3 nietypowe metody analizy aplikacji, czyli jak admin pomoże programiście.

CodeEurope

Piotr Pyciński

26 kwietnia 2017

Agenda

1. Pomysł na prezentację
2. Sysdig
   - Co to jest ?
   - Jak działa ?
3. Volatility
   - Co to jest ?
   - Jak działa ?
4. OSSEC
   - Co to jest ?
   - Dlaczego ?
   - Jak działa ?

O mnie słów kilka

Kto: Piotr Pyciński
Jak mówić: pyton
Co robię:

- Senior System Administrator @ Alior Bank
- Teacher @ Pedagogical University of Cracow

```
-----BEGIN GEEK CODE BLOCK-----
Version: 3.1
GCS/ED d-(-) s(): a C++(++++) UBL++++ P++ L+++
E--- !W+ N o K-- !w- !O !M !V PS PE Y PGP t+
5- X R tv- b+ DI D G+>++ e+++ h---- r+++ z+++
------END GEEK CODE BLOCK------
```

Dlaczego nietypowe, co ja tutaj robię ?

- Ponieważ logi nie zawsze wystarczają !
- Aby pokazać inne spojrzenie na rzeczywistość i programy diagnostyczne.
- Odświeżyć wiedzę administratorów starszej daty.
- Opisane programy są względnie "bezpieczne".
- Pokazane techniki, metody mogą wykorzystać programiści - starałem się takie dobrać.
- Administratorzy od dzisiaj mnie nie lubią ?

Sysdig

https://github.com/draios/sysdig

Co to jest ?

Sysdig, to tak naprawdę inny strace i coś więcej:

- Analiza na poziomie systemu operacyjnego.

Sysdig, to tak naprawdę inny strace i coś więcej:

- Analiza na poziomie systemu operacyjnego.
- Jeden spójny interface dla sieci, procesów, plików, pamięci.

Sysdig, to tak naprawdę inny strace i coś więcej:

- Analiza na poziomie systemu operacyjnego.
- Jeden spójny interface dla sieci, procesów, plików, pamięci.
- Bazuje głównie na syscall.

Sysdig, to tak naprawdę inny strace i coś więcej:

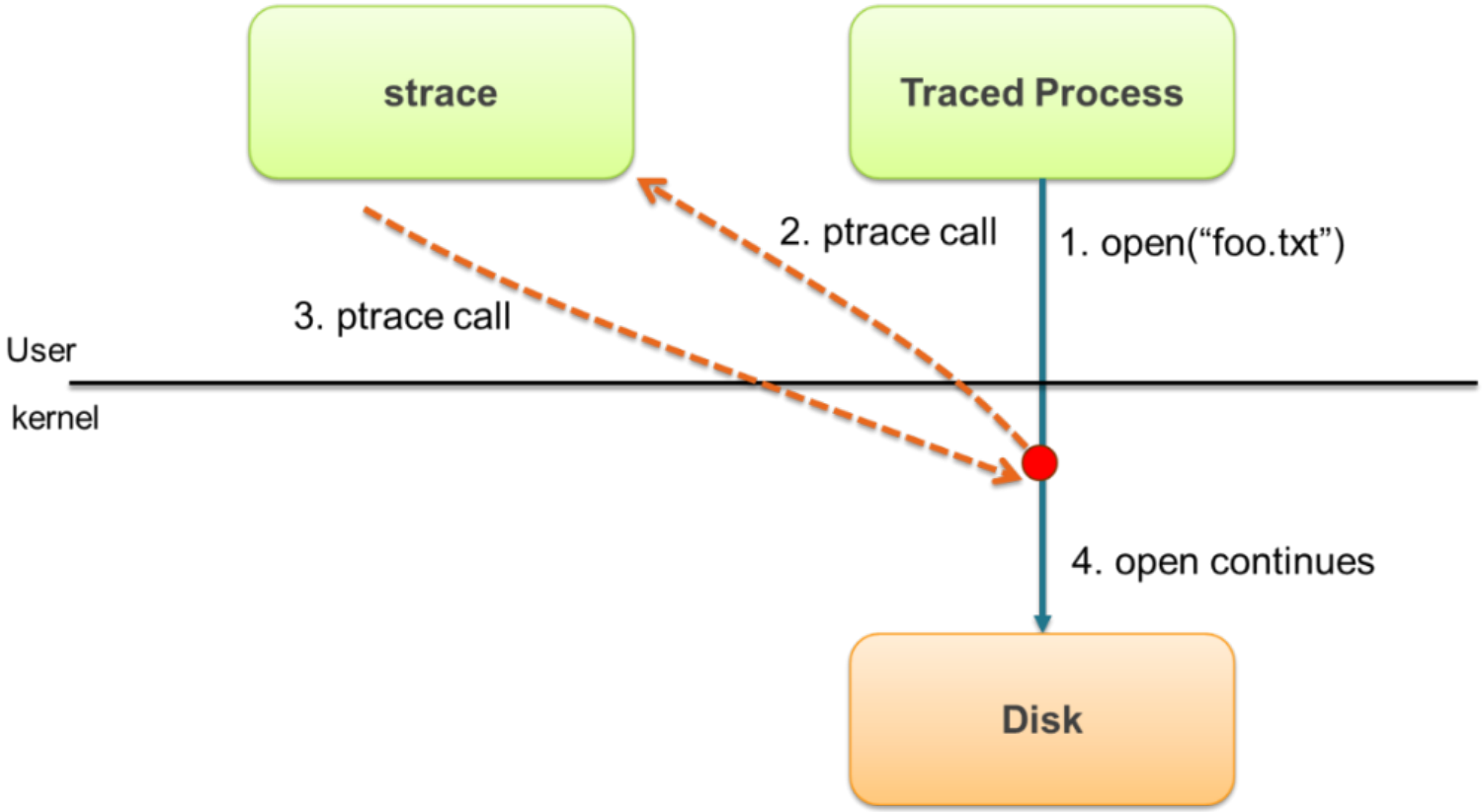
- Analiza na poziomie systemu operacyjnego.
- Jeden spójny interface dla sieci, procesów, plików, pamięci.
- Bazuje głównie na syscall.
- Możliwość zapisania (jak w wireshark) trace file.

Sysdig, to tak naprawdę inny strace i coś więcej:

- Analiza na poziomie systemu operacyjnego.
- Jeden spójny interface dla sieci, procesów, plików, pamięci.
- Bazuje głównie na syscall.
- Możliwość zapisania (jak w wireshark) trace file.
- *Połączenie strace + tcpdump + htop + iftop + lsof + ...*

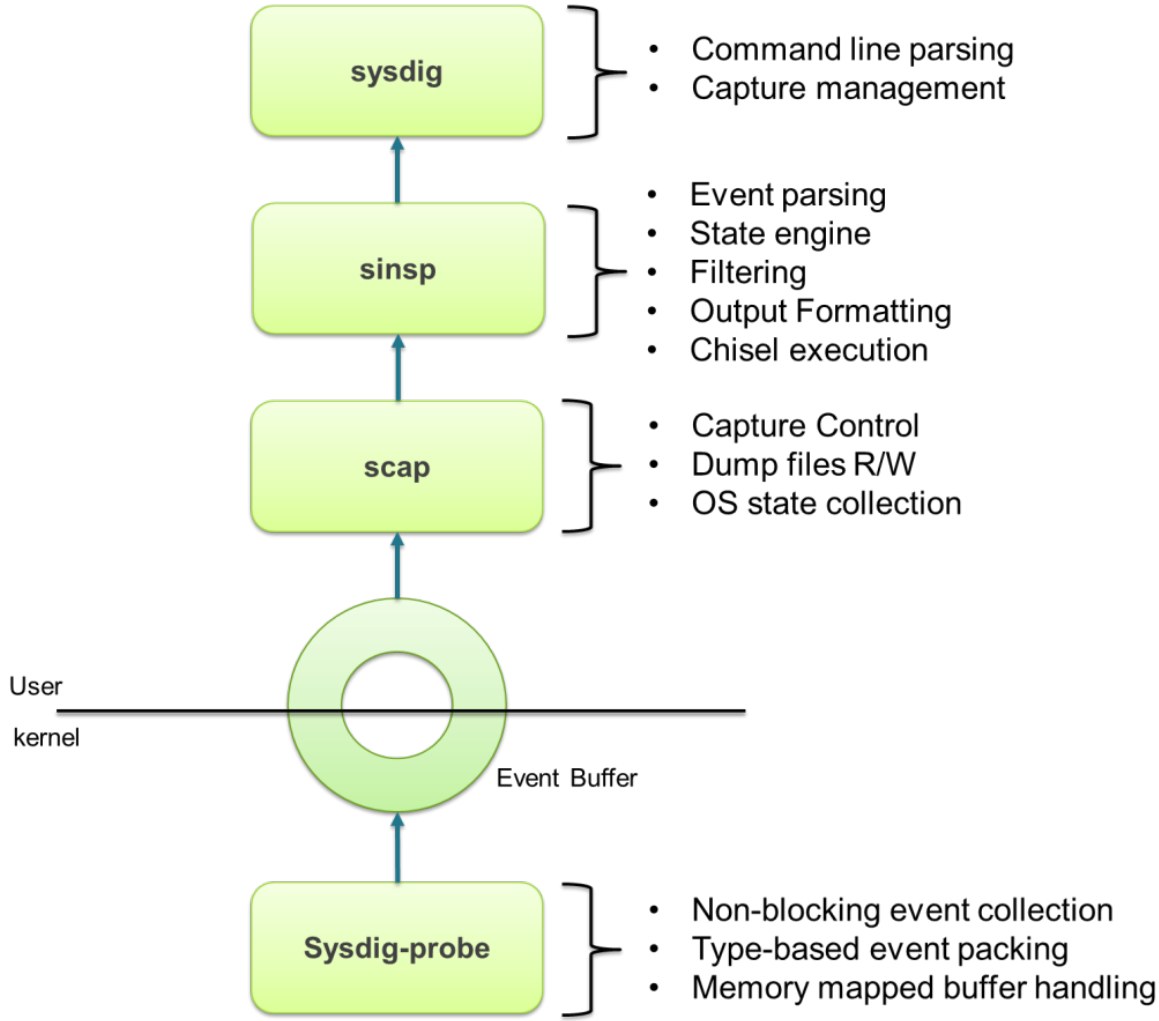
Co to jest ?

strace vs sysdig



Rysunek: STRACE i ptrace

Natomiast Sysdig używa mechanizmu zwanego Tracepoints i własnego drivera nazwanego sysdig-probe.



Tracepoints make it possible to install a "handler" that is called from specific functions in the kernel. Currently, sysdig registers tracepoints for system calls on enter and exit, and for process scheduling events.

Domyślny format prezentowanych danych:

```
event.number event.time event.cpu proces.name thread event.direction event.type event.arguments

2273 11:22:25.251263661 0 php5-fpm (1830) < read res=15 data=Testowy string.
```

Zalety

- **Jest nie blokujący !**

Zalety

- **Jest nie blokujący !**
- **Dekoduje FD** (adresy ip, nazwy plików)

Zalety

- **Jest nie blokujący !**
- **Dekoduje FD** (adresy ip, nazwy plików)
- **Zapis do pliku**

```
sysdig -w myfile.scap
```

Zalety

- **Jest nie blokujący !**
- **Dekoduje FD** (adresy ip, nazwy plików)
- **Zapis do pliku**
  `sysdig -w myfile.scap`
- **Dzielenie plików po rozmiarze**
  `sysdig -C 1 -w dump.scap`

Zalety

- **Jest nie blokujący !**
- **Dekoduje FD** (adresy ip, nazwy plików)
- **Zapis do pliku**
  `sysdig -w myfile.scap`
- **Dzielenie plików po rozmiarze**
  `sysdig -C 1 -w dump.scap`
- **Filtrowanie**
  `proc.name!=cat and evt.type=open`

Zalety

- **Jest nie blokujący !**
- **Dekoduje FD** (adresy ip, nazwy plików)
- **Zapis do pliku**
  `sysdig -w myfile.scap`
- **Dzielenie plików po rozmiarze**
  `sysdig -C 1 -w dump.scap`
- **Filtrowanie**
  `proc.name!=cat and evt.type=open`
- **Chisels** czyli małe skrypty w LUA.

Zalety

- **Jest nie blokujący !**
- **Dekoduje FD** (adresy ip, nazwy plików)
- **Zapis do pliku**
  `sysdig -w myfile.scap`
- **Dzielenie plików po rozmiarze**
  `sysdig -C 1 -w dump.scap`
- **Filtrowanie**
  `proc.name!=cat and evt.type=open`
- **Chisels** czyli małe skrypty w LUA.
- **Obsługuje kontenery.**

Zalety

- **Jest nie blokujący !**
- **Dekoduje FD** (adresy ip, nazwy plików)
- **Zapis do pliku**
  `sysdig -w myfile.scap`
- **Dzielenie plików po rozmiarze**
  `sysdig -C 1 -w dump.scap`
- **Filtrowanie**
  `proc.name!=cat and evt.type=open`
- **Chisels** czyli małe skrypty w LUA.
- **Obsługuje kontenery.**
- **csysdig.**

Demo

Volatility

https://github.com/volatilityfoundation

Co to jest ?

- Advanced memory forensics framework.

- Advanced memory forensics framework.
- Zastaw narzędzi, pluginów, profili do analizy zrzutów pamięci.

- Advanced memory forensics framework.
- Zastaw narzędzi, pluginów, profili do analizy zrzutów pamięci.
- Początki w 2007 roku na Black Hat. Volatility Foundation.

- Advanced memory forensics framework.
- Zastaw narzędzi, pluginów, profili do analizy zrzutów pamięci.
- Początki w 2007 roku na Black Hat. Volatility Foundation.
- **Windows 32 i 64 bit** - wszystkie wersje, raw format, hibernacja.

- Advanced memory forensics framework.
- Zastaw narzędzi, pluginów, profili do analizy zrzutów pamięci.
- Początki w 2007 roku na Black Hat. Volatility Foundation.
- **Windows 32 i 64 bit** - wszystkie wersje, raw format, hibernacja.
- **Wspiera VM** - vmware, virtualbox, qemu - snapshoot i core dumpy.

- Advanced memory forensics framework.
- Zastaw narzędzi, pluginów, profili do analizy zrzutów pamięci.
- Początki w 2007 roku na Black Hat. Volatility Foundation.
- **Windows 32 i 64 bit** - wszystkie wersje, raw format, hibernacja.
- **Wspiera VM** - vmware, virtualbox, qemu - snapshoot i core dumpy.
- **GNU/Linux** (LiME) - https://github.com/504ensicsLabs/LiME.

- Advanced memory forensics framework.
- Zastaw narzędzi, pluginów, profili do analizy zrzutów pamięci.
- Początki w 2007 roku na Black Hat. Volatility Foundation.
- **Windows 32 i 64 bit** - wszystkie wersje, raw format, hibernacja.
- **Wspiera VM** - vmware, virtualbox, qemu - snapshoot i core dumpy.
- **GNU/Linux** (LiME) - https://github.com/504ensicsLabs/LiME.
- **OSX** - od 10.5 do 10.9.4.

- Advanced memory forensics framework.
- Zastaw narzędzi, pluginów, profili do analizy zrzutów pamięci.
- Początki w 2007 roku na Black Hat. Volatility Foundation.
- **Windows 32 i 64 bit** - wszystkie wersje, raw format, hibernacja.
- **Wspiera VM** - vmware, virtualbox, qemu - snapshoot i core dumpy.
- **GNU/Linux** (LiME) - https://github.com/504ensicsLabs/LiME.
- **OSX** - od 10.5 do 10.9.4.
- **Android**.

- Advanced memory forensics framework.
- Zastaw narzędzi, pluginów, profili do analizy zrzutów pamięci.
- Początki w 2007 roku na Black Hat. Volatility Foundation.
- **Windows 32 i 64 bit** - wszystkie wersje, raw format, hibernacja.
- **Wspiera VM** - vmware, virtualbox, qemu - snapshoot i core dumpy.
- **GNU/Linux** (LiME) - https://github.com/504ensicsLabs/LiME.
- **OSX** - od 10.5 do 10.9.4.
- **Android**.
- Napisany w python.

Jak to działa ?

- Robimy dump (lokalny, sieciowy)
- Wybieramy profil OS'a
- Wybieramy plugin którym chcemy dokonać analizy
- Czytamy CheatSheet lub –info –help

Demo

OSSEC

https://ossec.github.io/

Co to jest ?

OSSEC to :

1. HIDS ?

OSSEC to :

1. HIDS ?
2. LIDS ?

OSSEC to :

1. HIDS ?
2. LIDS ?
3. FIC ?

OSSEC to :

1. HIDS ?
2. LIDS ?
3. FIC ?
4. IDS ?

OSSEC to :

1. HIDS ?
2. LIDS ?
3. FIC ?
4. IDS ?
5. IPS ?

OSSEC is a free, ***open-source host-based intrusion detection system*** *(HIDS). It performs log analysis, integrity checking, Windows registry monitoring, rootkit detection, time-based alerting, and active response. It provides intrusion detection for most operating systems, including Linux, OpenBSD, FreeBSD, OS X, Solaris and Windows.*

- OSSEC analizuje logi i na ich podstawie podejmuje akcję, wysyła maila, wpisuje do alert loga.

- OSSEC analizuje logi i na ich podstawie podejmuje akcję, wysyła maila, wpisuje do alert loga.
- Tani w implementacji.

- OSSEC analizuje logi i na ich podstawie podejmuje akcję, wysyła maila, wpisuje do alert loga.
- Tani w implementacji.
- Widoczność protokołów, które używają szyfrowania w komunikacji sieciowej.

- OSSEC analizuje logi i na ich podstawie podejmuje akcję, wysyła maila, wpisuje do alert loga.
- Tani w implementacji.
- Widoczność protokołów, które używają szyfrowania w komunikacji sieciowej.
- Możemy podpinać wszystko, ważne aby logowało.

- OSSEC analizuje logi i na ich podstawie podejmuje akcję, wysyła maila, wpisuje do alert loga.
- Tani w implementacji.
- Widoczność protokołów, które używają szyfrowania w komunikacji sieciowej.
- Możemy podpinać wszystko, ważne aby logowało.
- FIC - *File Integrity Checking*. Sumy kontrolne plików.

- OSSEC analizuje logi i na ich podstawie podejmuje akcję, wysyła maila, wpisuje do alert loga.
- Tani w implementacji.
- Widoczność protokołów, które używają szyfrowania w komunikacji sieciowej.
- Możemy podpinać wszystko, ważne aby logowało.
- FIC - *File Integrity Checking*. Sumy kontrolne plików.
- Audytuje rejestr systemów Windows.

- OSSEC analizuje logi i na ich podstawie podejmuje akcję, wysyła maila, wpisuje do alert loga.
- Tani w implementacji.
- Widoczność protokołów, które używają szyfrowania w komunikacji sieciowej.
- Możemy podpinać wszystko, ważne aby logowało.
- FIC - *File Integrity Checking*. Sumy kontrolne plików.
- Audytuje rejestr systemów Windows.
- RootKit detection.

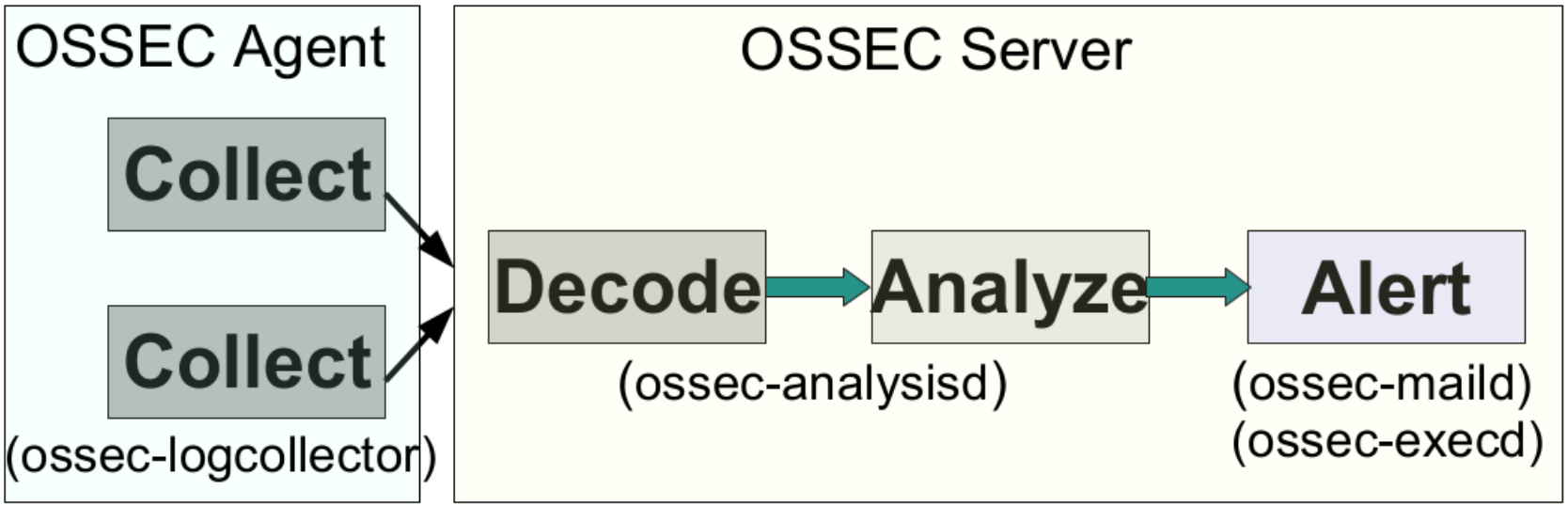
- OSSEC analizuje logi i na ich podstawie podejmuje akcję, wysyła maila, wpisuje do alert loga.
- Tani w implementacji.
- Widoczność protokołów, które używają szyfrowania w komunikacji sieciowej.
- Możemy podpinać wszystko, ważne aby logowało.
- FIC - *File Integrity Checking*. Sumy kontrolne plików.
- Audytuje rejestr systemów Windows.
- RootKit detection.
- Command output monitoring.

- OSSEC analizuje logi i na ich podstawie podejmuje akcję, wysyła maila, wpisuje do alert loga.
- Tani w implementacji.
- Widoczność protokołów, które używają szyfrowania w komunikacji sieciowej.
- Możemy podpinać wszystko, ważne aby logowało.
- FIC - *File Integrity Checking*. Sumy kontrolne plików.
- Audytuje rejestr systemów Windows.
- RootKit detection.
- Command output monitoring.
- Active response.

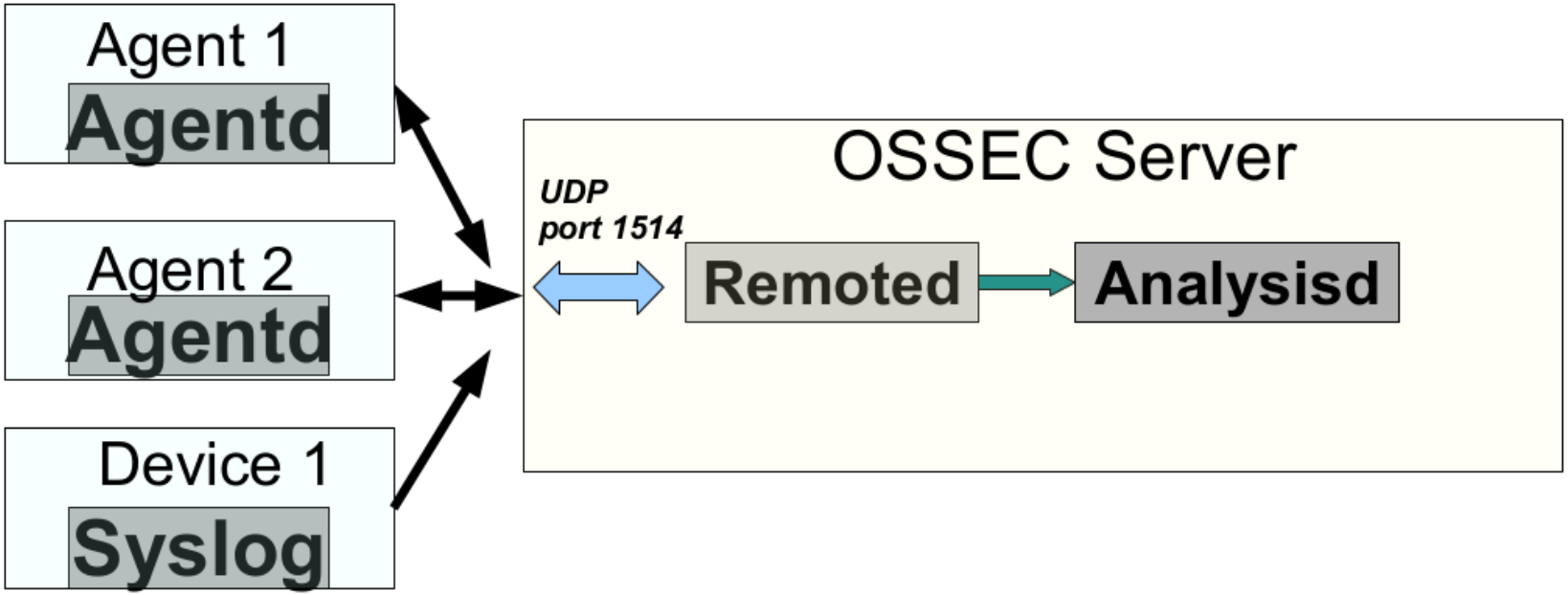
Dlaczego warto używać ?

- Porządnie analizuje logi - predecoding,decoding.
- **Prosty** w instalacji.
- Łatwy w konfiguracji (dla programistów - XML :/).
- Windows, Solaris, Linux.
- Z założenia bezpieczny - instalator + chroot, limitowane zadania dla komponentów.
- Przychodzi z ogromną ilością regexpów, dekoderów, reguł.

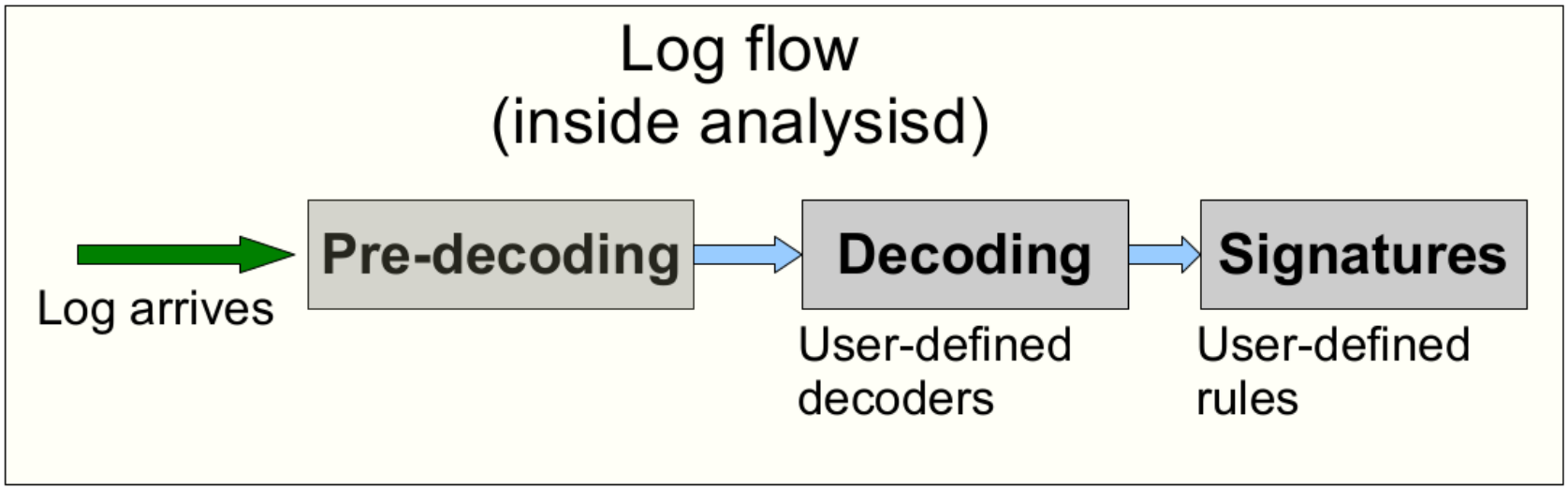
Jak to działa ?



Rysunek: Architektura OSSEC - logiczna

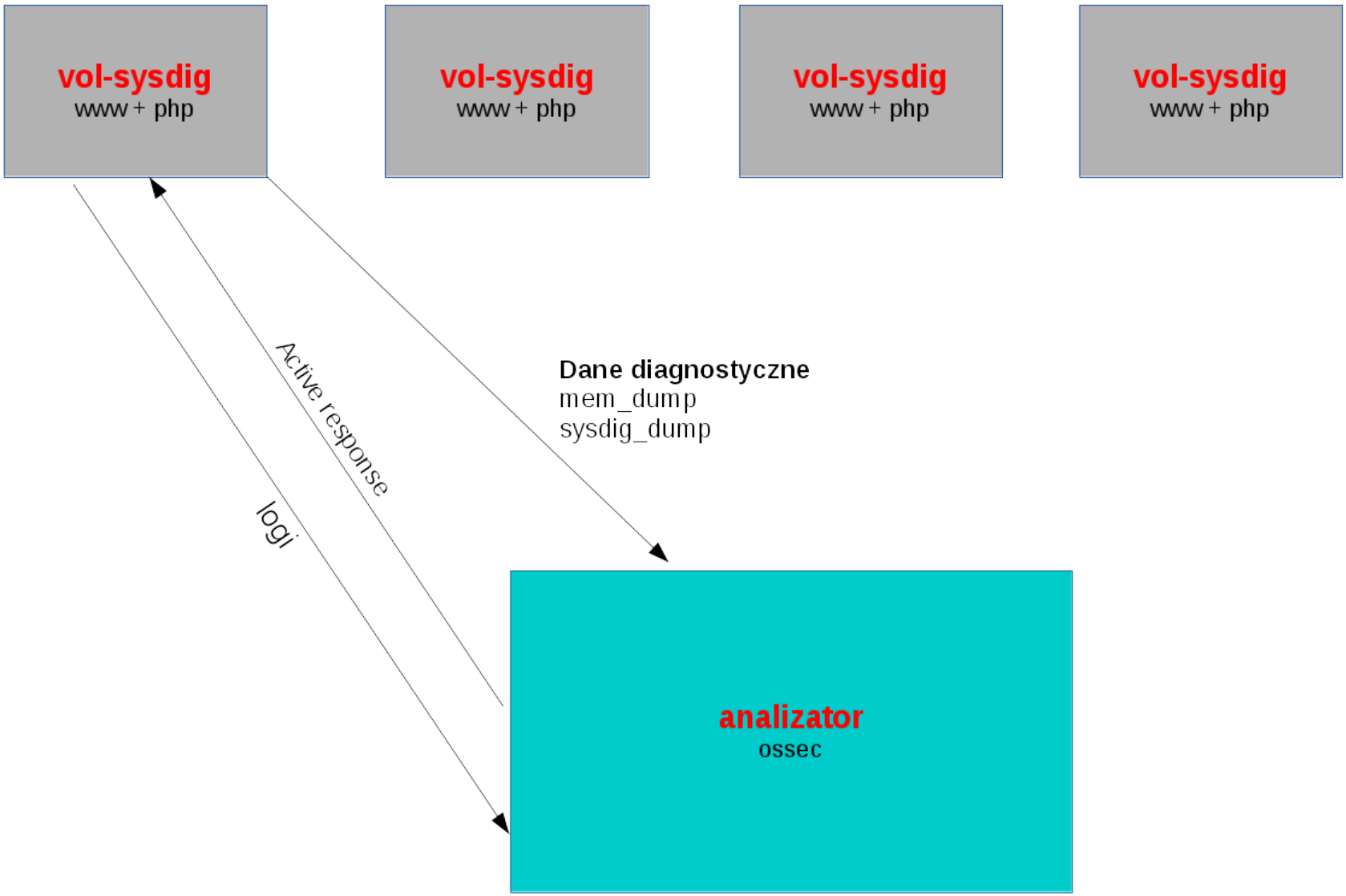


Rysunek: Architektura OSSEC - sieciowa



Rysunek: Architektura OSSEC - wewnętrzna

DEMO!


vol-sysdig
www + php
vol-sysdig
www + php
vol-sysdig
www + php
vol-sysdig
www + php
Active response
logi
Dane diagnostyczne
mem_dump
sysdig_dump
analizator
ossec

DZIĘKUJĘ

Pytania ?

mail: pyton@pyton.systems