AUSTRIAN INSTITUTE OF TECHNOLOGY
TOMORROW TODAY

MACHINE LEARNING FÜR LOGDATENANALYSE

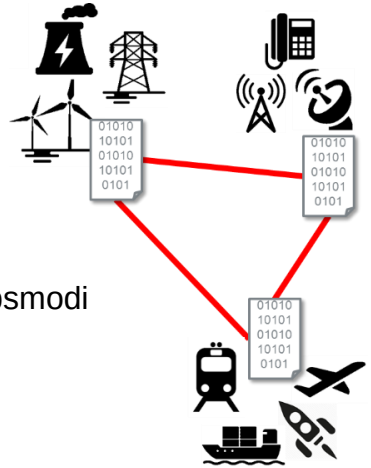
Ein Ausblick auf Morgen

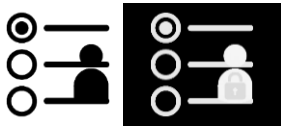
Florian Skopik, florian.skopik@ait.ac.at
Markus Wurzenberger
Max Landauer

01. Okt. 2019

MOTIVATION FÜR MACHINE LEARNING

- IoT, CPS und Industrie 4.0 führen zu einem **immer höheren Grad der Vernetzung** zwischen der physischen und der digitalen Welt
  - Systeme **wachsen „organisch"** und werden nicht mehr „top-down" designed
  - Es gibt oft **kein zentrales Verständnis** dafür, wie ein Gesamtsystem funktioniert.
  - Unterschiedliche Lebenszyklen von IT und OT führen zu **Mischsystemen unterschiedlicher Generationen**.
  - Einfachste Komponenten bieten mannigfaltige Konfigurationseinstellungen und Betriebsmodi
- Folglich ist **jeder komplexe Systemverbund "einzigartig"**
  - Die möglichen Angriffswege steigen rapide
  - **Nicht** alle **möglichen Angriffe** können **antizipiert** werden
  - Blacklisting-Ansätze hinken immer hinterher und sind nicht mehr ausreichend
- Eine mögliche Lösung bietet der **Einsatz von Machine Learning**
  - Um das **Normalverhalten** von komplexen Systemen automatisiert zu **erlernen**
  - Bedrohliche **Abweichungen** (idR. Anomalien) davon zu **erkennen**
  - Situationsabhängig zu alarmieren oder **Gegenmaßnahmen einzuleiten**





WOHER DIE DATEN KOMMEN …

- Rein **Netzwerkbasierte Lösungen** sind gut etabliert und einfach einzubinden, aber…
  - **Ende-zu-Ende Verschlüsselung** verhindert das Erkennen von Malware-Kommunikation
  - **Virtualisierung** verbirgt oft Verkehr/Ereignisse zwischen Maschinen am gleichen Hypervisor
- Daher: Paradigmenwechsel – Viele neuartige **Endpoint-zentrierte Lösungen**
  - EDR = Endpoint Detection and Response
  - Online memory inspection, event sequency mining, pattern recognition, file usage, …
- Der **kleinste gemeinsame Nenner** – und **unsere Spezialisierung** – sind Logdaten von:
  - Dezidierten Geräten (FW, Router, …)
  - Betriebssystemen (bis auf syscall Ebene)
  - Diensten/Applikationen
  - Sensoren (CPS)

WISSENSCHAFTLICHE HERAUSFORDERUNGEN

- Im Bereich **Machine Learning**
  - **Definitionen:** Was ist „Normalverhalten“? Was ist eine Anomalie? FPR?
  - Passender **Lernmodus**: Supervised, semi-supervised, unsupervised learning?
  - Nachvollziehbarkeit der gelernten Modelle und deren **Erklärbarkeit**
  - Woher die **Trainingsdaten** (für individuelle Zielsysteme) nehmen?
- Und **Lodatenanalyse** im speziellen
  - Logdaten sind **komplexe Textdaten**, keine „simplen“ numerischen Sensorikwerte
  - Logdaten besitzen **unbekannte Grammatik** – gelernt wird die Syntax, nicht die Semantik
  - Sehr schnelle Verarbeitung erforderlich um **Onlineverarbeitung** zu erreichen
  - Verfahren mit nur einem Durchlauf („**single pass**“); keine „multi-pass“ Ansätze
  - Im Betrieb ändernde Infrastruktur erfordert **überlappende Lern- und Detektionsphasen**

LOGDATENANALYSE: STAND DER TECHNIK

- **Vordefinierte Parser** für bekannte Ereignistypen und deren Repräsentationen
- **Vordefinierte Auswerteregeln** für antizipierte Ereignisse und Situationen
- **Periodische manuelle Anpassungen** um neue Anforderungen abzudecken

→ POSITIV:

- Hochoptimierte Lösungen
- großer Erfahrungsschatz in der Anwendung
- „proven/tested approach“ in der IT Security

→ NEGATIV:

- aufwändig zu warten und daher teuer
- fehleranfällig wenn manuelle Anpassungen erfolgen
- begrenzt brauchbar um neuartige Angriffe/Anomalien zu erkennen

LOGDATENANALYSE: WIE MACHT MAN LÖSUNGEN „SMART“?

- Lernen der automatisierten **Unterscheidung von Ereignissen** unterschiedlichen Typs
  - mittels inkrementellem Online Clusterings, stream-basiert, 1-pass Verfahren
- Lernen von gemeinsamen vorab **unbekannten Strukturen** in Ereignissen unterschiedlichen Typs
  - Mittels „Template Generierung“ unter Anwendung div. Alignment Ansätze und Metriken
- Lernen **Datenfelder** in Text unbekannter Grammatik zu identifizieren
  - Mittels „Tree-Parser Generation“ und Segmentierung in statische und variable Textteile
- Lernen des **normalen Systemverhaltens**
  - Mittels Bildung von Hypothesen und Abbildung von Ereignis-Abhängigkeiten in Korrelationsregeln
- Lernen von **Veränderungen des Verhaltens über der Zeit**
  - Mittels Zeitreihenanalysemethoden
- Lernen **komplexer Ausnahmesituationen**
  - Mittels Methoden zur intelligenten Alarm-Aggregierung

FORSCHUNGSPROGRAMM „ANOMALIEERKENNUNG UND THREAT INTEL“

Die Initiative in Zahlen -- https://aecid.ait.ac.at/

- **10** Patente zur Lösung der aufgezählten Herausforderungen
- **9** Jahre Arbeit, seit 2011
- **8** Personen involviert
- **7** Forschungsprojekte (national und international)
- **1** Lösung: AECID Komponente/PoC AECID (verfügbar auf launchpad und in Debian/Ubuntu)

AECID - Automatic Event Correlation for Incident Detection

- Verarbeitet Systemereignisse in Logdatenform und **analysiert sequentiell generierte textuelle Logdaten** (z.B. syslog, journald) welche das aktuelle Systemverhalten widerspiegeln
- **Lernt** dynamisch das **Normalverhalten** von Systemen bzw. das normale Nutzungsverhalten -- ermittelt Klasse, Abhängigkeiten, Parameter, Werteverteilungen und Auftreten von Ereignissen
- Ermittelt Abweichungen von erwartetem Normalzustand → Anomalieerkennung

WELCHE ANOMALIEN KÖNNEN WIR ERKENNEN?

- **Punkt-Anomalien**
  - Client mit unbekanntem User Agent (z.B. Chrome anstatt Firefox)
  - **Whitelisting**: gelernt wurde, dass nur Firefox „normal“ ist, alles andere triggert einen Alarm
  - **Blacklisting:** manuelle Definition, dass Chrome, Edge, Opera, IE usw. problematisch sind
- **Anormale Ereignis-Parameter (Kombinationen)**
  - z.B. Zugriff ausserhalb der Dienstzeiten oder von ungewöhnlichem Rechner/App aus
- **Anormale Ereignisfrequenz**
  - Z.B. Datendiebstahl aufgrund anormal hoher Anzahl an Zugriffen auf die Datenbank von einem einzelnen Rechner aus; oder Ransomware aufgrund sequentieller Dateizugriffe.
- **Anormale Ereignisabfolge**
  - Z.B. gelernt wurde Kette an Ereignissen bei Zugriff auf Web Server; nun erfolgt Zugriff auf Web Server ohne Verbindung über Firewall

AECID CONCEPT

- **Online log based anomaly detection:**
  - Monitor any (unstructured) **textual event data** (e.g., syslog, windows event log)
  - **Self-learning** and **whitelisting** → no attack signatures required
  - No semantic interpretation → only constant syntax
  - Automatic detection of **relevant log parts**

- → **Flexible** and **domain-independent** general applicable solution:
  - **Network**-, **application**- and **cross-layer** usage
  - Coverage of legacy systems, systems with small market shares and poor documentation → **no signatures and parsers** exist

- Prerequisite: logging

THE ÆCID APPROACH

1. **Log parser** generation
   - A “recipe” on how to dissect log lines of unknown grammar
   - Make log data usable for analysis → structured representation & easy to access
2. **Hypotheses** proposal
   - Distribution of property values (e.g., IP addresses, user names, …) in single events
   - And across multiple events
   - Correlation of event types
3. **Rule** generation through continuous hypotheses evaluation
   - Sort out unstable hypotheses and create rules for stable ones
   - Constitution of the system behavior model (learned behavior model)
4. **Anomaly detection**: rate the deviation of actual system behavior from the learned behavior model (anomalous points / context / frequency / sequence of events)

All steps take place in parallel, i.e., even during the anomaly detection phase, new hypotheses are created on the fly.

STEP 1: PARSING – FAST LOG DATA PROCESSING

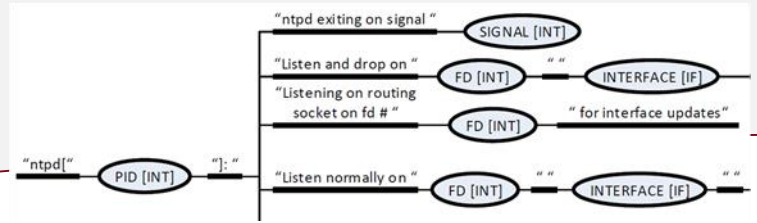
- **Parser model:** describes system behavior
  - **Loglines** represented as **tree-like graph (O(log(n))** → Parse data once!
  - **No regex (O(n))** for whole line required → **fast line processing**, rule evaluation
  - → **Online Anomaly Detection**
  - Describe information most efficiently – with **minimal storage requirements**
  - **Efficient** log line **classification**

```
Dec 15 00:10:27 www0.some.domain apache: 30086 10.0.0.1:80 "www.seite.at"
"www.seite.at" 192.168.0.1 - - [15/Dec/2015:00:10:27 +0000] 126 "GET / HTTP/1.1" 302
212 "-" „Monitoring Agent„
```

/model/syslog/time: 2015-12-15 00:10:27

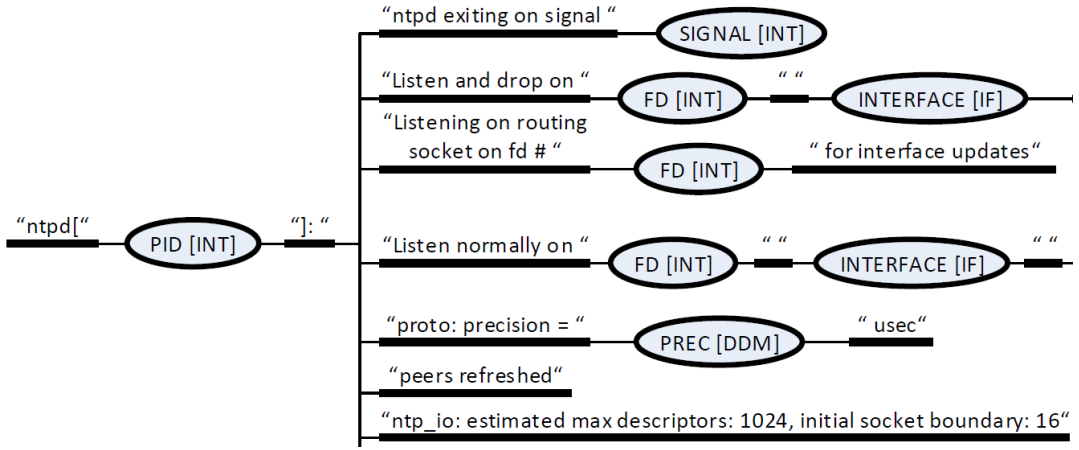
/model/syslog/host: www0.some.domain

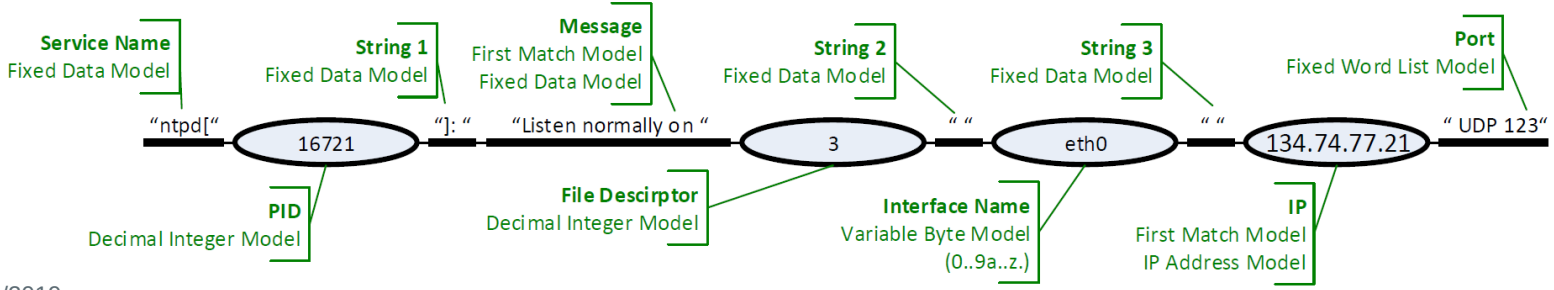
/model/services/apache/sname: apache



APPROACH – STEP 1: PARSING

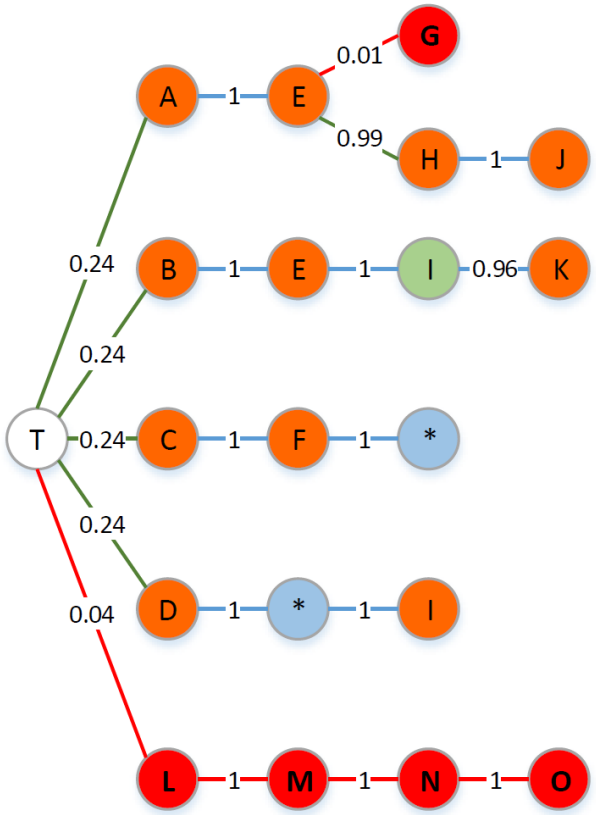
```
ntpd[16721]: Listen and drop on 0
   v4wildcard 0.0.0.0 UDP 123
ntpd[16721]: Listen and drop on 1
   v6wildcard :: UDP 123
ntpd[16721]: Listen normally on 2 lo
   127.0.0.1 UDP 123
ntpd[16721]: Listen normally on 3
   eth0 134.74.77.21 UDP 123
ntpd[16721]: Listen normally on 4
   eth1 10.10.0.57 UDP 123
```

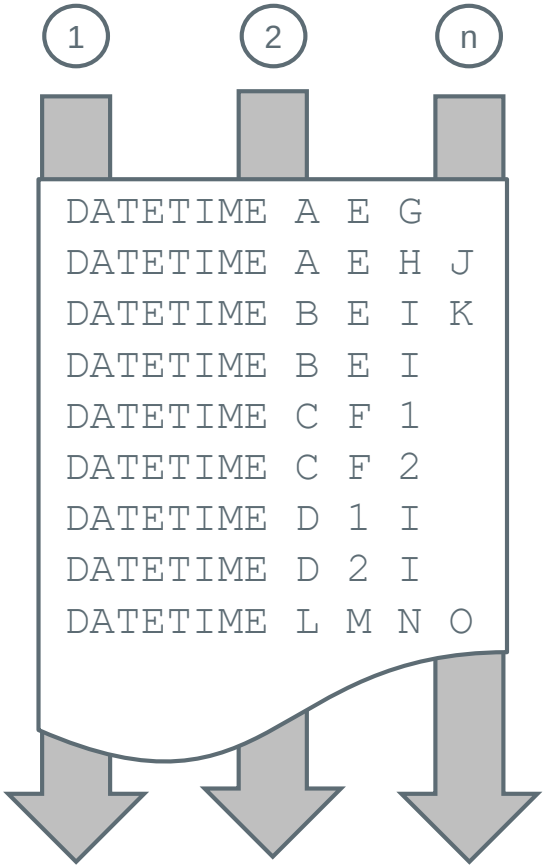




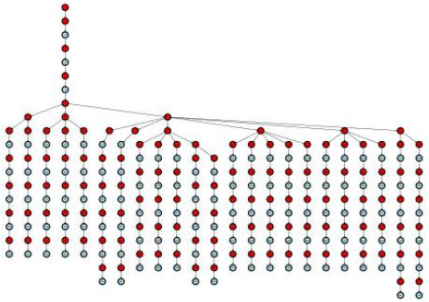
PARSER-GENERATOR

- Legend:
  - Fixed
  - Optional
  - Variable
  - Deleted

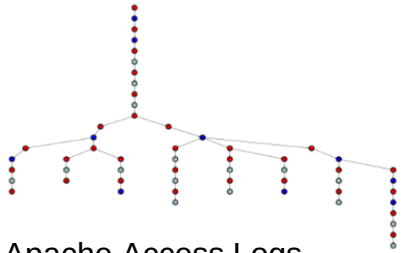




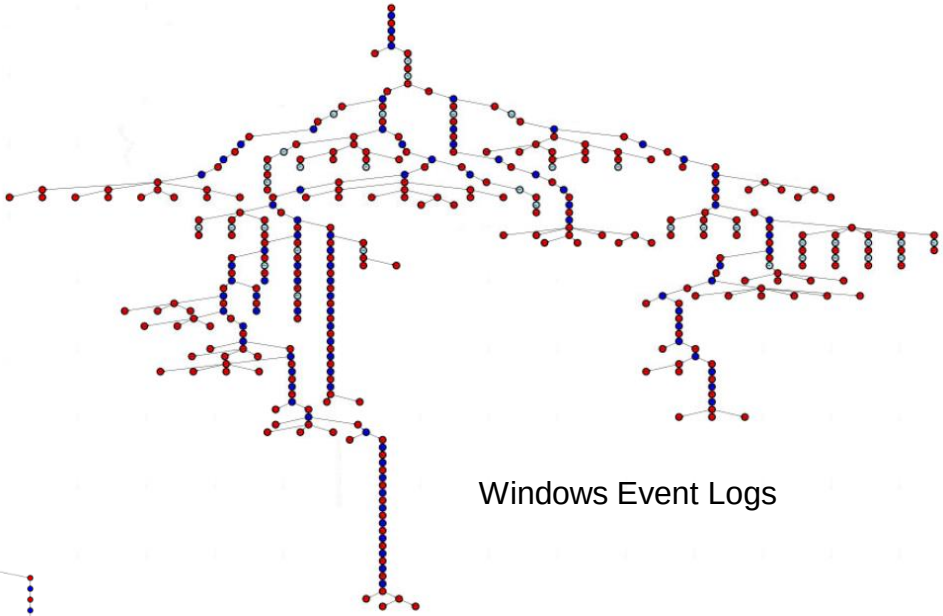
PARSER TREES: REAL-WORLD EXAMPLES



Suricata Logs

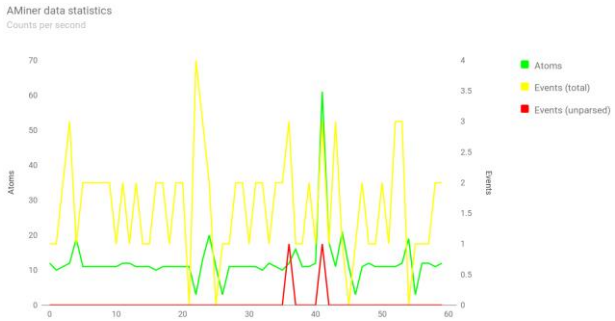


Apache Access Logs

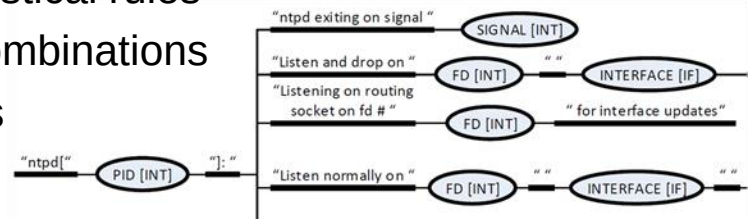


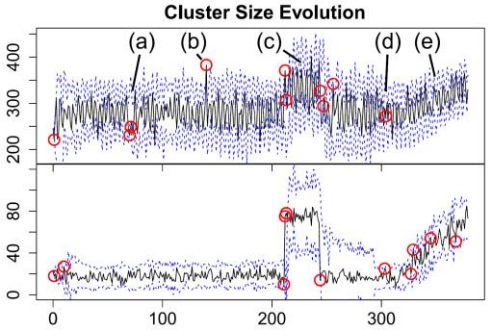
Windows Event Logs

AECID DETECTION MECHANISMS



- **Whitelisting** to overcome limitations of blacklisting
  - Detect unknown patterns
  - Signatures can be evaded by modification of attacks
  - Blacklists only flag clearly malicious behavior:
    - **Behavior acceptable in one context**, e.g. system update replaces files, **is clearly abnormal in different context**, e.g. graphic card firmware loading component replaces files
    - Normal system operation, **only some system states are encountered** even if plenty of system states (rare states, error states) are possible
- **Rule-based** detection to monitor complex system processes
  - Correlation and statistical rules
  - Values and value combinations
  - Time series analysis





APPROACH – STEP 2: HYPOTHESES PROPOSAL (1/2)

```
Jun 20 00:59:37 localhost sshd[1008]: Accepted public key for backup
from 172.29.147.33 port 54149 ssh2: RSA SHA256:9k...
```

/model/syslog/time: **Jun 20 00:59:37**
/model/syslog/host: **localhost**
/model/services/sshd/sname: **sshd**
/model/services/sshd/msg/acceptedpk/pid: **1008**
/model/services/sshd/msg/acceptedpk/user: **backup**
/model/services/sshd/msg/acceptedpk/originip: **172.29.147.33**
/model/services/sshd/msg/acceptedpk/port: **54149**
/model/services/sshd/msg/acceptedpk/protocol: **ssh2**
/model/services/sshd/msg/acceptedpk/crypto: **RSA**
/model/services/sshd/msg/acceptedpk/fingerprint: **SHA256:9k...**

Simple Example Hypotheses:

```
user{backup} ~ remoteip{172.29.147.33}
user{backup} ~ fingerprint{SHA256:9k...}
user{backup} only allowed in time_hh{[00,03]}
...
```

- Different Methods for hypothesis generation (incl. brute force)
- Coverage of events is complex to determine
- Maximize detection capabilities with minimum number of (stable) hypotheses
- Continuous learning in parallel to detection

APPROACH – STEP 2: HYPOTHESES PROPOSAL (2/2)

Firewall Logs

permitted HTTP traffic sourced from inside (eth1) with NAT (Check Point FW/VPN 1)

```
Dec 15 09:10:26 accept
www0.some.domain >eth1 product VPN-1 &
Firewall-1  src 10.0.0.1 s_port 45213
dst 192.168.0.1 service http proto
tcp  xlatesrc 192.168.0.10 rule 5
```

Web Server logs

Ressource retrieval via HTTP on Apache Webserver

```
Dec 15 09:10:27 www0.some.domain
apache: 30086 192.168.0.1:80
"www.page.at" "www.page.at"
192.168.0.10 - - [15/Dec/2015:09:10:27
+0000] 126 "GET / HTTP/1.1" 302 212 "-
" "Mozilla/5.0"
```

Cross-System Example Hypotheses:

- event „HTTP retrieval“ on Apache with parameters „**www.page.at**“ conditions „permit HTTP“ from src={**10.0.0.1, ...**} on FW in a time window of 5000ms
- src={**192.168.0.10**} in „HTTP retrieval“ ~ src={**10.0.0.1**} in „permit HTTP“ in a time window of 5000ms
- ...

AECID - ARCHITECTURE

AMiner

- Lightweight base implementation
- Parses log lines
- Verifies rules
- Triggers alarms
- License: open source

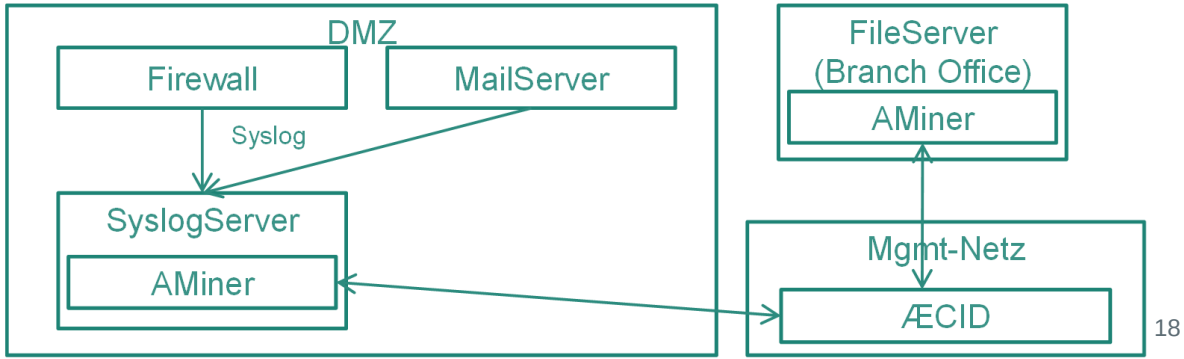
launchpad

https://git.launchpad.net/logdata-anomaly-miner

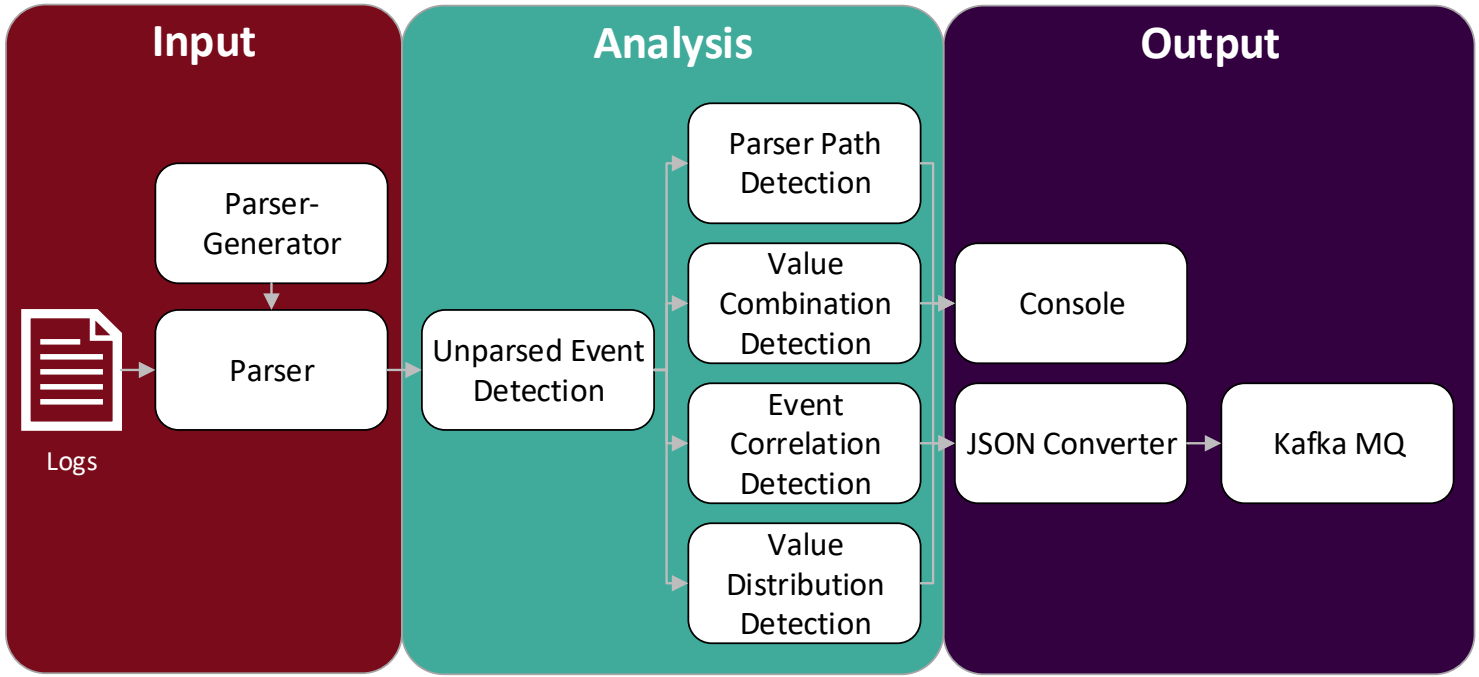
debian

AECID Central

- Intelligent control center
- Receives unknown log lines from AMiner instances
- Distributes and adapts system model and rule-set
- License: commercial https://aecid.ait.ac.at/



DEMONSTRATION



TAKE AWAY MESSAGES

- Besuchen Sie uns auf unserem **Stand**!
- Besuchen Sie uns auf unserer **Homepage**: https://aecid.ait.ac.at/
- Der AMiner ist Open Source verfügbar und darf auch kommerziell eingesetzt werden
  - **Bezugsquellen**: Debian/Ubuntu; Launchpad https://launchpad.net/logdata-anomaly-miner
- **AMiner Workshop**: Interessenten an einem kostenlosen Halbtagesworkshop wenden sich bitte an uns (persönlich oder via E-Mail)!
- **AECID** ist jene Toolbox, die darüber hinausgehende smarte Komponenten beinhaltet
  - Bitte kontaktieren Sie uns bei Interesse an einer Evaluierung oder Zusammenarbeit

AUSTRIAN INSTITUTE OF TECHNOLOGY
TOMORROW TODAY

VIELEN DANK FÜR IHRE AUFMERKSAMKEIT!

Florian Skopik, Markus Wurzenberger, Max Landauer

01. Oktober 2019