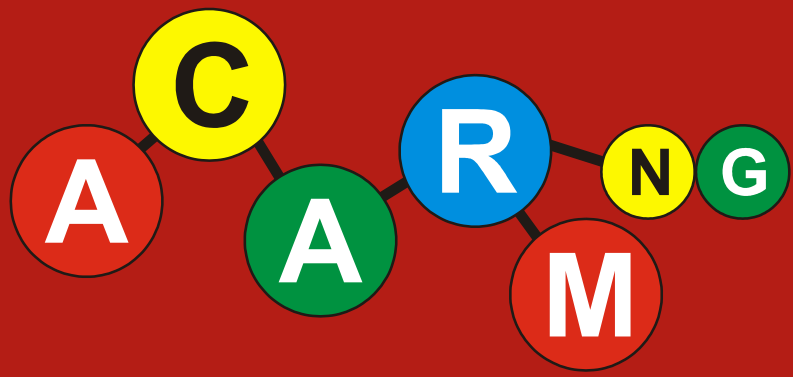
Wrocław University of Technology

ACARM-ng: An Intrusion Protection System for HPC and Grid Environments



Bartlomiej Balcerek, Bartosz Szurgot,
Mariusz Uchronski and Wojciech Waga
Wrocław Centre for Networking and Supercomputing

June 2011

Outline

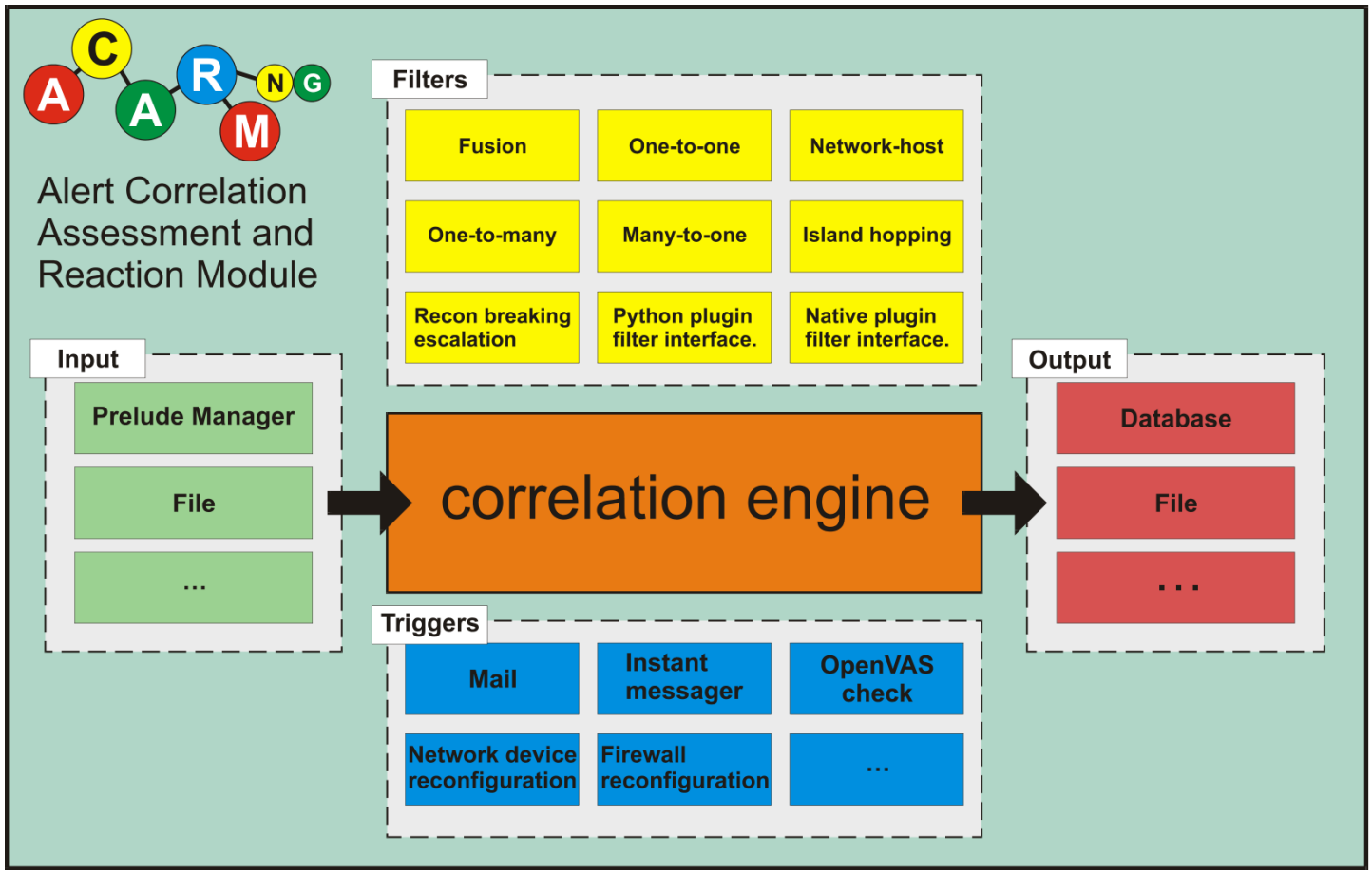
Introduction

- Although computer networks are ubiquitous for more than a decade, there is still no ultimate solution to ensure proper security.
- There are two major approaches to the intrusion detection: anomaly and misuse detection.
- For a large network you can get a number of reports that you are not able to process in real time.
- The best solution to this problem is to use a correlation engine which is able to merge similar events together as well as discard irrelevant data.

Acarm-ng overview

- ACARM-ng is an extensible, plug-in-based alert correlation framework.
- Developed to improve security in Polish NGI (National Grid Initiative) – PL-Grid.
- It consists of abstractions for correlation, reporting, reaction, gathering data from multiple sources and data storage.
- Real-time reporting is supported – alerts can be reported while still being correlated.
- WWW is provided for the administrator to present gathered and correlated data in a consistent way.

Architecture (top-level design)



Architecture

Main system's part is a FIFO queue of events (meta-alerts in ACARM-ng's terminology). All other system's parts have been logically split into the following categories:

- Filters – data processing abstraction.
- Triggers – notification and reaction abstraction.
- Inputs – data collecting abstraction.
- Persistencies – data storage abstraction.

Each category has a generic interface so that user-provided implementations can be loaded as plug-ins.

Data flow

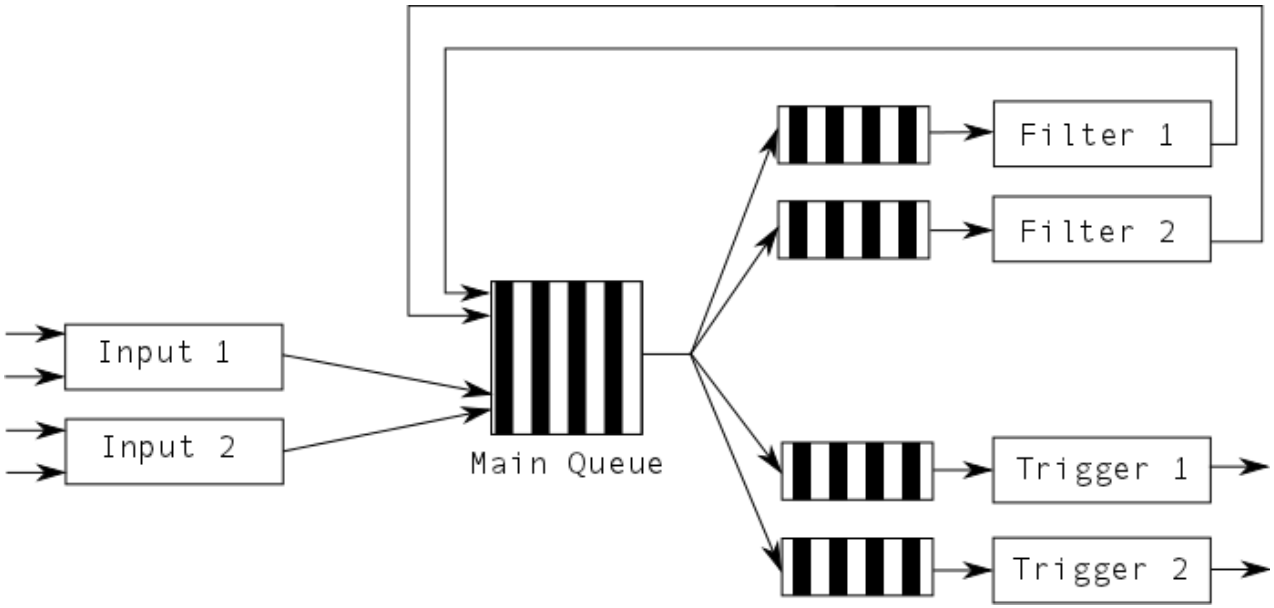


Figure: Data (i.e. meta-alerts) processing flow overview in ACARM-ng.

Methods

1. A newly captured alert enters the system via the input plug-in.
2. It is checked by preprocessor which acts like an anti–SPAM filter and discards group of alerts we are not interested in.
3. Alert enters subsequent phase of processing – it is placed in the memory of correlation engine and scheduled for filtering.
4. All filters act in parallel trying to find similar alerts and merge them together.

Methods

5. When the severity of a correlated group of alerts, also referred to as a meta-alert, exceeds a given threshold, triggers are launched.
6. When the lifetime of an alert, specified in configuration file is over, alert exits system.
7. Database cleanup procedure, run periodically, gets rid of the oldest alerts making place for new ones.

Correlation and reaction modules

Filters (correlation):

- One-to-one
- One-to-many
- Many-to-one
- Many-to-many
- Same name
- DNS resolver
- IP-Blacklist
- Event chain
- User monitor
- Similarity
- New event

Triggers (reaction):

- Save to file (IDMEF)
- E-mail notification
- Jabber notification
- Execute external application

I/O modules

- Inputs
  - Prelude–Manager
  - IDMEF files
- Outputs
  - PostgreSQL
  - Stub (without data saving)

Web user interface

Using web-based UI provides easy access to the users and a fast way to implement new features to the developers. It is a good balance between simplicity and usability.
Example features of the WUI are:

- Incoming alerts list.
- Heatmap of the most often reported source and destination hosts.
- Correlated meta alerts list.
- Alerts types.
- Analyzers list (i.e. facilities reporting alerts).
- Alerts time series.
- ...

Web user interface

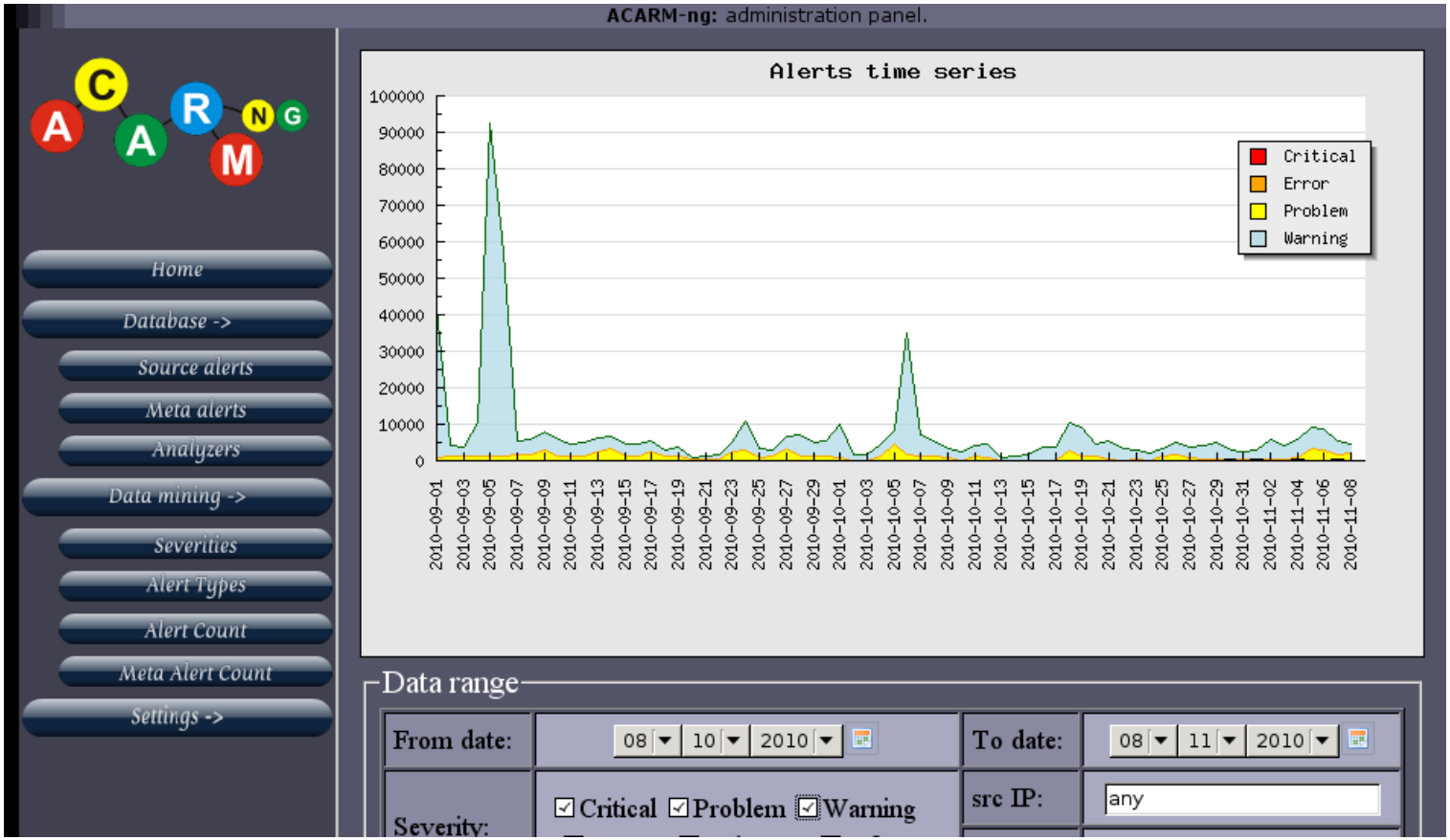


Figure: Alert time series – number of alerts arriving everyday.

Web user interface

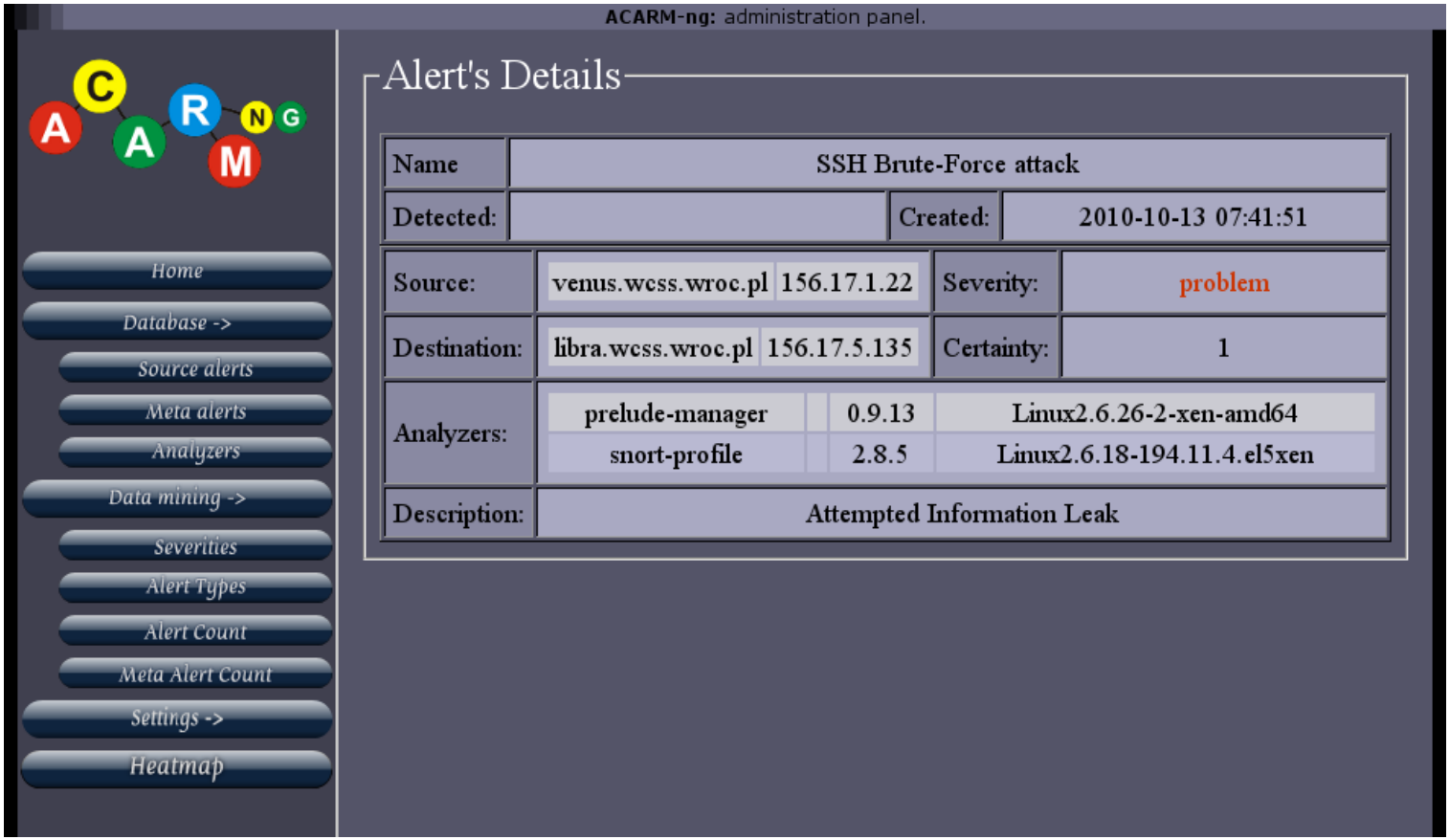


Figure: Example alert.

Web user interface

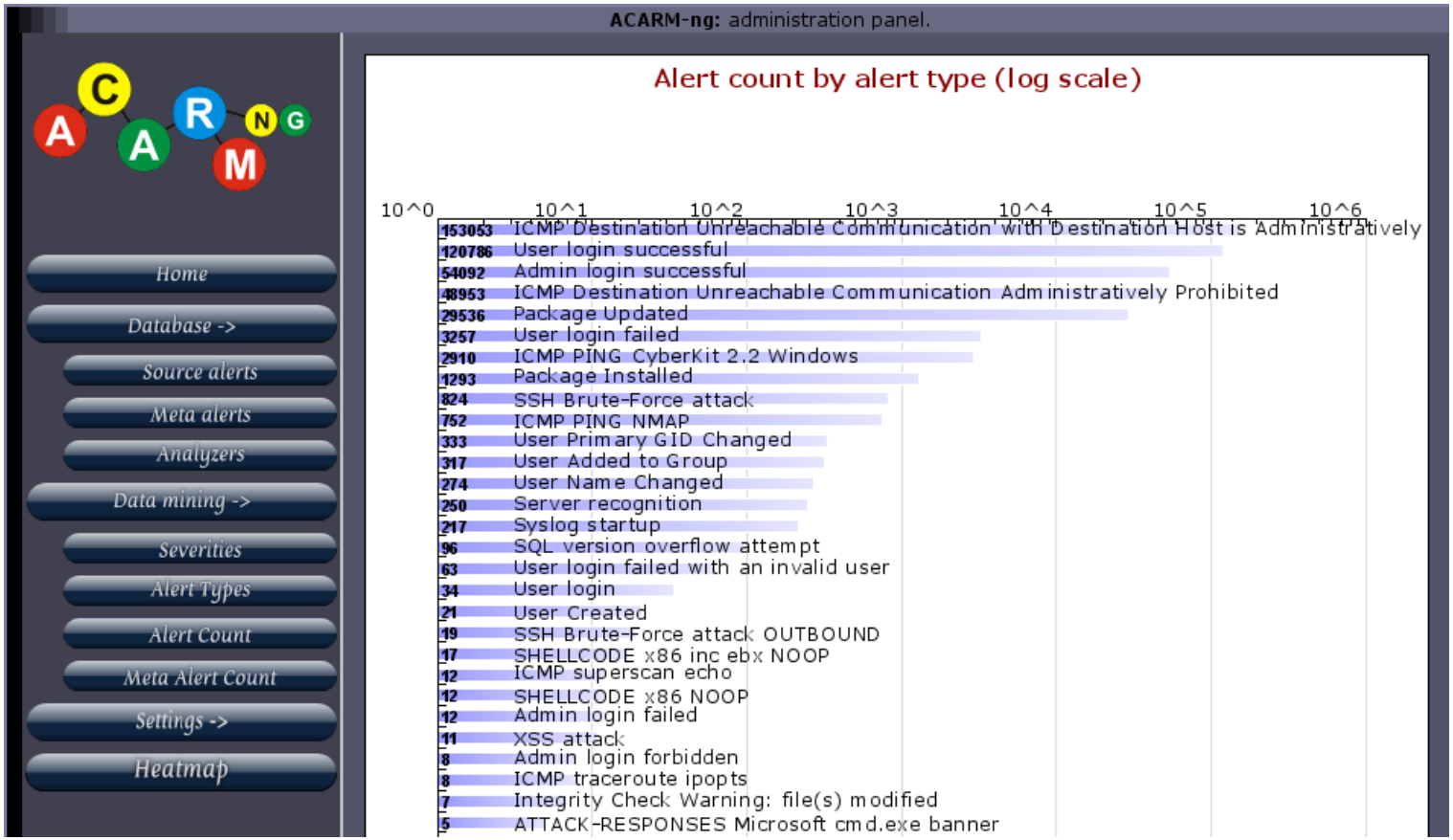


Figure: The most often seen alerts types in a given period.

Web user interface

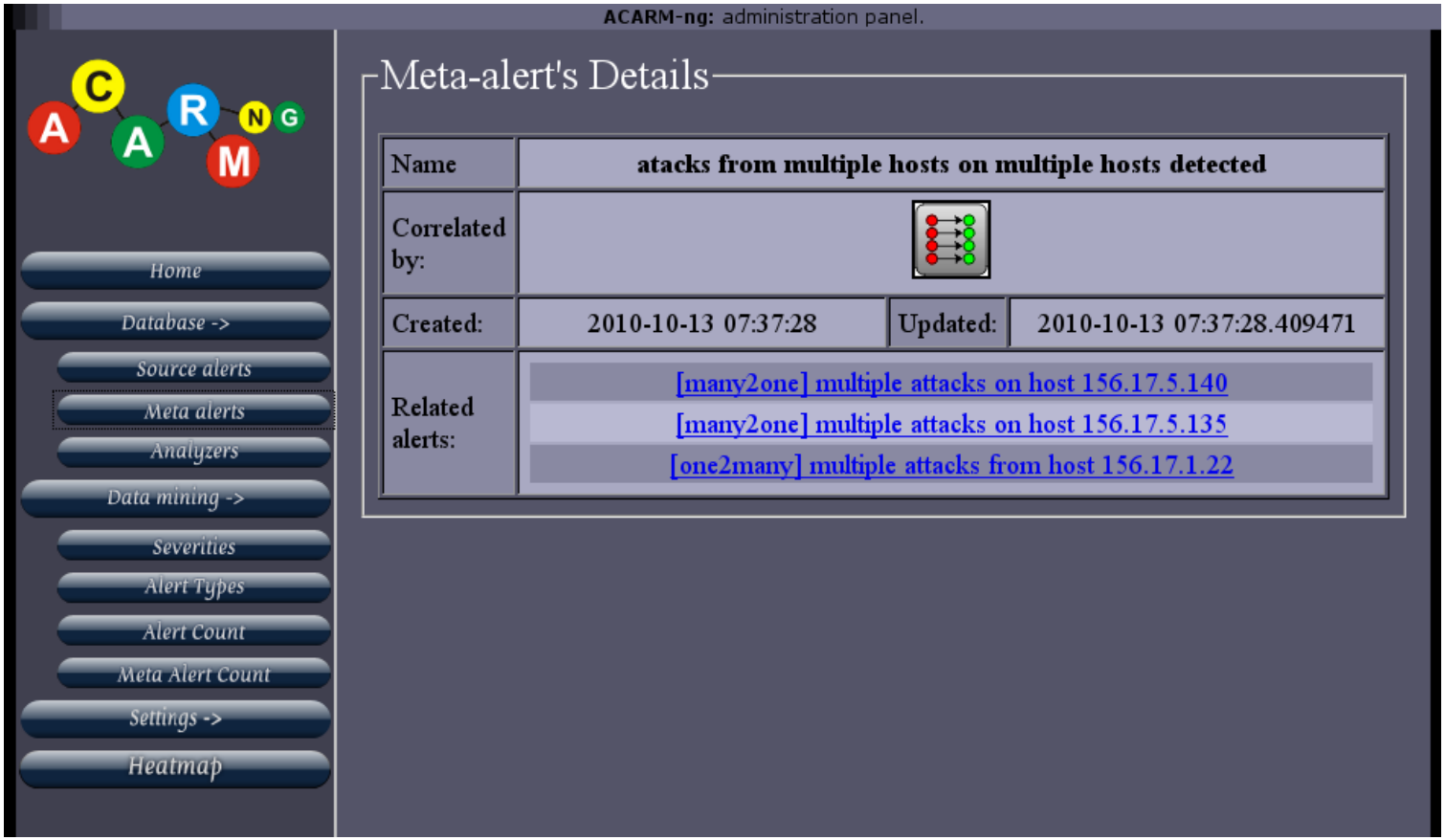


Figure: Example meta-alert.

Web user interface

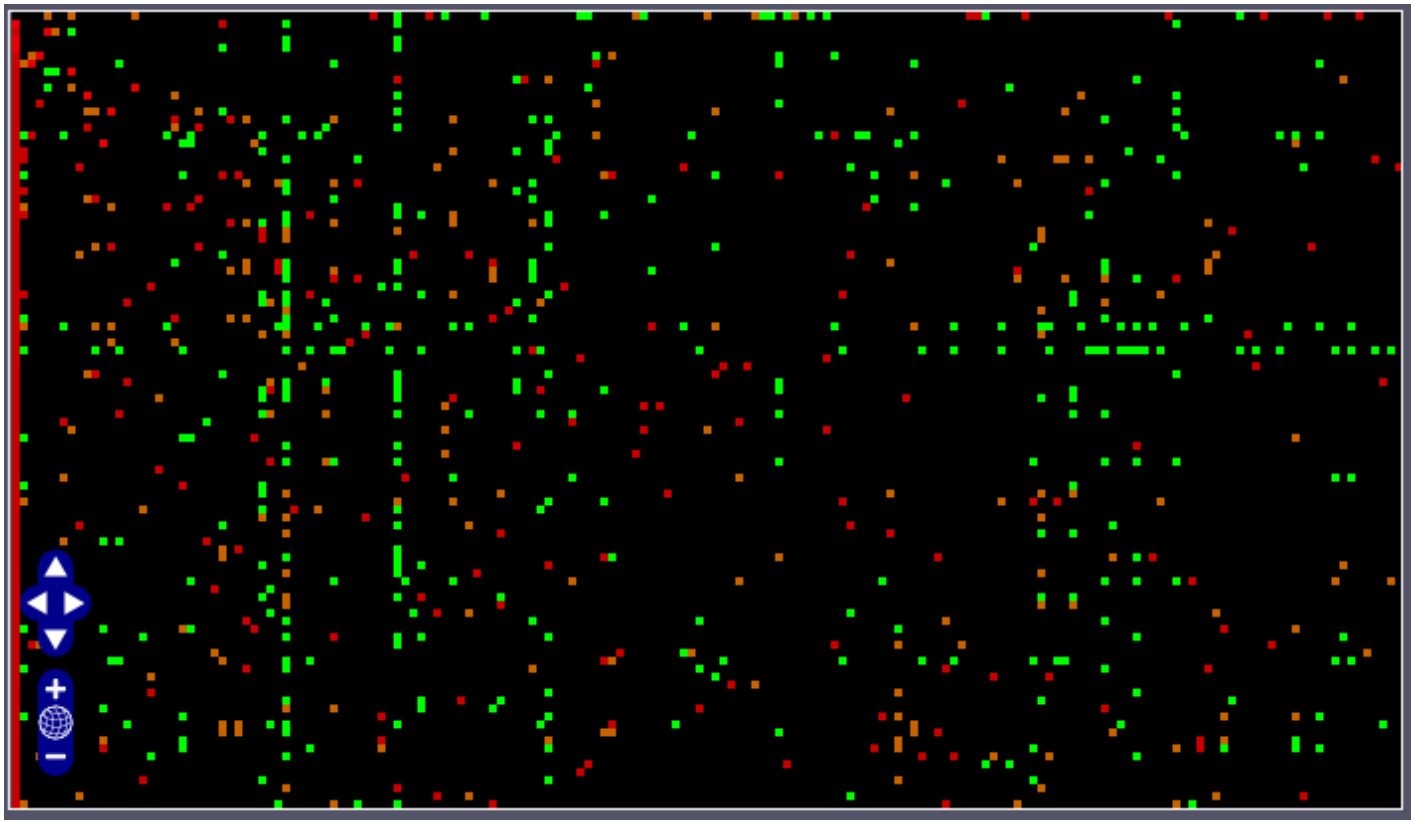


Figure: The most troublesome hosts interactions seen as a heatmap.

Performance

- System implemented in C++
- Low memory usage
- Test results (XEN, 3CPU, 1GB RAM)

Persistency	filters	[alerts/s]
none	none	12000
none	all	1100
PostgreSQL	all	90

- The main performance bottleneck of ACARM-ng is access to the database.

Alerts reduction

- Alerts reduction of ACARM-ng depends heavily on configuration.
- Not all of the reduced (i.e. correlated) alerts are reported as an issue to administrator.
- Different correlation techniques produce different results, while working on the same data.
- This feature of ACARM-ng gives the administrator view of the monitored system security from different sides.

Alerts reduction

filter name	1000*alerts	reports
one to one	54.0	1
many to one	44.4	1
similarity	17.4	6
one to many	13.3	1
users' monitor	12.2	1
event chain	10.7	0
same name	1.2	9

Table: Number of output meta-alerts and reports produced by chosen filters.

Conclusions

- Open source software licensed under GPLv2, thus can be used by anyone, free of charge.
- Each subsystem can be easily substituted without changing system's code.
- Correlated pieces of information are reported while still being processed inside the system.
- Scalable on multi-core architectures.
- Object–oriented design implemented in C++ makes it fast with low memory usage.
- Slow I/O doesn't block whole system.
- Data visualization with WWW.

Future work

- Further work include mostly extending ACARM-ng implementation with new correlation techniques.
- Planned filters include testing artificial intelligence techniques for correlation and using topology-based reasoning (i.e. introducing extra knowledge on network layout).
- Oncoming version v0.3.0, among other extensions, is also planned to have support for Python-written plugins.

The End

Many thanks for your attention.

www.acarm.wcss.wroc.pl

acarm@kdm.wcss.wroc.pl