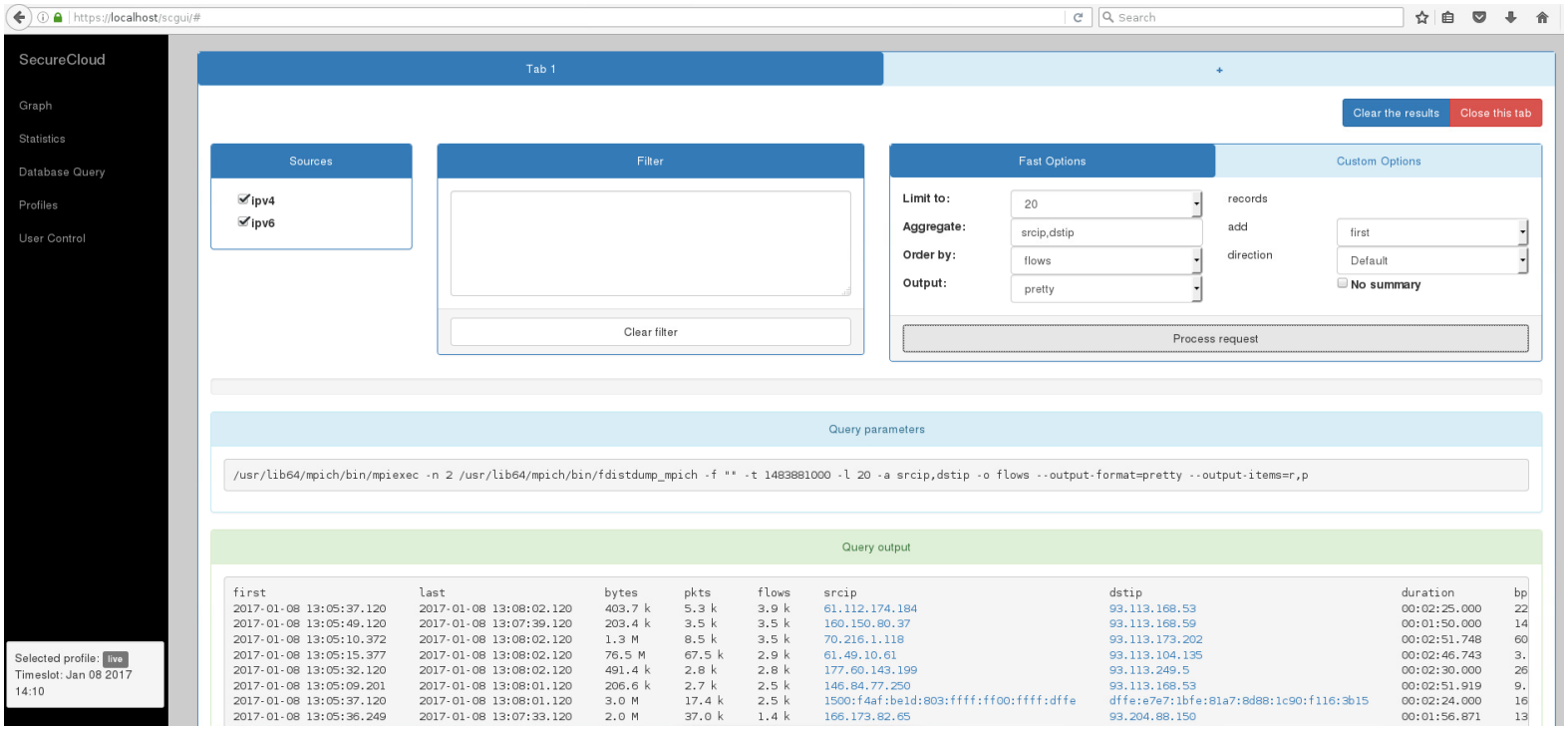
- Modular, consisting of independent interconnected NEMEA modules
- Flow-based
- Stream-wise
- Application-aware (can work with L7-extend flow records)

# Network Measurements Analysis (NEMEA) (2/2)

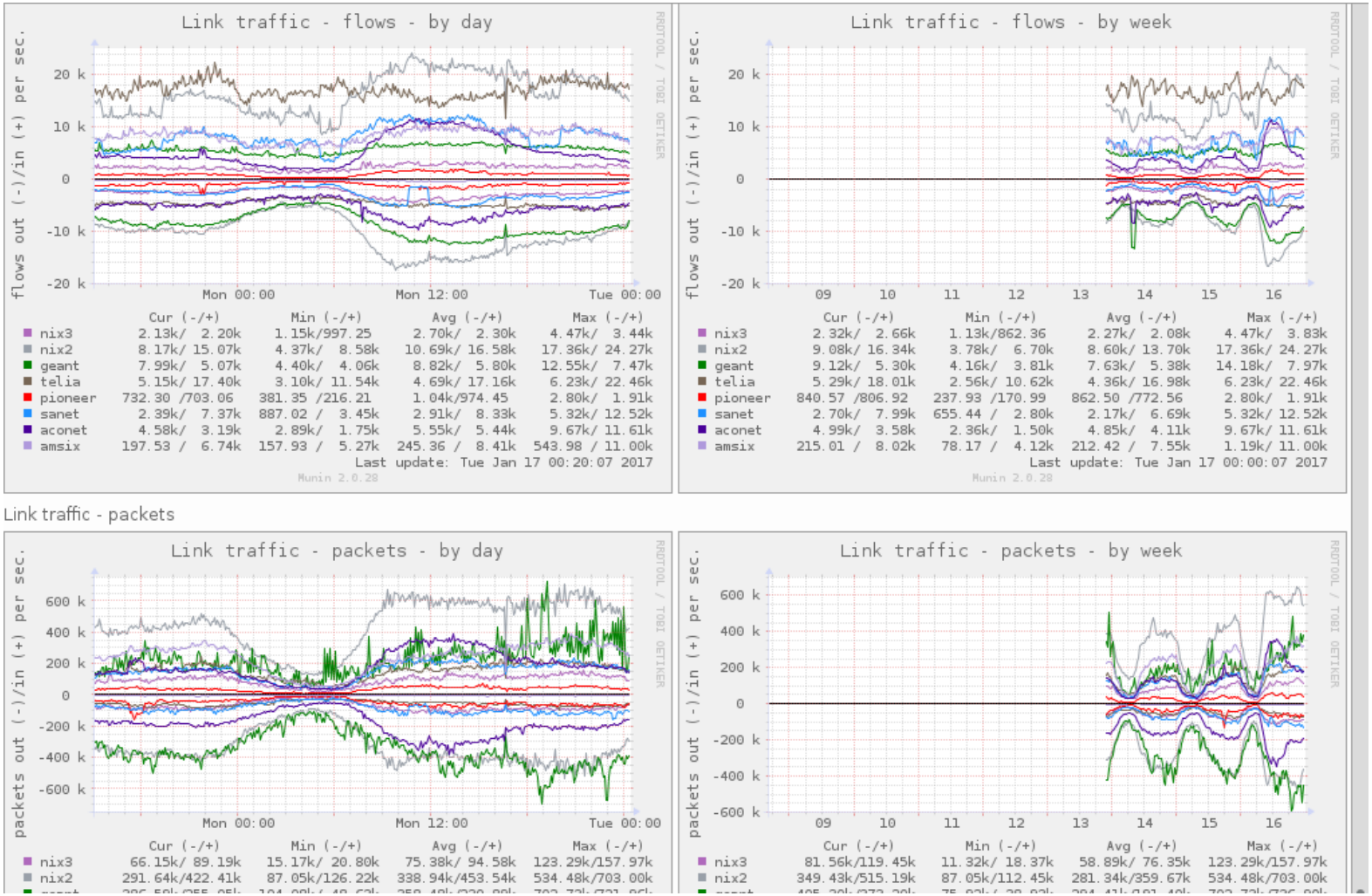


# NEMEA and other tools: screenshots of visualization (1/4)

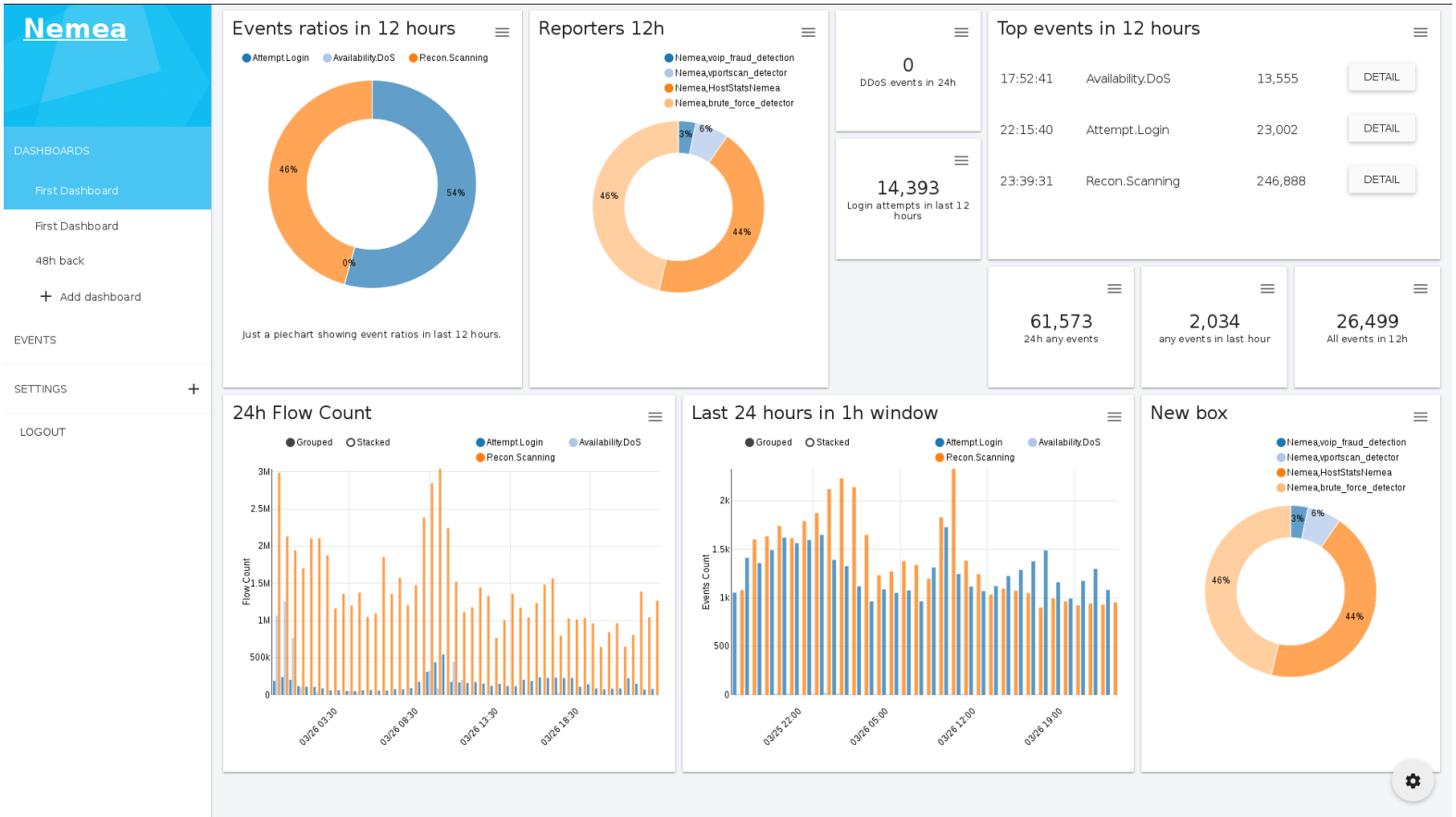
# NEMEA and other tools: screenshots of visualization (2/4)



# NEMEA and other tools: screenshots of visualization (3/4)



# NEMEA and other tools: screenshots of visualization (4/4)




- http://nemea.liberouter.org/
- http://github.com/CESNET/LiST
- http://github.com/CESNET/SecurityCloudGUI

# FastNetMon

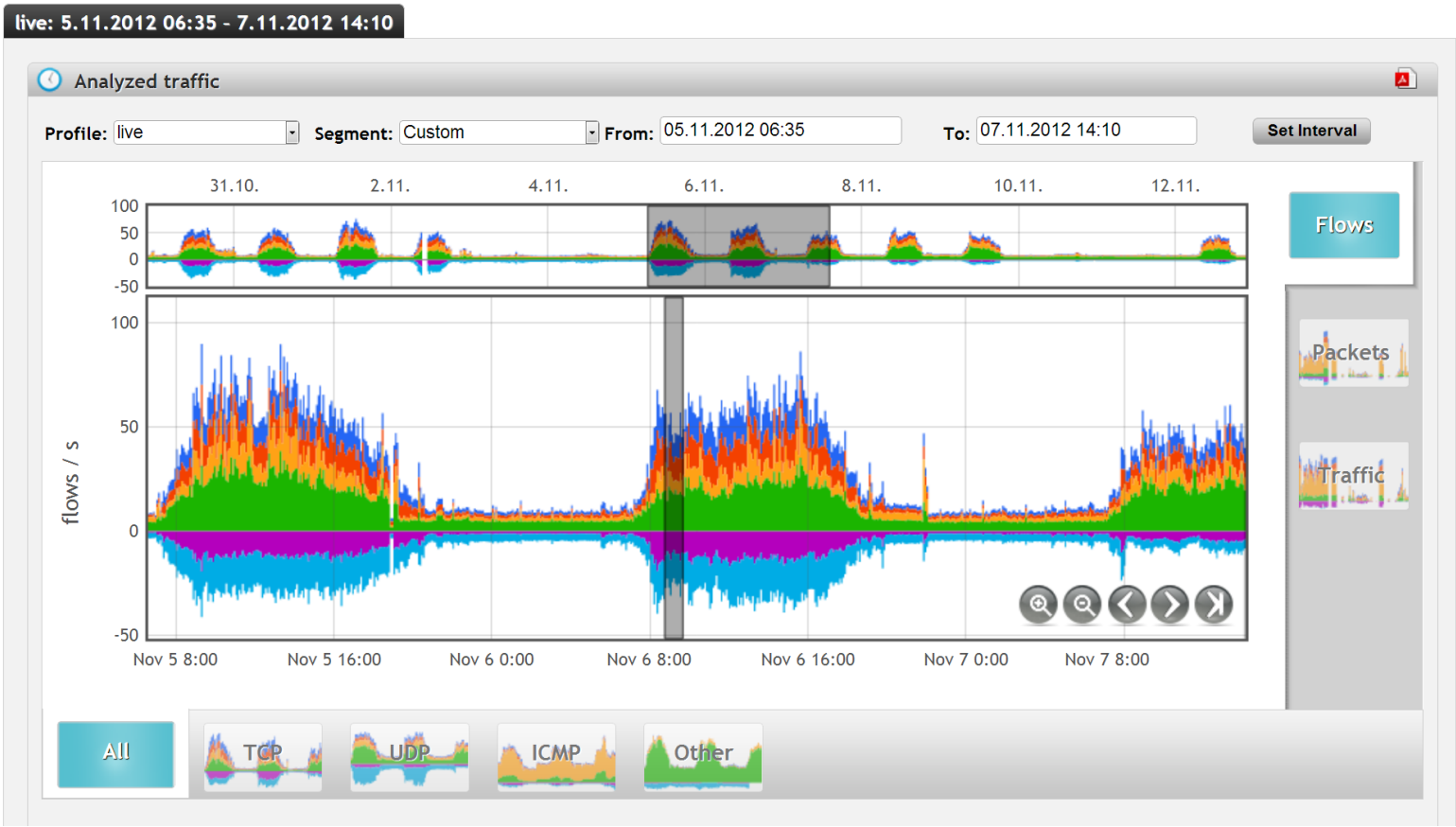
- Detects DDoS Attacks in 2 seconds
- Supports flow data, sFlow, port mirror/SPAN
- Supports BGP

```
FastNetMon v1.0
IPs ordered by: packets (use keys 'b'/'p'/'f' for change) and use 'q' for quit
Threshold is: 35000 pps and 1000 mbps total hosts: 13568

Incoming traffic        171015 pps    384 mbps  11973 flows
159.11.22.33              3309 pps   33.3 mbps     77 flows
159.11.22.33              3116 pps   34.8 mbps      2 flows
159.11.22.33              2567 pps   29.5 mbps      2 flows
159.11.22.33              2439 pps    1.8 mbps     76 flows
159.11.22.33              2364 pps    1.4 mbps     55 flows
159.11.22.33              2104 pps    1.5 mbps     19 flows
159.11.22.33              1938 pps    1.3 mbps     36 flows

Outgoing traffic        225121 pps   1905 mbps  17893 flows
159.11.22.33              3699 pps   39.9 mbps     83 flows
159.11.22.33              3557 pps   37.3 mbps    124 flows
159.11.22.33              2965 pps   32.8 mbps     98 flows
159.11.22.33              2645 pps   29.7 mbps     38 flows
159.11.22.33              2522 pps   26.1 mbps     65 flows
159.11.22.33              2474 pps   26.8 mbps     61 flows
159.11.22.33              2285 pps   18.9 mbps    194 flows

Internal traffic             0 pps      0 mbps

Other traffic               56 pps      0 mbps

Traffic calculated in:  0 sec 14670 microseconds
Packets received:       2308537
Packets dropped:        0
Packets dropped:        0.0 %
```

https://fastnetmon.com

# Flowmon

## Commercial




https://www.flowmon.com/en/

Section 3

# Closing Words

# There are many more...

Discussion?

What are Your experiences?

# Questions?