SECURITY-RELATED MONITORING WITH ZABBIX

Kaspars Mednis
ZABBIX Technical Support Engineer
SECURITY MONITORING – WHY?

Potential issues

- Software vulnerabilities
- Weak configurations
- Unnecessary open ports
- Physical intrusions
WHY ZABBIX?

Zabbix is not a dedicated security monitoring tool.... but you can monitor the following

- configuration files
- log files
- SNMP traps
- and much more......
CHECKSUM MONITORING
SECURITY-RELATED MONITORING
WITH ZABBIX
WHY MONITOR CHECKSUMS?

Because it is the simplest way to detect changes to important files.

- works out of box
- very simple to setup
- efficient
HOW TO MONITOR CHECKSUMS?

Two types of checksums supported:

  ```
  vfs.file.cksum[/etc/passwd] = 1222364044
  ```

- `vfs.file.md5sum[file]` - calculates a 128-bit MD5 hash:
  ```
  vfs.file.md5sum[/etc/passwd] = 7ba56280db8862ebd1c462d3a6b7383f03175
  ```
CONFIG FILES
SECURITY-RELATED MONITORING WITH ZABBIX
Default configuration gives a lot of information
And while it is very useful for deployment and troubleshooting....

It can contain known weaknesses
It can also give very valuable information to potential hackers !!!
HOW TO MONITOR CONFIGURATION?

You can monitor the content of a configuration file:

`vfs.file.contents[file]` - returns back the content of a file

The most important parts of a configuration file can be monitored using dependent items.
### RESULTS

<table>
<thead>
<tr>
<th>Name</th>
<th>Interval</th>
<th>History</th>
<th>Trend</th>
<th>Type</th>
<th>Last check</th>
<th>Last value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apache (3 Items)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Apache config</td>
<td>60s</td>
<td>0</td>
<td></td>
<td>Zabbix agent</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>vfs.file.contents[/etc/httpd/conf/ht...</code></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Apache server signature</td>
<td>90d</td>
<td></td>
<td></td>
<td>Dependable</td>
<td>2019-10-10 14:39:19</td>
<td>On</td>
</tr>
<tr>
<td><code>apache.server.signature</code></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Apache server tokens</td>
<td>90d</td>
<td></td>
<td></td>
<td>Dependable</td>
<td>2019-10-10 15:14:19</td>
<td>Full</td>
</tr>
<tr>
<td><code>apache.server.tokens</code></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Severity</th>
<th>Recovery time</th>
<th>Status</th>
<th>Info</th>
<th>Host</th>
<th>Problem</th>
</tr>
</thead>
<tbody>
<tr>
<td>High</td>
<td>PROBLEM</td>
<td></td>
<td></td>
<td>Test</td>
<td>Apache server tokens is not &quot;prod&quot;</td>
</tr>
</tbody>
</table>

**Status:**
- **PROBLEM**
- **Full**

**Host:** Test

**Problem:** Apache server tokens is not "prod"
VULNERABILITY SCANS
SECURITY-RELATED MONITORING
WITH ZABBIX
WHAT IF YOU ARE NOT A SECURITY EXPERT?

- External programs can be used to check vulnerabilities
- Output can be parsed, and useful information extracted
- Triggers can be created to send out alerts
HOW IT WORKS

External check → Report item → Dependent Items → Triggers → Alarm
Example of a security report

```
[root@home]# nikto -o report.xml -C all -Tuning 9 -h http://127.0.0.1/
  ***** RFIURL is not defined in nikto.conf--no RFI tests will run ******
  Nikto v2.1.6
+ Target IP: 127.0.0.1
+ Target Hostname: 127.0.0.1
+ Target Port: 80
+ Start Time: 2019-10-02 18:45:26 (GMT3)
+ 1 host(s) tested
```
HOW TO EXTRACT INFORMATION?

Use Zabbix built – in preprocessing

- Regular expressions
- JSON PATH
- XML PATH
- CSV to JSON
- JavaScript
## LLD Preprocessing Possibilities

<table>
<thead>
<tr>
<th>Preprocessing steps</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Regular expression</td>
</tr>
<tr>
<td></td>
<td>Text</td>
</tr>
<tr>
<td></td>
<td>Structured data</td>
</tr>
<tr>
<td></td>
<td>Custom scripts</td>
</tr>
<tr>
<td></td>
<td>Validation</td>
</tr>
<tr>
<td></td>
<td>Throttling</td>
</tr>
<tr>
<td></td>
<td>Prometheus</td>
</tr>
</tbody>
</table>

- **Text**: Regular expression
- **Structured data**: XML XPath, JSONPath, CSV to JSON
- **Custom scripts**: JavaScript
- **Validation**: Does not match regular expression, Check for error in JSON, Check for error in XML
- **Throttling**: Discard unchanged with heartbeat
- **Prometheus**: Prometheus to JSON
## DEPENDENT ITEMS EXAMPLE

### Preprocessing

<table>
<thead>
<tr>
<th>Name</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regular expression</td>
<td><code>(\d+\?= error)</code></td>
</tr>
</tbody>
</table>

### Preprocessing steps

<table>
<thead>
<tr>
<th>Item</th>
<th>Preprocessing steps</th>
<th>Name</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>Regular expression</td>
</tr>
</tbody>
</table>

### Hosts

<table>
<thead>
<tr>
<th>Host</th>
<th>Name</th>
<th>Interval</th>
<th>History</th>
<th>Trends</th>
<th>Type</th>
<th>Last check</th>
<th>Last value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Test</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Report (4 items)</td>
<td>90d</td>
<td>365d</td>
<td></td>
<td>Dependent item</td>
<td>2019-10-02 19:39:25</td>
<td>0 errors</td>
</tr>
<tr>
<td></td>
<td>Scan errors</td>
<td>90d</td>
<td>365d</td>
<td></td>
<td>Dependent item</td>
<td>2019-10-02 19:39:25</td>
<td>0 errors</td>
</tr>
<tr>
<td></td>
<td>Scan reports</td>
<td>90d</td>
<td>365d</td>
<td></td>
<td>Dependent item</td>
<td>2019-10-02 19:39:25</td>
<td>5 reports</td>
</tr>
<tr>
<td></td>
<td>Scan requests</td>
<td>90d</td>
<td>365d</td>
<td></td>
<td>Dependent item</td>
<td>2019-10-02 19:39:25</td>
<td>2121 requests</td>
</tr>
<tr>
<td></td>
<td>Security report</td>
<td>30s</td>
<td>90d</td>
<td></td>
<td>External check</td>
<td>2019-10-02 19:39:25</td>
<td>- ***** RFIURL is not define...</td>
</tr>
</tbody>
</table>

*Scan.sh[/http://127.0.0.1/zabbix]*
ADVANCED VULNERABILITY SCANS
SECURITY-RELATED MONITORING WITH ZABBIX
CAN YOU DO SOMETHING WITH THIS REPORT?
Any JSON format data can be processed by LLD
LLD ITEM PROTOTYPES

<table>
<thead>
<tr>
<th>Item prototype</th>
<th>Preprocessing</th>
</tr>
</thead>
<tbody>
<tr>
<td>* Name</td>
<td>Vulnerability [#ITEMID]</td>
</tr>
<tr>
<td>Type</td>
<td>Dependent item</td>
</tr>
<tr>
<td>* Key</td>
<td>vulnerability[#ITEMID]</td>
</tr>
<tr>
<td>* Master item</td>
<td>Test: Vulnerability data</td>
</tr>
<tr>
<td>Type of information</td>
<td>Character</td>
</tr>
</tbody>
</table>

Preprocessing steps:

1. JSONPath
   - $n$ike$scan$.scdetails item [#ITEMID] description

Actions:
- Update
- Clone
- Delete
- Cancel
## ITEMS CREATED FROM THE REPORT

<table>
<thead>
<tr>
<th>Wizard</th>
<th>Name</th>
<th>Triggers</th>
<th>Key</th>
<th>Interval</th>
<th>History</th>
<th>Trends</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Vulnerability LLD: Vulnerability data: Vulnerability 999967</td>
<td>Triggers</td>
<td>vulnerability[999967]</td>
<td>90d</td>
<td></td>
<td></td>
<td>Dependent item</td>
</tr>
<tr>
<td></td>
<td>Vulnerability LLD: Vulnerability data: Vulnerability 999971</td>
<td>Triggers</td>
<td>vulnerability[999971]</td>
<td>90d</td>
<td></td>
<td></td>
<td>Dependent item</td>
</tr>
<tr>
<td></td>
<td>Vulnerability LLD: Vulnerability data: Vulnerability 999984</td>
<td>Triggers</td>
<td>vulnerability[999984]</td>
<td>90d</td>
<td></td>
<td></td>
<td>Dependent item</td>
</tr>
<tr>
<td></td>
<td>Vulnerability LLD: Vulnerability data: Vulnerability 999985</td>
<td>Triggers</td>
<td>vulnerability[999985]</td>
<td>90d</td>
<td></td>
<td></td>
<td>Dependent item</td>
</tr>
<tr>
<td></td>
<td>Vulnerability LLD: Vulnerability data: Vulnerability 999990</td>
<td>Triggers</td>
<td>vulnerability[999990]</td>
<td>90d</td>
<td></td>
<td></td>
<td>Dependent item</td>
</tr>
<tr>
<td></td>
<td>Vulnerability data</td>
<td></td>
<td>report.pid(data)</td>
<td>1h</td>
<td>0</td>
<td></td>
<td>External check</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Time</th>
<th>Severity</th>
<th>Info</th>
<th>Host</th>
<th>Problem</th>
</tr>
</thead>
<tbody>
<tr>
<td>2019-10-02 19:02:20</td>
<td>Warning</td>
<td>Test</td>
<td>Allowed HTTP Methods: GET, HEAD, POST, OPTIONS, TRACE</td>
<td></td>
</tr>
<tr>
<td>2019-10-02 19:02:20</td>
<td>Warning</td>
<td>Test</td>
<td>Retrieved x-powered-by header: PHP/5.4.16</td>
<td></td>
</tr>
<tr>
<td>2019-10-02 19:02:20</td>
<td>Warning</td>
<td>Test</td>
<td>Web Server returns a valid response with junk HTTP methods, this may cause false positives.</td>
<td></td>
</tr>
<tr>
<td>2019-10-02 19:02:20</td>
<td>Warning</td>
<td>Test</td>
<td>Server leaks inodes via ETags. header found with file /zabbix/robots.txt. fields: 0x3ce 0x593eb000cf4c0</td>
<td></td>
</tr>
<tr>
<td>2019-10-02 19:02:20</td>
<td>Warning</td>
<td>Test</td>
<td>HTTP TRACE method is active, suggesting the host is vulnerable to XST</td>
<td></td>
</tr>
</tbody>
</table>
WHAT IF THE SCRIPT TAKES TOO LONG TO EXECUTE?

Maximum execution time is 30s ..... 00:30

In this case, cron jobs or other scheduling mechanisms can be used.
SERVICES MONITORING
SECURITY-RELATED MONITORING
WITH ZABBIX
CAN WE MONITOR SERVICES OUT OF BOX?

Yes – using new Zabbix agent:

Two new item keys supported

**systemd.unit.discovery[<type>]** List of systemd units and their details.
  - type - *all*, *automount*, *device*, *mount*, *path*, *service* (default), *socket*, *swap*, *target*

**systemd.unit.info[<unit name>,<property>,<interface>]** Systemd unit information
  - unit name - unit name
  - property - unit property (e.g. ActiveState (default), LoadState, Description)
  - interface - unit interface type (e.g. Unit (default), Socket, Service)
SERVICES MONITORING EXAMPLE

**Item** | **Preprocessing**
---|---
* Name | Systemd firewall status
Type | Zabbix agent
* Key | systemd.unit.info[firewalld.service]
* Host interface | 127.0.0.1:10050
Type of information | Character
* Update interval | 1m

**Name** | **Interval** | **History** | **Trends** | **Type** | **Last check** | **Last value**
---|---|---|---|---|---|---
- other - (1 Item)
Systemd firewall status | 1m | 90d | | Zabbix agent | 2019-10-09 15:42:30 | inactive
systemd.unit.info[firewalld.service]
PORT MONITORING
SECURITY-RELATED MONITORING WITH ZABBIX
CAN WE MONITOR OPEN PORTS?

Yes, of course!

Zabbix can do it out of box

- check open ports using `net.tcp.port[]` simple check
- use discovery to scan your entire network for open ports
WHY WE NEED TO MONITOR OPEN PORTS?

Why do you need this?

- Applications with weak security (telnet, ftp)
- Unneeded applications with known vulnerabilities
- Less open ports – more secure system
SIMPLE NETWORK DISCOVERY RULE

Name: Local network
IP range: 192.168.0.1-254
Update interval: 1h
Checks:
- Zabbix agent "system uname"
- FTP
- Telnet
- SNMPv2 agent "1.3.6.1.2.1.5.0"

Check type: SNMPv2 agent
Port range: 161
SNMP community: public
SNMP OID: 1.3.6.1.2.1.5.0
UNSECURE WEB PAGES
SECURITY-RELATED MONITORING
WITH ZABBIX
HOW CAN WE FIND HTTP ENABLEDPages?

HTTPS is the recommended web protocol today
Open HTTP port does not mean the page is not redirected to HTTPS

How to check it?

- Use Zabbix built in web scenarios
- check the response code

The HTTP response status code 301 Moved Permanently is used for permanent URL redirection
WEB SCENARIO EXAMPLE

Step of web scenario

* Name: Zabbix HTTP page
* URL: http://www.zabbix.com

Query fields:

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Remove</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>value</td>
<td></td>
</tr>
</tbody>
</table>

Follow redirects: [ ]

Retrieve mode:
- Body
- Headers
- Body and headers

* Timeout: 15s

Required string:

Required status codes: 301
CAN WE FIND UNSECURE HTTPS PAGES?

You can use a webscenario to authenticate the certificate

- verify that SSL certificate of the web server is valid
  trusted by a known certificate authority, not expired etc.

- verify that the Common Name field or the Subject Alternate Name field of the web server certificate matches the server name.
HTTPS CERTIFICATE VALIDATION

Scenario   Steps   Authentication

HTTP authentication  None

SSL verify peer  ✔

SSL verify host  ✔

SSL certificate file  my_secure_certificate.cer

SSL key file  ssl_key

SSL key password  supersecurepassword

Update   Clone   Clear history and trends   Delete   Cancel
EXPIRED CERTIFICATES
SECURITY-RELATED MONITORING
WITH ZABBIX
CAN WE FIND EXPIRED CERTIFICATES?

Community made externals cripts can be used to warn you about your certificate expiration.

You can monitor (for example):
- Time until expiration (if valid)
- Expired days ago (if expired)
SNMP TRAPS
SECURITY-RELATED MONITORING WITH ZABBIX
CAN WE MONITOR SNMP TRAPS?

Yes, using Zabbix SNMP trapper item

What to monitor?

- Administrative logins
- Ports status up/down
- New devices (MAC security)
- Thresholds reached (Network attacks)
- Any other security related checks
HOW SNMP TRAPS WORK?
SENSOR MONITORING

SECURITY-RELATED MONITORING
WITH ZABBIX
MONITOR YOUR ENVIRONMENT WITH ZABBIX

- Temperature sensors
- Smoke sensors
- Humidity sensors
- Door sensors
- Motion detection sensors
WHY DO YOU NEED TO MONITOR LOGS?

A lot of security related information can be found in log files.

For example:
- Unsuccessful logins
- Successful logins!
- Elevation of privileges
LOG FILE MONITORING

Log files can be parsed to find important information
Dependent items can be created from log items
Triggers can be created to alert about serious security issues
Information from log files can be extracted and used in trigger names and tags
MASTER LOG ITEM

Master item contains all important log information

- **Name**: secure log
- **Type**: Zabbix agent (active)
- **Key**: log["/var/log/secure"]
- **Type of Information**: Log
- **Update Interval**: 1s
- **History storage period**: Do not keep history, Storage period: 90d
- **Log time format**: None
- **New application**: security

Applications: None
**DEPENDENT LOG ITEMS**

Dependent items extract information from the main log.

<table>
<thead>
<tr>
<th>Item</th>
<th>Preprocessing</th>
</tr>
</thead>
<tbody>
<tr>
<td>* Name</td>
<td>sudo failed</td>
</tr>
<tr>
<td>Type</td>
<td>Dependent item</td>
</tr>
<tr>
<td>* Key</td>
<td>sudo.fail</td>
</tr>
<tr>
<td>* Master item</td>
<td>Zabbix42: secure log</td>
</tr>
<tr>
<td>Type of information</td>
<td>Log</td>
</tr>
<tr>
<td>* History storage period</td>
<td>Do not keep history</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Item</th>
<th>Preprocessing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preprocessing steps</td>
<td>Name</td>
</tr>
<tr>
<td>1:</td>
<td>Regular expression</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Add</td>
<td></td>
</tr>
</tbody>
</table>

Update | Clone | Check now | Clear history and trends | Delete | Cancel
## DEPENDENT LOG ITEMS

<table>
<thead>
<tr>
<th>Wizard</th>
<th>Name</th>
<th>Triggers</th>
<th>Key</th>
<th>Interval</th>
<th>History</th>
<th>Trends</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>secure log: login failed</td>
<td>Triggers 2</td>
<td>root.login.failed</td>
<td>90d</td>
<td></td>
<td></td>
<td>Dependent item</td>
</tr>
<tr>
<td></td>
<td>secure log: login succeeded</td>
<td></td>
<td>root.login.success</td>
<td>90d</td>
<td></td>
<td></td>
<td>Dependent item</td>
</tr>
<tr>
<td></td>
<td>secure log</td>
<td></td>
<td>log(&quot;/var/log/secure&quot;)</td>
<td>1s</td>
<td>90d</td>
<td></td>
<td>Zabbix agent (active)</td>
</tr>
<tr>
<td></td>
<td>secure log: sudo failed</td>
<td>Triggers 1</td>
<td>sudo.fail</td>
<td>90d</td>
<td></td>
<td></td>
<td>Dependent item</td>
</tr>
</tbody>
</table>

### Host: Zabbix42

- **- other - (4 items)**
  - **login failed**
    - root.login.failed
    - Interval: 90d
    - History: 90d
    - Trends: 90d
    - Type: Dependent item
    - Last check: 2019-10-03 08:20:23
    - Last value: Failed password for kasper...
  - **login succeeded**
    - root.login.success
    - Interval: 90d
    - History: 90d
    - Trends: 90d
    - Type: Dependent item
    - Last check: 2019-10-03 08:20:55
    - Last value: Accepted password for kasper...
  - **secure log**
    - log("/var/log/secure")
    - Interval: 1s
    - History: 90d
    - Trends: 90d
    - Type: Zabbix agent (active)
    - Last check: 2019-10-03 08:43:10
    - Last value: Oct 3 08:43:09 zabbix42 su...
  - **sudo failed**
    - sudo.fail
    - Interval: 90d
    - History: 90d
    - Trends: 90d
    - Type: Dependent item
    - Last check: 2019-10-03 08:43:10
    - Last value: Oct 3 08:43:09 zabbix42 su...
GATHERING USEFUL INFORMATION

Information can be extracted from the logs using function `regsub (<pattern>,<output>)`

Extracted information can be used in

- Trigger names
- Trigger tags
LOG TRIGGERS

Log line:

```
sudo: kaspars : user NOT in sudoers ; TTY=pts/3 ; PWD=/home/kaspars ; USER=zabbix ; COMMAND=/bin/ping
```

Examples to extract user and executed command

```
{{ITEM.VALUE}.regsub("sudo: (.+) ", user: \1)}
{{ITEM.VALUE}.regsub("COMMAND=(.+)", command: \1)}
```
## LOG BASED TRIGGER EXAMPLE

<table>
<thead>
<tr>
<th>Trigger</th>
<th>Tags</th>
<th>Dependencies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Severity</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Expression</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Expression

`Sudo attempt without permissions {{ITEM.VALUE}.regsub("sudo: (.+): user:\".\"\", user:\"1\")}}`  

### Expression Constructor

- OK event generation: Expression
- PROBLEM event generation mode: Single
- Allow manual close: Yes
- URL: 

---

ZABBIX '19
USE TAGS TO FILTER INFORMATION!

<table>
<thead>
<tr>
<th>Time</th>
<th>Severity</th>
<th>Recovery time</th>
<th>Status</th>
<th>Info</th>
<th>Host</th>
<th>Problem</th>
<th>Duration</th>
<th>Ack</th>
<th>Actions</th>
<th>Tags</th>
</tr>
</thead>
<tbody>
<tr>
<td>14:00:46</td>
<td>High</td>
<td></td>
<td></td>
<td></td>
<td>Zabbix42</td>
<td>sudo attempt by : user: kaspers command: /bin/ping</td>
<td>23s</td>
<td>No</td>
<td></td>
<td>USER: kaspers</td>
</tr>
</tbody>
</table>

14:00
CAN WE MONITOR WINDOWS LOGS?

Yes, a special key `eventlog` can be used.

You can filter event logs by:

- **Source** (Security, System etc...)
- **Severity** ("Warning", "Error", "Critical" etc...)
- **Eventid**
  
  - 4625 – Logon Failure
  - 4740 - A user account was locked out)
ZABBIX INTEGRATIONS
SECURITY-RELATED MONITORING
WITH ZABBIX
ZABBIX INTEGRATION

Security monitoring
Antivirus
Barracuda
Check Point
F5 Networks
Fortinet
Kaspersky
SELinux
HTTPS
Blue Coat
Firewalls
OpenVPN
THANK YOU!

Kaspars Mednis
ZABBIX Technical Support Engineer