

Automated Incident Handling with IntelMQ TF-CSIRT/FIRST Malaga

Aaron Kaplan <kaplan@cert.at>
Sebastian Wagner <wagner@cert.at>

The IntelMQ project



Started in 2014 by Tomas Lima (CERT.pt at that time) and Aaron Kaplan (CERT.at)

- We receive “Threat intelligence” feeds for AT
 - Infected systems, vulnerable servers, hacked websites, etc
- We want to contact the infrastructure owners

For CERTs, SOCs, other security teams

Included in MeliCERTes and csirt-kit

github.com/certtools/intelmq

CERT.at's role



Contributing development

Maintainer role

- Packaging
- Release management
- Coordination



**Co-financed by the Connecting Europe
Facility of the European Union**

Where to get help?



Ask us in case of problems or questions
IntelMQ is a community

- User Guide:
<https://github.com/certtools/intelmq/blob/develop/docs/User-Guide.md#where-to-get-help>
- Users mailinglist:
<https://lists.cert.at/cgi-bin/mailman/listinfo/intelmq-users>
- Developers mailinglist:
<https://lists.cert.at/cgi-bin/mailman/listinfo/intelmq-dev>
- For bugs: <https://github.com/certtools/intelmq/issues>

Core idea



- Getting (high volume) data (infections, vulnerabilities, ...)
- Sharing the data with the affected parties
 - Requires joining some other context data
 - Exact workflows differ from CERT to CERT
 - Recipient should not need any tool

Requirements I

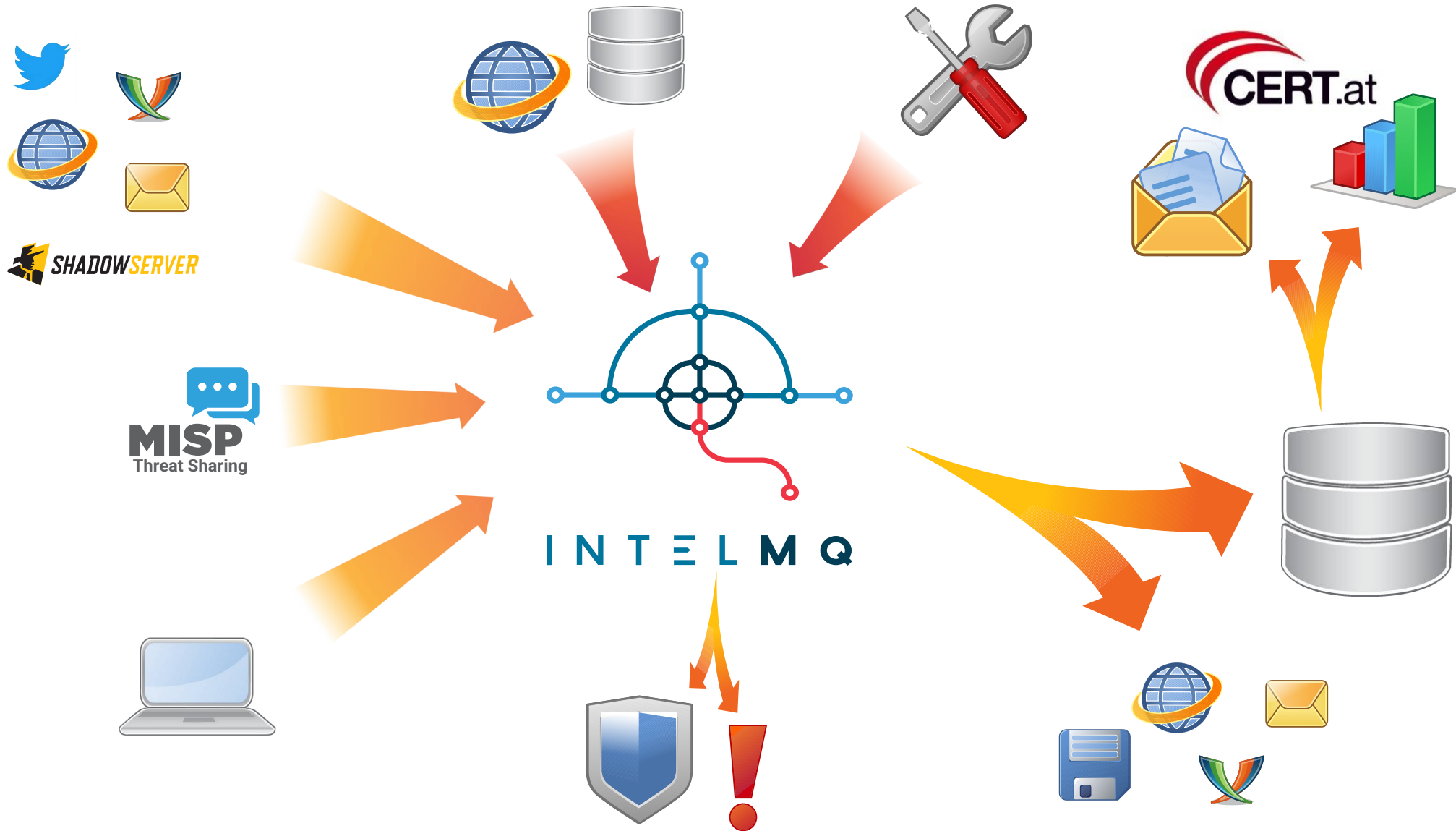


- Data Format
 - Sanitation: Syntactical
- Harmonizes data:
 - Add Malware Family Name (optional)
 - Classification
 - Reference Incident Classification Taxonomy
- Scalability

Requirements II



- Easy to use
- Easy to extend
- Flexible in the data format
 - Extensible
 - But syntactical correctness
- Sharing code / community work
- Handle changes in feeds



Demo / Familiarization



- Check SSH connection
- Check HTTP connection
 - IntelMQ-manager configuration tab
 - <http://localhost:8080/?page=configs>

Base Concept



- Structured data in
- Processing
- Data out
- → **no state** (with exceptions)

Concepts I

- “Bots”
 - Single & separated programs (processes)
 - One task per bot
 - “botnet” is the entirety of bots
- Pipeline
 - Connection between bots
 - “message broker” (messaging queue)
 - Message types: “Report” & “Event” in JSON format

4 types of “bots”

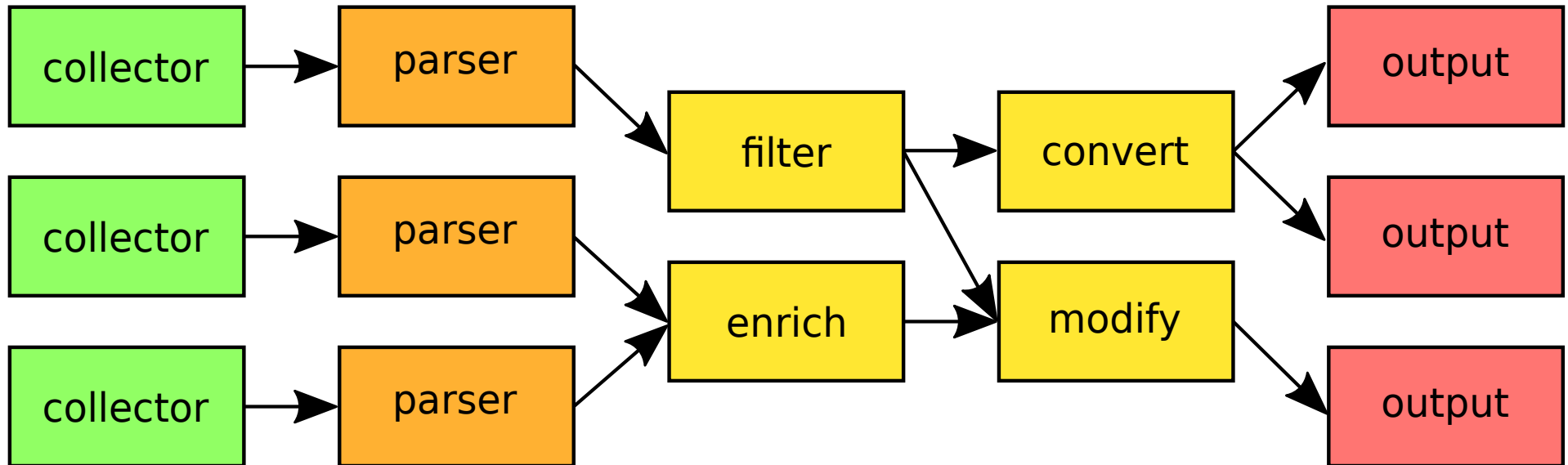
- Collectors
- Parsers
- Experts*n
- Output

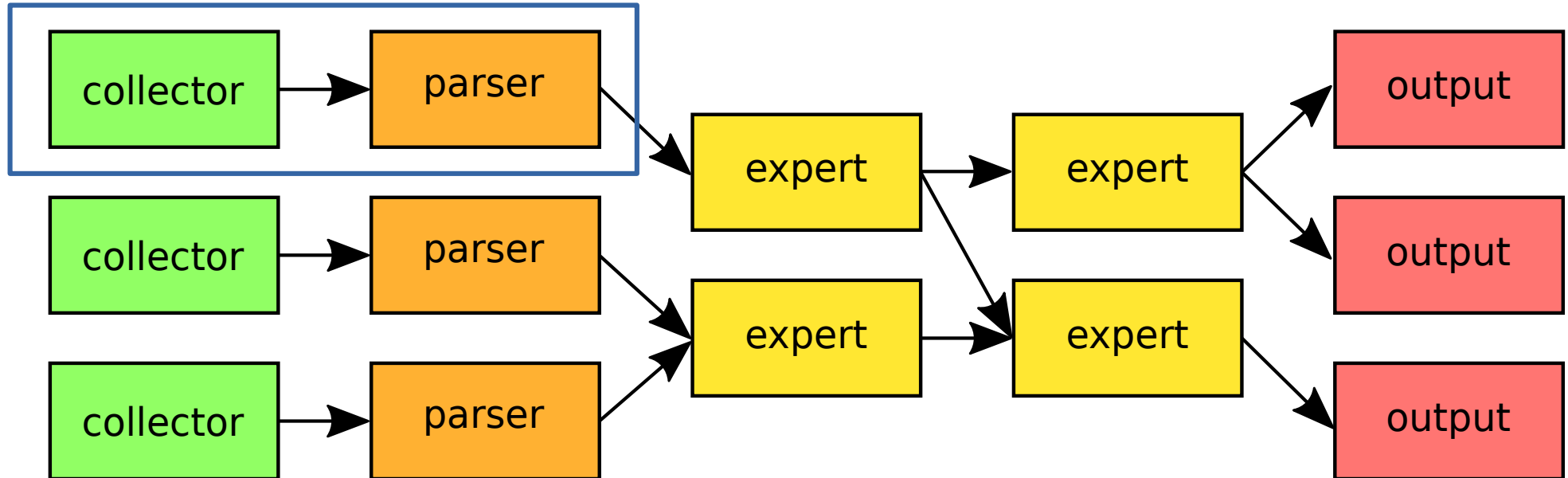


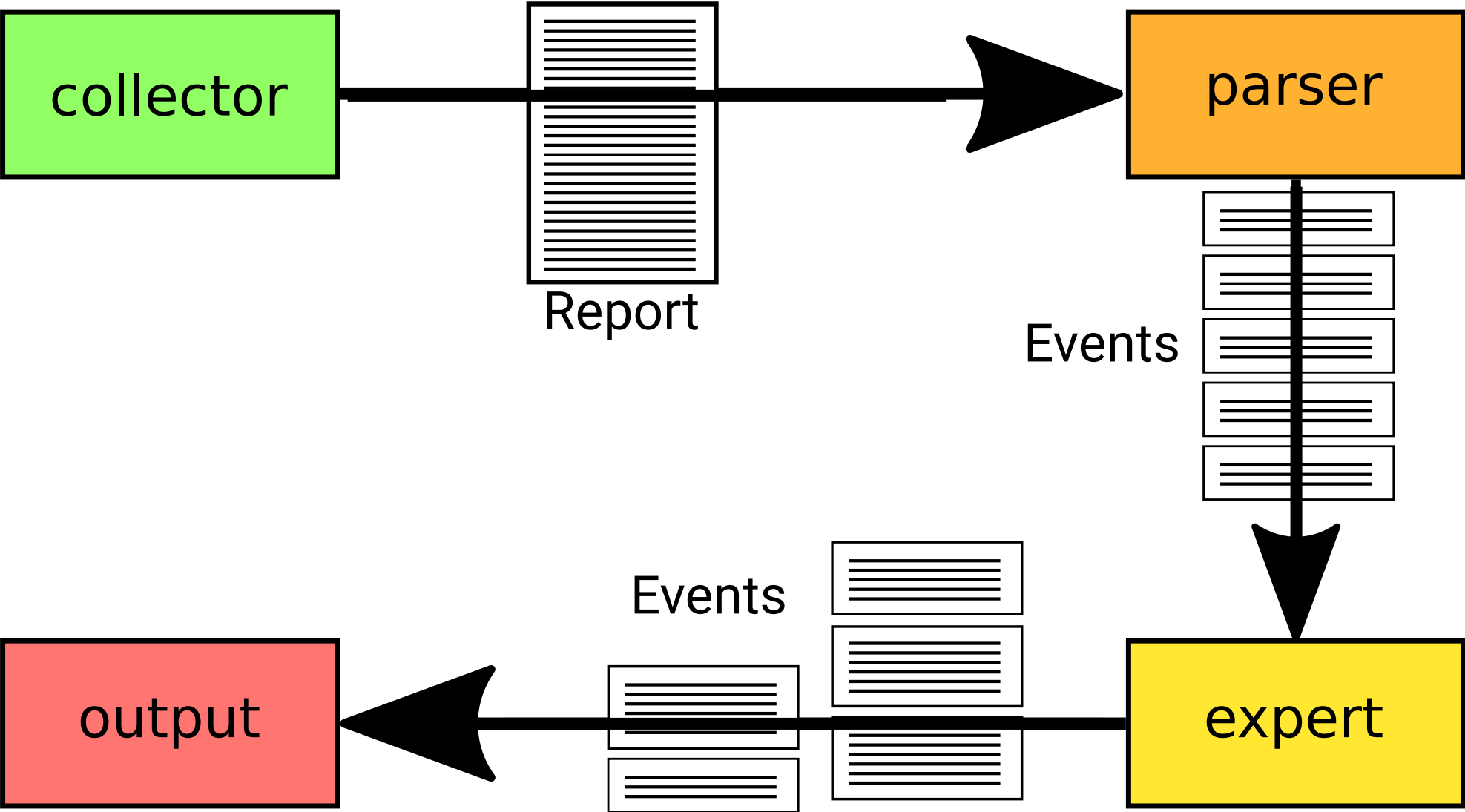
2 types of messages



- Reports
 - Raw data + some metadata
- Events
 - Parsed data + some metadata







Concepts II

- EventDB (optional)
 - (PostgreSQL) event database
 - Same schema as IntelMQ's format
- “Data Harmonization Ontology”
 - Data format specification
 - (structured) input data is converted to internal data format
 - Syntactical checks on data
 - Normalization

Demo / Hands-On

Normalization examples



- Time: “Thu 27 Jun 2019 14:02:03+02” → “2019-06-27T14:02:03+02:00”
- Classification: conficker infection → “Malicious code” > “infected device” > “conficker”
- Malware names: b67-ss-kins → zeus, avalanche-panda-banker g → zeus
- Network number: AS65536 → 65536
- FQDN: example.com. → example.com
- FQDN: österreich.gv.at → xn--sterreich-z7a.gv.at
- IP address: 127.0.0.1/32 → 127.0.0.1
- Protocol: HTTP → http
- TLP: tlp:amber → AMBER
- Country Code: “si” → “SI”

Sanitation

- *Syntactical* Validation
 - Not a *verification on correctness* of information (can be added though)
- Check for syntactical correctness
- Check for completeness of information
- Further processing steps can rely on these checks

Concepts III

- No data should get lost
 - Parsing errors
 - As strict as possible but not more
 - Handling of changes in data feeds
 - Any other problems
 - Raise admin attention

Data collection



- Active / passive
- Direct (API) / Indirect (Ticketing system, IMAP)
- Pull / Push
- Wide range of formats
- Stateless / Stateful (keeping track of fetched data)
- Stream / bulk
- Global / specific scope (filtering)
- Public / private

Expert functions

- Filtering & Dropping
- Routing
- Modifying & Extending (“Lookups”)

Enrichment challenges



- Online Lookups latency (Name resolution)
 - Example:
 - 99% 5ms, 1% 10s: 543/minute
 - 100% 10ms: 600/minute
 - → Download databases
- Accuracy of data

SetUp Workflow



- get data from
 - HTTP APIs, E-Mails (IMAP, RTIR, ...)
 - internal sources, public sources
- define data processing flow
 - filtering
 - routing
 - enriching
- define how to act
 - output to other systems
 - direct actions

Hands-On

Available Bots and Feeds



- 22 collectors
- 60 parsers
- 26 experts
- 16 outputs

<https://github.com/certtools/intelmq/blob/develop/docs/Bots.md>

- 142 *documented* feeds

(as of version 2.1.1)

<https://github.com/certtools/intelmq/blob/develop/docs/Feeds.md>

Classification

- 3 levels
- *classification.taxonomy* (ENISA WG)
- *classification.type* (ENISA WG)
- *Classification.identifier* (freetext)
- <https://github.com/certtools/intelmq/blob/develop/docs/Data-Harmonization.md#classification>

IntelMQ Manager



- Graphical (pipeline) Configuration
 - Including live view (queued messages)
 - Reads/Writes configuration files
 - Visualization interface
- Botnet management
 - Interface for *intelmqctl*
- (User-based) Monitoring
- Simple PHP Backend

Hands-On

The last mile

- Everybody has a different workflow
 - SMTP vs different Ticketing systems
 - Different grouping
 - Providing API or rendering data
 - ...
- No standard-procedure

IntelMQ vs. MISP



- MISP:
 - Synchronization and collaboration as core-feature
 - manually curated IoCs
 - Database represents a state
 - Linking & correlation
- IntelMQ:
 - Stream-processing for mass-automation
 - little/no human interaction
 - (more or less) stateless

Run modes



- Continuous vs scheduled
- Enabled true / false (“autostart”)
- rate_limit (seconds)

Hands-On

Some related software

<https://github.com/certtools/intelmq/blob/develop/docs/Ecosystem.md>

Intelmq-mailgen / -fody



- Sending with SMTP
- Compatible with OTRS
- Rulesets and contacts from intelmq-fody
- Webinterface for tickets, contacts, rules, event database search & stats
- Developed by BSI/Intevation

<https://github.com/Intevation/intelmq-mailgen/>
<https://github.com/Intevation/intelmq-fody/>



TOOLS

 Dashboard

 Tickets

 Contacts

 Statistics

DASHBOARD

Overview of environment

Home > DASHBOARD



TICKETS TODAY

91



EVENTS TODAY

3380



RECENTLY SENT

**20170809-
10000091**

← → localhost:8070/contacts?cidr=77.87.224.0%2F21

CONTACTS Maintain contact database

Home > CONTACTS

Lookup ASN

49234

Search CIDR

77.87.224.0/21

Found 1 auto-imported and 1 manual organisations.

Search Email

abuse@bund.de

Search Name

Bundesamt fuer Sicherheit in der Informationstechnik

Bundesamt fuer Sicherheit in der Informationstechnik **manual**

<abuse@bund.de>

ASN49234 **daily**

Networks:

77.87.224.0/21

Inhibition:

event_field ['classification.type'] == 'botnet drone'

comment:

first_handle:

ripe_org_hdl: ORG-BA202-RIPE

sector_id:

ti_handle:

Bundesamt fuer Sicherheit in der Informationstechnik **auto**

<abuse@bund.de>

ASN49234

Networks:

77.87.224.0/21

comment:

first_handle:

import_source: ripe

import_time: 2017-03-29T15:40:34.357995

ripe_org_hdl: ORG-BA202-RIPE

sector_id:

ti_handle:

Clone



<https://github.com/Intevation/intelmq-mailgen/>
<https://github.com/Intevation/intelmq-fody/>

Hands-On

IntelMQ-fody events

intelmq-webinput-csv



- Webform to submit
- Shows data validity
- Python WSGI
- <https://github.com/certat/intelmq-webinput-csv/>

intelmq-webinput-csv



timezone

+00:00

dry run



Constant fields (fallback):

classification type

malware configuration

Taxonomy: malicious code

feed.code

oneshot

classification.identifier

test

time.source	source.ip	destination.ip	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
timestamp	ip	protocol	port
2018-12-09 02:53:18	85.126.145.244	tcp	548
2018-12-09 02:53:18	77.119.237.121	tcp	548
2018-12-09 02:53:18	77.119.229.227	tcp	548
2018-12-09 02:53:21	178.18.164.3	tcp	548
2018-12-09 02:53:21	78.132.127.172	tcp	548
2018-12-09 02:53:49	62.40.138.97	tcp	548
2018-12-09 02:53:49	212.197.156.245	tcp	548

<https://github.com/certat/intelmq-webinput-csv/>

Hands-On

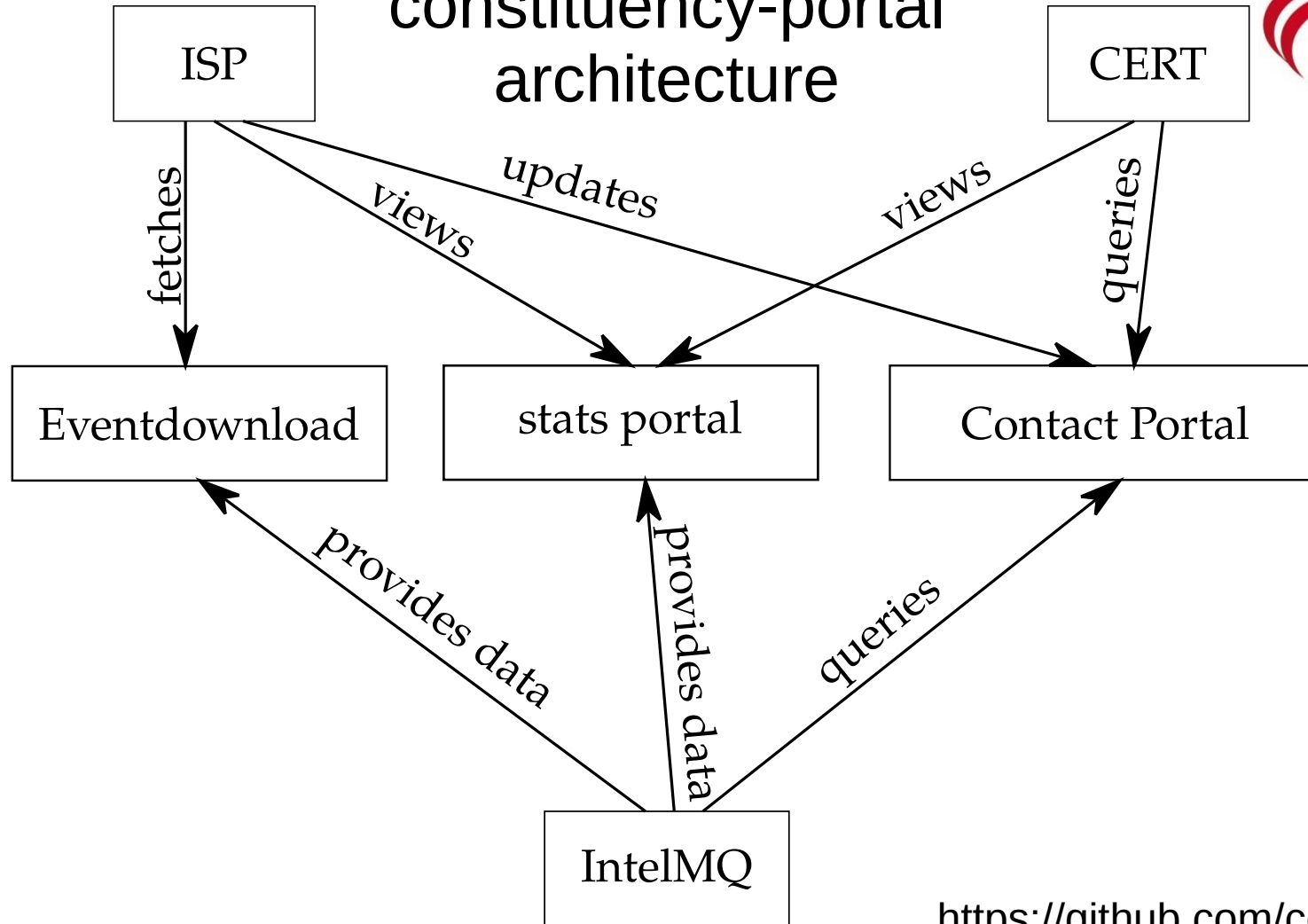
webinput-csv

constituency-portal



- In development at cert.at
- Provides self-administration for constituents
- Integrates RIPE-data
- Shows statistics to constituents (stats-portal)

constituency-portal architecture





top tags over time filtered by taxonomy ▾



Last 90 days



taxonomies All ▾

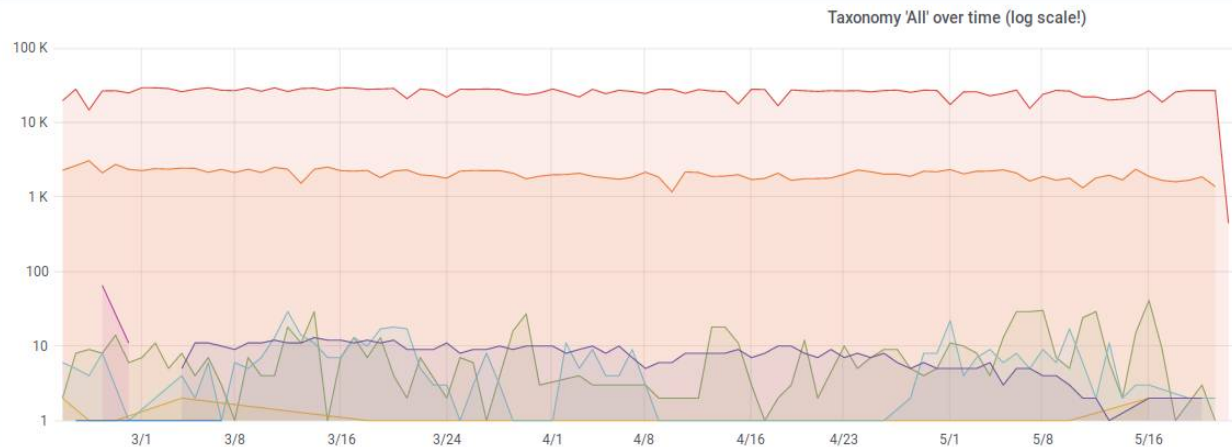
ASN list 30971 + 49488 + 201612 + 35492 + 1901 + 8447 + 8562 + 12793 + 15824 + 5403 + 13008 + 201951 + 1764 + 29330 + 50047 + 44865 + 25575 + 12605 + 16305 + 42587 + 24792 + 34347 + 12577 + 200026 + 21039 + 34885 + 28889 ▾

Panel Title

Classification.* over time

The following graph shows the 3 classification.* fields over time:

- "classification.taxonomy",
- "classification.type",
- "classification.identifier".



	min	max	avg	current▾	total
malicious code	1.17 K	3.08 K	2.05 K	1.39 K	180.70 K
vulnerable	446	29.32 K	25.36 K	446	2.25718 Mil
information content security	11	65	38	11	76
other	1	13	8	2	579
intrusion attempts	1	29	7	2	431
information gathering	1	2	1	2	17
intrusions	1	1	1	1	4
abusive content	1	41	9	1	701

Type over time (log scale!) / All



(Unsorted) list of some features

Named queues aka *paths*

- One source queue
- One or more destination queues
 - Implicit name: *_default*
- Error path
 - *_on_error*
- Bot-specific ones (filter, sieve)
 - Filter: *action_other*, *filter_match*, *filter_no_match*
 - Sieve: user-defined

Hands-On

filtering

Sieve expert



```
if :exists source.fqdn {
    keep // aborts processing of subsequent rules and forwards the event.
}
if :notexists source.abuse_contact || source.abuse_contact =~ '.*@example.com' {
    drop // aborts processing of subsequent rules and drops the event.
}
if source.ip << '192.0.0.0/24' {
    add! comment = 'bogon'
}
if classification.type == ['phishing', 'malware'] && source.fqdn =~ '.*\.(ch|li)$' {
    path 'domainabuse'
    keep
} else {
    remove comment
}
```

Hands-On

Sieve expert

Working interactively



- `intelmqctl run -l DEBUG bot_name`
- `intelmqctl run -l DEBUG bot_name process`
 - `--show-sent`
 - `--dryrun`
 - `--msg '{"source.ip": "127.0.0.1"}'`

Hands-On

Intelmqctl run

Provided installation modes



- Pip / pypi
 - Default path is */opt/intelmq/*
 - *pip3 install -e* for editable installations
- rpm/deb
 - For various distributions
 - */etc/intelmq/ /var/lib/intelmq /var/run/intelmq*
- Switch used paths with `INTELMQ_PATHS_OPT=1 / INTELMQ_PATHS_NO_OPT=1`

Error handling

- `error_log_message`: true/false
- `error_log_exception`: true/false
- `error_procedure`: stop/pass
- `error_max_retries`: int
- `error_retry_delay`: int (seconds)
- `error_dump_message`: true/false
- Special path `'_on_error'`

Error handling

- Dumped to *.dump files in log directory
 - *Not* kept in memory
 - Raise administrator attention
- JSON with exception and full message
- *intelmqdump* tool to inspect and re-insert

Malware Name Mapping

- Goal: different names in different *feeds* for the same or very similar malware (version)
- Non-goal: map names between av vendors
- Includes:
 - Malpedia
 - MISP Galaxy Threat Actors
- *malware.name* → *classification.identifier* (by default)
- https://github.com/certtools/malware_name_mapping

Malware Name Mapping

- ^b66(-(exe|https?(-unknown)?|ir|tv(ir)?))?\$
 - andromeda
- ^(cridex|bugat|feodo)\$
 - cridex
- ^dark-?mailer[23]?\$
 - darkmailer

Write a Bot



```
class MyBot(Bot):  
    def init(self):  
        # optional initialization  
    def process(self):  
        event = self.receive_message()  
        # process event  
        self.send_message()  
  
BOT = MyBot
```

Write a Bot



```
class MyBot(Bot):
    def init(self):
        with open('foobar.txt'):
            self.database = db.readlines()
    def process(self):
        event = self.receive_message()
        if 'source.ip' in event:
            if event['source.ip'] in self.database:
                event['source.tor_node'], True)
        self.send_message()

BOT = MyBot
```

Hands-On

Performance and Monitoring



- Generic Multithreading for AMQP
 - Additional to Multiprocessing
- AMQP as optional message broker
- Bot load's statistics
- Use local lookups, not over the Internet

Outlook (to IntelMQ 3.0)



- Expand to more use-cases
 - IDS/IPS, RPZ, SIEMs
 - Data-lakes, ML/AI input
 - Sensor networks
- Tighter IntelMQ – n6 (cert.pl) integration
- Data format changes
- Docker as optional process management
- Kafka as optional message queue
- Documentation: Tutorials and demo VMs
- Aggregation

Future discussion



Not a one-click solution



- You need to configure IntelMQ
- Everyone's workflow differs
 - Sending E-Mails vs providing APIs vs stats-only
 - Grouping differs
 - Intervals differ
- IntelMQ can be extended
 - By bots
 - and programs interfacing the database

Wrap-Up



Contribute!



Sharing is caring

- Documentation
 - usage examples
 - Feed support
- Adaptions and extensions
- ...

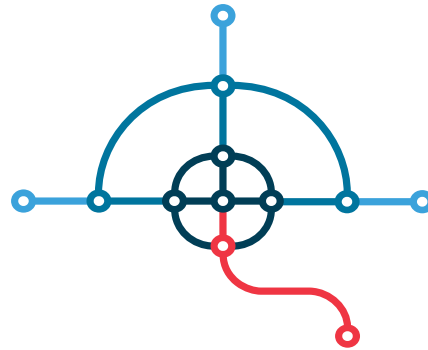
[Github.com/certtools/intelmq](https://github.com/certtools/intelmq)



Contributions

IHAP meetings

CERT.at Maintenance



I N T E L M Q

Mailing Lists

Github

[Github.com/certtools/intelmq](https://github.com/certtools/intelmq)

Thanks for listening!
Questions?

Aaron Kaplan <kaplan@cert.at>
Sebastian Wagner <wagner@cert.at>

github.com/certtools/intelmq



Co-financed by the Connecting Europe
Facility of the European Union