

IBM Security Verify Access monitoring and alerting with Zabbix

IBM Security Expert Labs

Version 0.1.0, 10-12-2021: First draft

Table of Contents

Description	1
Contents	1
How to use Zabbix for monitoring IBM Security Verify Access appliances	1
1. Monitoring and alerting in the ISVA Appliances	1
1.1. Rsyslog	2
1.2. SNMP Alerts	2
1.3. SNMP Polling	3
1.4. Federation runtime monitoring	4
1.5. Rest API monitoring	5
1.6. Rest API Statistics	6
1.7. Front end monitoring	6
1.8. Overlapping data	7
1.8.1. More information	7
2. Zabbix installation	7
2.1. Requirements	7
2.2. Zabbix Container installation	7
2.3. Create a pod	8
2.4. Create a mysql database container.	8
2.5. Zabbix snmp trapper	8
2.5.1. Create the zabbix trapper.	8
2.6. Zabbix server	9
2.7. Zabbix web interface	9
3. Zabbix installation from source or package	10
4. Zabbix configuration with Ansible	10
4.1. Get playbooks	10
4.2. Configure Python	11
4.3. Ansible example files	11
4.4. Galaxy	12
4.5. Zabbix playbooks	12
4.5.1. Create ISVA templates in Zabbix	12
4.5.2. Create ISVA hosts in Zabbix	14
5. ISVA configuration with Ansible	15
5.1. Configure Python	15
5.2. Ansible example files	15
5.3. Galaxy	15
5.4. ISVA playbooks	16

5.4.1. Prepare the inventory	16
5.4.2. Configure system in ISVA	16
5.4.3. Configure monitoring in ISVA	16
5.4.4. Configure snmp alert in ISVA	16
5.4.5. Create a test reverse proxy	17
6. Zabbix Traps handling	18
6.1. Zabbix logs	18
6.2. Make sure the zabbix server is enabled to handle snmp traps	18
6.3. Traps on the zabbix trap container	18
6.4. Unmatched traps	19
6.5. Send test events	19
6.6. Polling SNMP	20
6.7. MIB file	20
7. MIB MANAGER (Netcool)	20
7.1. Download MIB manager	20
7.2. Install MIB Manager	20
7.3. Start the tool	21
7.4. Import the MIB	21
7.4.1. Export (generate traps)	21
8. Resources	22

Description

IBM Security Verify Access monitoring and alerting with Zabbix

How to use the opensource Zabbix monitoring tool for monitoring IBM Security Verify Access appliances. This document includes instructions to install Zabbix on your own machine, and Ansible playbooks to configure both Zabbix and an IBM ISVA Appliance.

Contents

How to use Zabbix for monitoring IBM Security Verify Access appliances

At multiple customer engagements, the same question always pops up:

How is everybody else monitoring ISVA/ISAM, and how should we do that exactly ?

– literally all, isva customers

This question seems simple enough to answer at first glance, because there is a lot of information out there. However, if you dig deeper, it turns out there is no practical example available and the information lacks in detail, or there's simply too much information so it's difficult to start.

This how-to guide describes how to:

- Set up Zabbix as a monitoring solution for ISAM.
- Configure it to perform basic monitoring of the main parts of ISAM.

The goal is to provide a complete (or at least, reasonably complete) solution for monitoring ISAM, including examples that you can use directly in your own environment.

Chapter 1. Monitoring and alerting in the ISVA Appliances

This document focuses on monitoring and alerting for the ISVA appliances (hardware or virtual). Some of the information here is relevant as well for the Docker version of ISVA, but a lot of it is not. In the container version of ISAM, you can use the container native methods of accessing log files and you can monitor systems by monitoring the health of the container.

The best complete overview is the OpenMic presentation given by IBM Support in 2019 ([Appliance monitoring](#)).

The Openmic document gives a complete overview of the options, and is worth while reviewing.

However, what I want to do here is give a practical implementation , and focus on the things that I deem more useful. These are also the elements that are built in the Zabbix setup (except for the rsyslog forwarding).

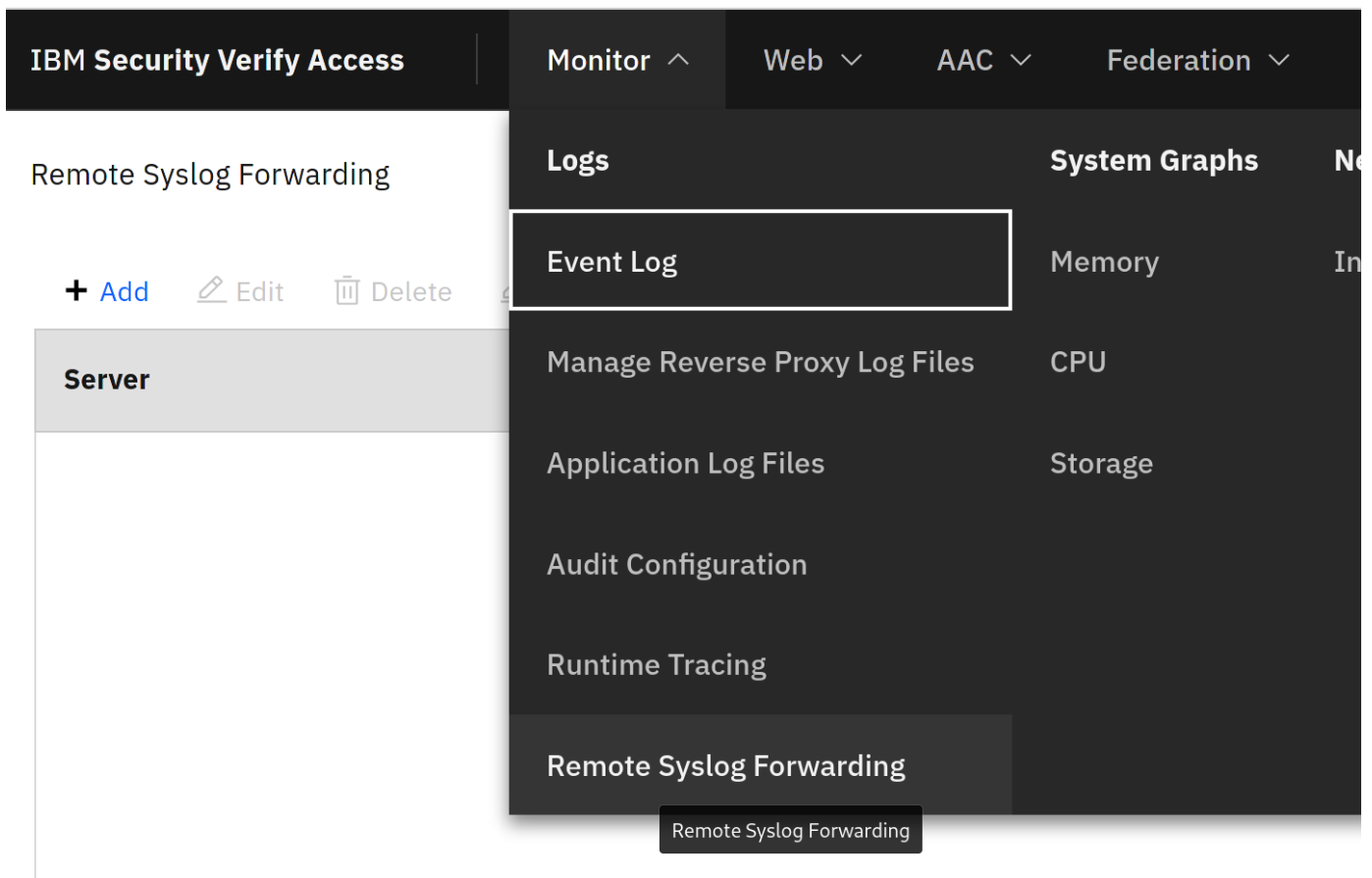
- rsyslog log forwarding (not covered here)
- snmp alerts
- snmp polling (agentless monitoring)
- federation runtime monitoring endpoint

- rest api monitoring
- rest api statistics
- front end monitoring (not included in the demo)

1.1. Rsyslog

You should forward your log files to a rsyslogd system. Typically, this is a SIEM tool like Splunk or IBM QRadar, or an ELK stack. Standard Linux rsyslogd would also work fine.

The preferred configuration is to use the Remote Syslog Forwarding configuration, instead of configuring each logfile separately for rsyslog. ISVA should be configured to store the logfiles locally, and the rsyslog forwarder picks them up and forwards them. This allows you to survive (longer) outages of your rsyslogd infrastructure.



Zabbix does not include an rsyslogd.

1.2. SNMP Alerts

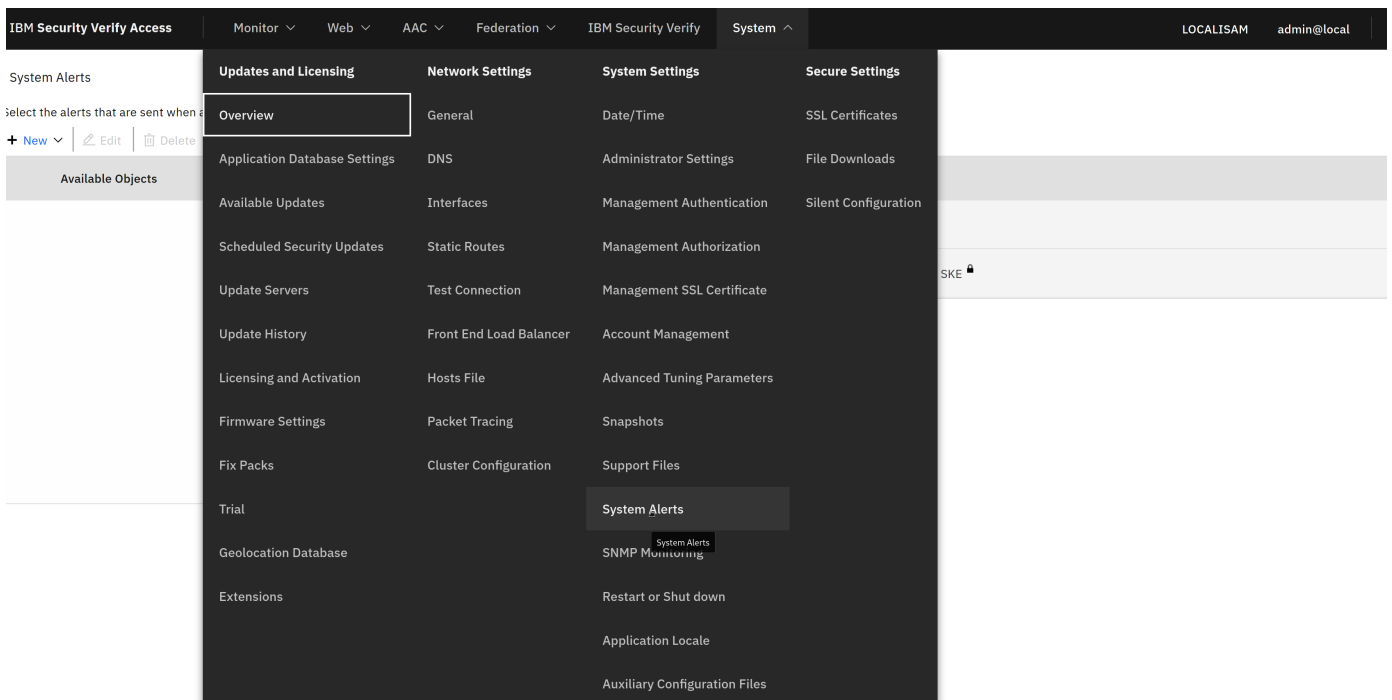
SNMP Alerts are sent by the ISVA appliance to an snmp trapper host. Typically in enterprise environments these are systems that are more geared towards checking the base health and availability of systems. Popular tools here are HP Openview, IBM Netcool or Nagios.

The snmp traps sent by ISVA are pretty crude , and there is not a whole lot that can be configured. There is a set of thresholds that can be configured, and it's possible to filter out a events from being sent (in recent versions of the product only, 10.0+).

Still, this is a very powerful way of having an immediate warning that something is wrong . As soon as the event happens, the trap is sent out in realtime.

In the demo, I have configured a trap trigger in Zabbix to show a warning when a reverse proxy is stopped.

In ISAM, the configuration is done under System/System Alerts.



There you can create an SNMP alert. The important options here are the level of snmp (v2c or 3), the snmp host that will process your traps and the snmp community.

Most trapper systems rely on a MIB file to tell them what to expect. The MIB file that is currently provided by IBM, does not do this. I've updated the ISAM MIB file (based on a more generic IBM Appliance mib file) to play a little nicer.

[modified isam mib](#)

Refer to this section [MIB MANAGER](#) for more information

1.3. SNMP Polling

SNMP Polling is the inverse of the snmp alerts. This allows external systems to poll ISVA for snmp data. There is a huge amount of data exposed in the snmp polling.

Most of the data is similar or equal to any Linux system, and there are some additions specific for ISVA.

There is quite a bit of information on this in the IBM Support repositories on Github.

[IBM Support on Github](#)

But the challenge is to get the data out in a meaningful way. In general, the Linux OS data obtained from the snmp polling is very useful, because monitoring tools can handle that automatically. This means that the following statistics are relevant:

- CPU
- Memory
- Storage

- Ethernet / Network
- general system availability (this means that snmp polling fails, or icmp ping fails)

That is a different story when you start to look at ways to get more detailed information. For instance, the individual resource usage of each webseald process is exposed in the SNMP data.

In theory , you can monitor the data, but in reality it is pretty hard to actually parse out the information, and this may not be worth the effort !

1.4. Federation runtime monitoring

The federation runtime can be enabled for monitoring.

In version 10 and up, you also have the possibility to enable a Prometheus-style monitoring endpoint (/metrics), but Zabbix doesn't accommodate for that so I'm using the older /monitoring endpoint here.

The federation endpoint can be accessed through the browser as well, but you do need to configure it:

- enable the monitoring endpoint (as mentioned already) in System / Advanced Tuning

IBM Security Verify Access | Monitor ▾ | Web ▾ | AAC ▾ | Federation ▾ | IBM Security Verify | System ▾

Advanced Tuning Parameters (Change advanced tuning parameter values only under the supervision of IBM Customer Support.)

+ New | Edit | Delete

<input type="checkbox"/>	Key	Value	Comment
<input type="checkbox"/>	nist.sp800-131a.strict	false	Advanced tuning for the FIPS and NIST SP800-131a configuration. T
<input type="checkbox"/>	wga_rte.embedded.ldap.ssl.port	636	The port on which the embedded LDAP server will listen for LDAPS re
<input type="checkbox"/>	wga_rte.embedded.ldap.include.in.snapshot	false	Include the user registry data from the embedded user registry in the
<input type="checkbox"/>	java.enable.ibmjceplus	false	This parameter will control whether the IBMJCEPlus security provide performance improvements over Java based cryptographic operator
<input type="checkbox"/>	lmt.enabled	true	Boolean value which is used to control whether or not the LMT tool is
<input type="checkbox"/>	password.policy	minlen=8 dcredit=1 ucredit=1 lcredit=1	Enforced PAM password quality (pam_pwquality.so) for management
<input type="checkbox"/>	runtime_profile.enable.monitor	true	enable runtime monitoring

1 - 7 of 7 items | 10 | 25 | 50 | 100 | 200

- configure a listening interface for the runtime (Federation / Runtime tuning parameters)

Runtime Listening Interfaces

[+ Add](#) [Edit](#) [Delete](#)

Interface	Port	SSL
Local Interface	80	False
Local Interface	443	True
1.1 192.168.151.200	80	False
1.1 192.168.151.200	443	True



security

The endpoint is accessible as anonymous user. You should make sure it's not publicly accessible (eg. not accessible via the internet).

You could make a junction to it, but from a monitoring standpoint there's a big disadvantage. You'd be monitoring the junction just as much as the actual JVM Runtime.

broken functionality

In version 10.0.2 (any fixpack level) and 10.0.3.0, the Prometheus monitoring endpoint is broken. Although the /monitor endpoint works, there are performance issues related to this. So at the moment (Dec. 2021), it's advised to disable the Federation runtime monitoring until this problem is fixed. There's no technote yet.

1.5. Rest API monitoring

The REST API that ISAM offers, also includes some monitoring functions. These REST calls are also used by the LMI to get to the data.

To enable this, you should configure a user with restricted rights that you can use for the connections to the endpoint. Assign the role "System monitoring" to the user, for instance to a local registry user as in the example below:

Management Authorization

Management Authorization [Users](#) [Groups](#)

Management Authorization

 Enable Authorization Roles[+ New](#) [Delete](#)

Roles
Full Read
Full Write
Global Administrator
Global Read Only Access
Policy Administration Users
Security Administrator
Security Monitoring
Security Viewer
System Administrator
System Monitoring

Local User Database

Remote LDAP User Registry

Features

[Edit](#)**Username**

monitoring01

1.6. Rest API Statistics

These are what the LMI uses to show the graphs on various statistics. This information is not real time, but aggregated every minute.

I think this is somewhat superfluous and not worth spending any time on. You would have realtime data from snmp polling already anyway.

There are other ways of getting this information anyway (for example by adding response times to the request log).

1.7. Front end monitoring

To do front end monitoring, you'd typically use tools like JMeter , tools that are also used in performance testing.

This basically means that you select some endpoints that users would hit, and test if they are still up and running and responding fast enough.

The nice thing in Zabbix is that it also supports this functionality!

1.8. Overlapping data

There's a lot of data exposed in different ways in ISVA. So instead of implementing everything, you should select what makes sense for you!

In an ideal world, you should combine the information gathered here with the information gathered by your SIEM , into meaningful dashboards and actions for your DevOps or support team.

Examples of overlapping data are:

- http audit logging data and http request logs
- response times in statistics, in request logs and in results of front end monitoring
- reverse proxy health in SNMP Polling, SNMP traps and REST monitoring
- ...

1.8.1. More information

[Appliance monitoring](#)

[Configuring SNMP monitoring](#)

[Configuring system alert](#)

[IBM Support on Github](#)

[ISAM MIB File](#)

Chapter 2. Zabbix installation

2.1. Requirements

These instructions require a working Podman setup on a Linux host, for the Zabbix installation.

For the ISVA appliance, I've got a virtual appliance running on VMWare workstation, with the LMI interface configured as "host-only" network.

2.2. Zabbix Container installation

I'm going to use the container installation here, that is explained on the Zabbix website:

<https://www.zabbix.com/documentation/current/manual/installation/containers>

We need a Zabbix Server (with the mysql database), a Zabbix Web Server and the Zabbix snmp trapper.

We don't need a Zabbix agent or proxy at this point, nor the Java gateway.

I've used Podman here, but you can just as well use Docker, or even Kubernetes.



I use 'sudo' with Podman to avoid all sorts of rights problems related to non-root rights.

2.3. Create a pod

```
sudo podman pod create --name zabbix-pod \  
-p 80:8080 \  
-p 10051:10051 \  
-p 162:1162/udp
```

2.4. Create a mysql database container.

You need to create the persistent directory, obviously you can use any directory you like, I use "/data/var/mysql" on my system.

```
sudo podman run --name zabbix-mysql-server -t \  
-e MYSQL_DATABASE="zabbix" \  
-e MYSQL_USER="zabbix" \  
-e MYSQL_PASSWORD="zabbix_pwd" \  
-e MYSQL_ROOT_PASSWORD="root_pwd" \  
-v /data/var/mysql:/var/lib/mysql:Z \  
--restart=always \  
--pod=zabbix-pod \  
-d docker.io/library/mysql:8.0 \  
--character-set-server=utf8 --collation-server=utf8_bin \  
--default-authentication-plugin=mysql_native_password
```

2.5. Zabbix snmp trapper

This container needs to be started first, because we link the Zabbix server to it in the next step.

The directories for the mib files and the traps need to exist on your system.

```
sudo mkdir -p /data/var/zabbix/mibs  
sudo mkdir -p /data/var/zabbix/snmptraps
```

Copy the isam_new.mib file into the /data/var/zabbix/snmptraps directory, so it's available.

2.5.1. Create the zabbix trapper.

```
sudo podman run --name zabbix-snmptraps \  
-v /data/var/zabbix/snmptraps:/var/lib/zabbix/snmptraps:z \  
-v /data/var/zabbix/mibs:/var/lib/zabbix/mibs \  
--pod=zabbix-pod \  
--rm=True \  
-d docker.io/zabbix/zabbix-snmptraps:5.4-alpine-latest
```

Note that the small 'z' option is required to allow podman to set the correct selinux context...

2.6. Zabbix server

The Zabbix server is the core component of the Zabbix infrastructure.

The Zabbix server shares the mounts with the Zabbix trapper, to receive the snmp traps.

```
sudo podman run --name zabbix-server-mysql -t \  
  -e DB_SERVER_HOST="127.0.0.1" \  
  -e MYSQL_DATABASE="zabbix" \  
  -e MYSQL_USER="zabbix" \  
  -e MYSQL_PASSWORD="zabbix_pwd" \  
  -e MYSQL_ROOT_PASSWORD="root_pwd" \  
  -e ZBX_ENABLE_SNMP_TRAPS=True \  
  -v /data/var/zabbix/snmptraps:/var/lib/zabbix/snmptraps:z \  
  --pod=zabbix-pod \  
  --restart=no \  
  --rm=True \  
  -d docker.io/zabbix/zabbix-server-mysql:5.4-alpine-latest
```

Options:

```
-d to run it in the background, instead of in the console  
--rm=True : deletes the container if it exists
```



You should change the passwords for the MYSQL users!

2.7. Zabbix web interface

Start Zabbix.

```
sudo podman run --name zabbix-web-nginx-mysql -t \  
  -e ZBX_SERVER_HOST="zabbix-server-mysql" \  
  -e ZBX_SERVER_NAME="ISVA Zabbix" \  
  -e DB_SERVER_HOST="127.0.0.1" \  
  -e MYSQL_DATABASE="zabbix" \  
  -e MYSQL_USER="zabbix" \  
  -e MYSQL_PASSWORD="zabbix_pwd" \  
  -e MYSQL_ROOT_PASSWORD="root_pwd" \  
  --pod=zabbix-pod \  
  --restart=no \  
  --rm=True \  
  -d docker.io/zabbix/zabbix-web-nginx-mysql:alpine-5.4-latest
```



Use `localhost` or the container name?

Depending on the Podman/Docker host state, sometimes it works better with `127.0.0.1` as the `ZBX_SERVER_HOST`.

```
-e ZBX_SERVER_HOST="127.0.0.1"
```

Now the Zabbix frontend is ready! The default user name is **Admin**, password **zabbix**.

You can access it on localhost, or on any of the IP addresses assigned to your machine.

<http://localhost>

ZABBIX

Username

Password

Remember me for 30 days

Sign in

or [sign in as guest](#)

Chapter 3. Zabbix installation from source or package

It's obviously also possible to install Zabbix on a (Linux) host directly.

However, this can be a lot more work and I don't recommend this for a Demo or Development setup. The container version is easier to get to work.

<https://www.zabbix.com/documentation/current/en/manual/installation/install>

Chapter 4. Zabbix configuration with Ansible

4.1. Get playbooks

```
git clone https://github.com/tombosmansibm/isva-zabbix-demo
```

4.2. Configure Python

Create a virtual environment. This is optional, but highly recommended.

```
python3 -m virtualenv ~/venvzabbix  
source ~/venvzabbix/bin/activate
```

Install the zabbix pip package and ansible

```
pip install zabbix-api  
pip install ansible  
pip install selinux
```

selinux is necessary so that ansible can write files (eg. using copy) to the local system.

4.3. Ansible example files

You can find an Ansible example configuration here :

[ansible/zabbix_playbooks/](#)

These are also available in a separate git repository, for convenience:

<https://github.com/tombosmansibm/isva-zabbix-demo>

The playbook will use an inventory that you've prepared for deployment of ISAM and deploy that to a Zabbix system.

So the inventory is NOT an inventory of Zabbix hosts, but rather an inventory of ISAM hosts that you want to deploy.

The inventory does assume some variables to be present!

An example inventory is included [ansible/inventory](#) or in the github link.

Important variables are:

- `primary_lmi_interface.address` : the LMI interface
- `runtime_interface.address` : the IP address that we'll assign to the Federation Liberty Runtime
- `monitoring_user_lmi_full` : the full name of the monitoring user (eg. admin@local)
- `monitoring_password` : the password of the monitoring user
- `snmp_monitoring_community` : the SNMP v2c community string (the default is 'public')

Optional variables:

- `runtime_protocol` : http or https

- `runtime_port` : the port the runtime server listens on (typically 443)

4.4. Galaxy

```
ansible-galaxy collection install -r collections/requirements.yml
```

The playbooks use the ISAM inventory that is provided here (it's configured in the `ansible.cfg` in each playbook directory)

4.5. Zabbix playbooks

Go to the directory of the `zabbix_playbooks` and execute the playbooks.

4.5.1. Create ISVA templates in Zabbix

```
cd zabbix_playbooks
ansible-playbook create_zabbix_templates.yml
```

```

(venvzabbix) [tbosmans@tbosmans-p73 zabbix_playbooks]$ ansible-playbook
create_zabbix_templates.yml
[DEPRECATION WARNING]: [defaults]callback_whitelist option, normalizing names to
new standard, use
callbacks_enabled instead. This feature will be removed from ansible-core in
version 2.15. Deprecation
warnings can be disabled by setting deprecation_warnings=False in ansible.cfg.
[WARNING]: Collection ansible.posix does not support Ansible version 2.12.0

PLAY [all]
*****
*****

TASK [load vars]
*****
*****
vrijdag 03 december 2021  14:24:41 +0100 (0:00:00.017)          0:00:00.017 *****
ok: [192.168.155.128]

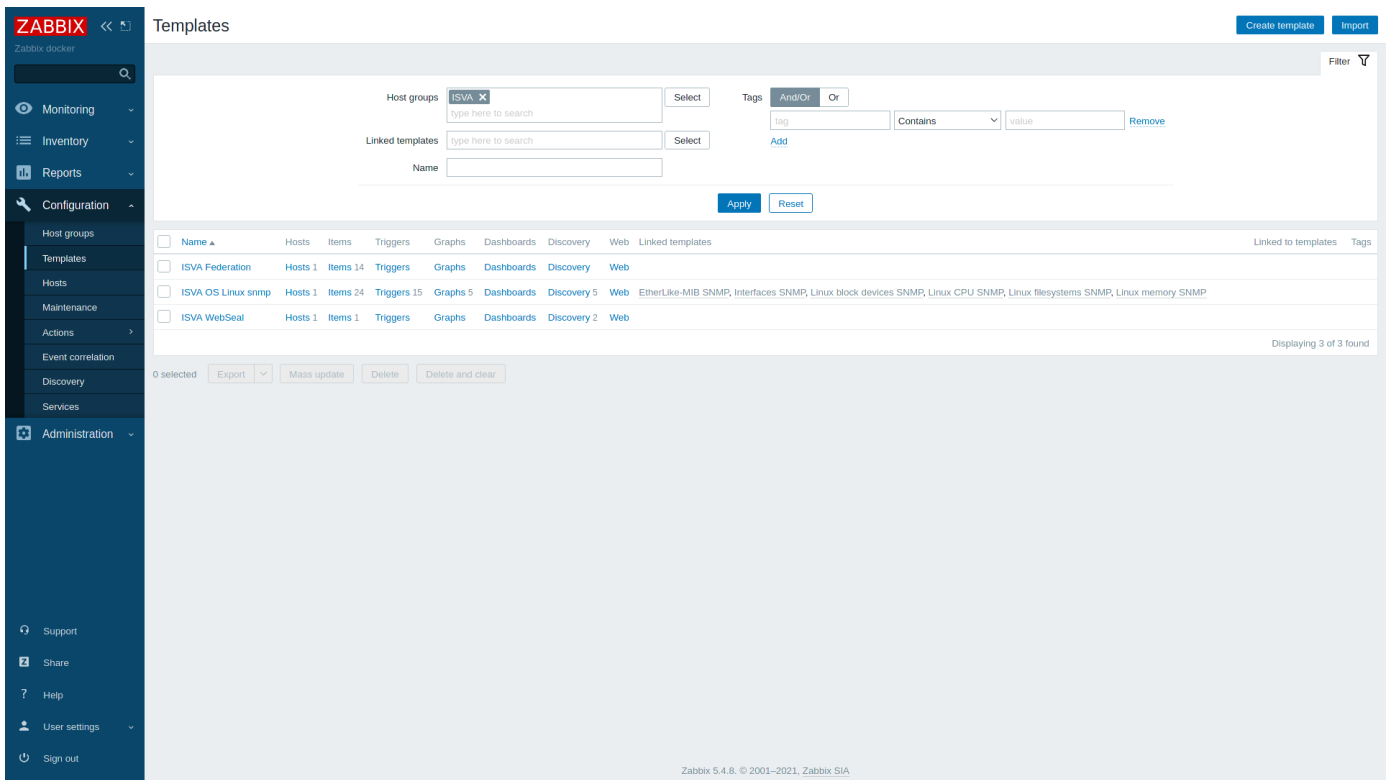
TASK [Import template from xml]
*****
vrijdag 03 december 2021  14:24:41 +0100 (0:00:00.014)          0:00:00.031 *****
changed: [192.168.155.128 -> localhost] => (item
=zabbix_templates/template_isva_federation.xml)
changed: [192.168.155.128 -> localhost] => (item
=zabbix_templates/template_isva_webseal.xml)
changed: [192.168.155.128 -> localhost] => (item
=zabbix_templates/template_isva_os.xml)

PLAY RECAP
*****
*****
192.168.155.128          : ok=2    changed=1    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0

vrijdag 03 december 2021  14:24:43 +0100 (0:00:02.328)          0:00:02.360 *****
=====
Import template from xml
----- 2.33s
load vars
-----
----- 0.01s
Playbook run took 0 days, 0 hours, 0 minutes, 2 seconds

```

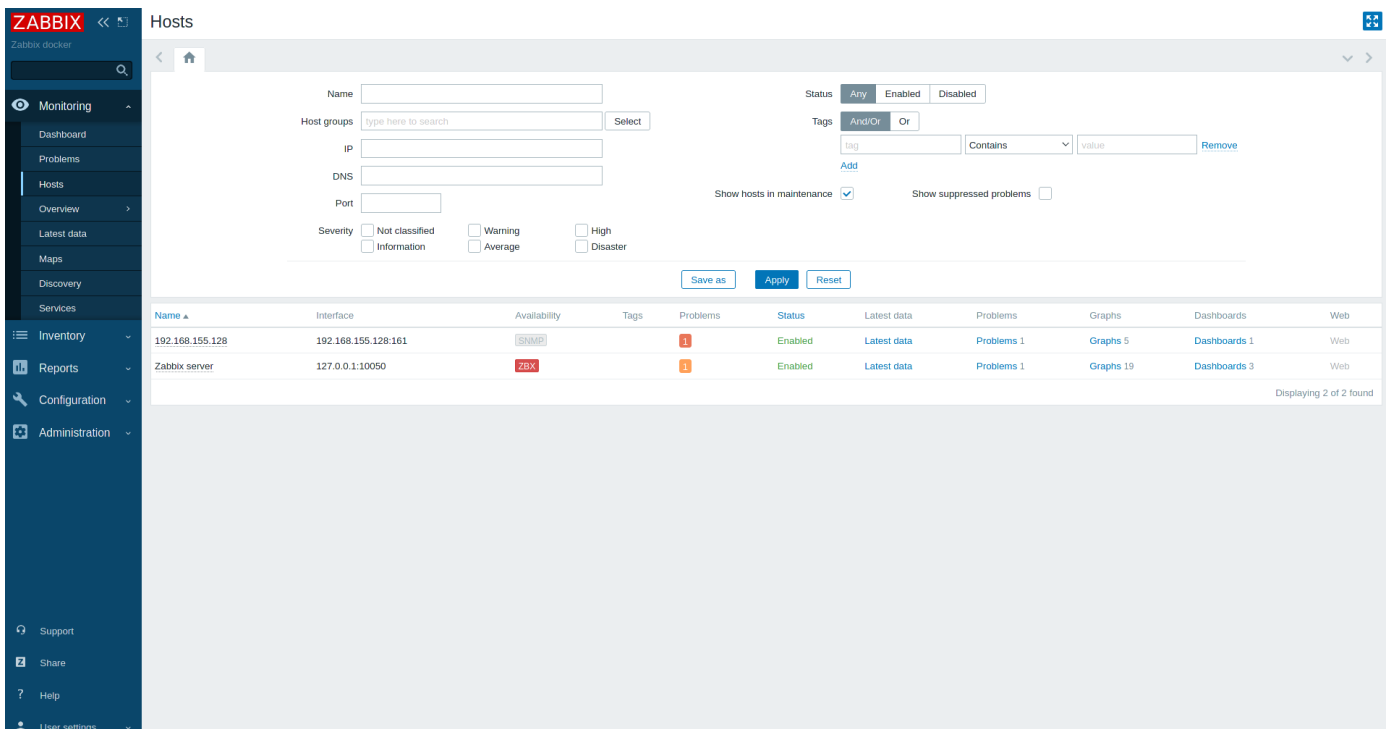
The result is that the templates are available in Zabbix.



4.5.2. Create ISVA hosts in Zabbix

The next step is to create the hosts, based on the ISVA inventory.

```
cd zabbix_playbooks
ansible-playbook create_zabbix_hosts.yml
```



At this moment, it's normal that there's an error. We have not configured IBM Security Verify Access for snmp polling nor for the REST connections that will be made.



zabbix server in constant error

Because we've not installed a Zabbix Agent container, the Zabbix server will be in constant error. We'll ignore that for now.

Chapter 5. ISVA configuration with Ansible

5.1. Configure Python

You could use the virtual environment we created earlier (`~/venvzabbix`), but it's cleaner to create a separate virtual environment for working with the IBM ISAM Ansible python code.

```
python3 -m venv ~/venvibmsecurity
source ~/venvibmsecurity/bin/activate
```

Install ansible and the ibmsecurity package.

```
pip install ibmsecurity
pip install jmespath
pip install ansible
```

5.2. Ansible example files

You can find an Ansible example configuration here : [ansible/isva_playbooks/](#)

The playbook use an inventory that is ready for deployment of ISVA.

You may need to update some variables !

- host name: in the hosts file and the host_vars directory
- zabbix hostname: is configured to depend on the hostname of the ISAM appliance:

```
snmp_zabbixproxy: "{{ inventory_hostname.split('.')[0] | join('.') }}.1"
```

- passwords: they are in vault.yml files.

There is an inventory included [ansible/inventory](#)

The vault files are NOT encrypted for this demo, but obviously they should be (using ansible-vault)

5.3. Galaxy

Install the collections that are required.

```
cd isva_playbooks
ansible-galaxy collection install -r collections/requirements.yml
```

The playbooks use the ISAM inventory that is provided here (it's configured in the `ansible.cfg` in each playbook directory)

5.4. ISVA playbooks

Go to the directory of the `isva_playbooks` and execute the playbooks.

5.4.1. Prepare the inventory

Modify the `hosts` file and the `host_vars` directory to match the IP address of your ISAM machine. The network parameters will then be assigned based on the IP range of that IP address (in a /24 subnet), so if the IP address of your host is

- 192.168.151.128

The runtime interface will be:

- 192.168.151.200

The proxy interface will be:

- 192.168.151.201

5.4.2. Configure system in ISVA

This playbook configures the networking.

```
cd isva_playbooks
ansible-playbook 0_configure_system.yml
```

5.4.3. Configure monitoring in ISVA

This playbook configures a monitoring user (that is used by Zabbix to poll REST api's). It also configures the Liberty runtime, with an interface IP address and to enable the monitoring REST endpoint.

```
cd isva_playbooks
ansible-playbook 1_configure_monitoring.yml
```

5.4.4. Configure snmp alert in ISVA

```
cd isva_playbooks
ansible-playbook 2_configure_snmp_alerts.yml
```

```
cd isva_playbooks
ansible-playbook 3_enable_snmp_polling.yml
```

5.4.5. Create a test reverse proxy

This creates a reverse proxy , with a junction pointing to your Zabbix server. The result is that you can access Zabbix through an ISVA junction.



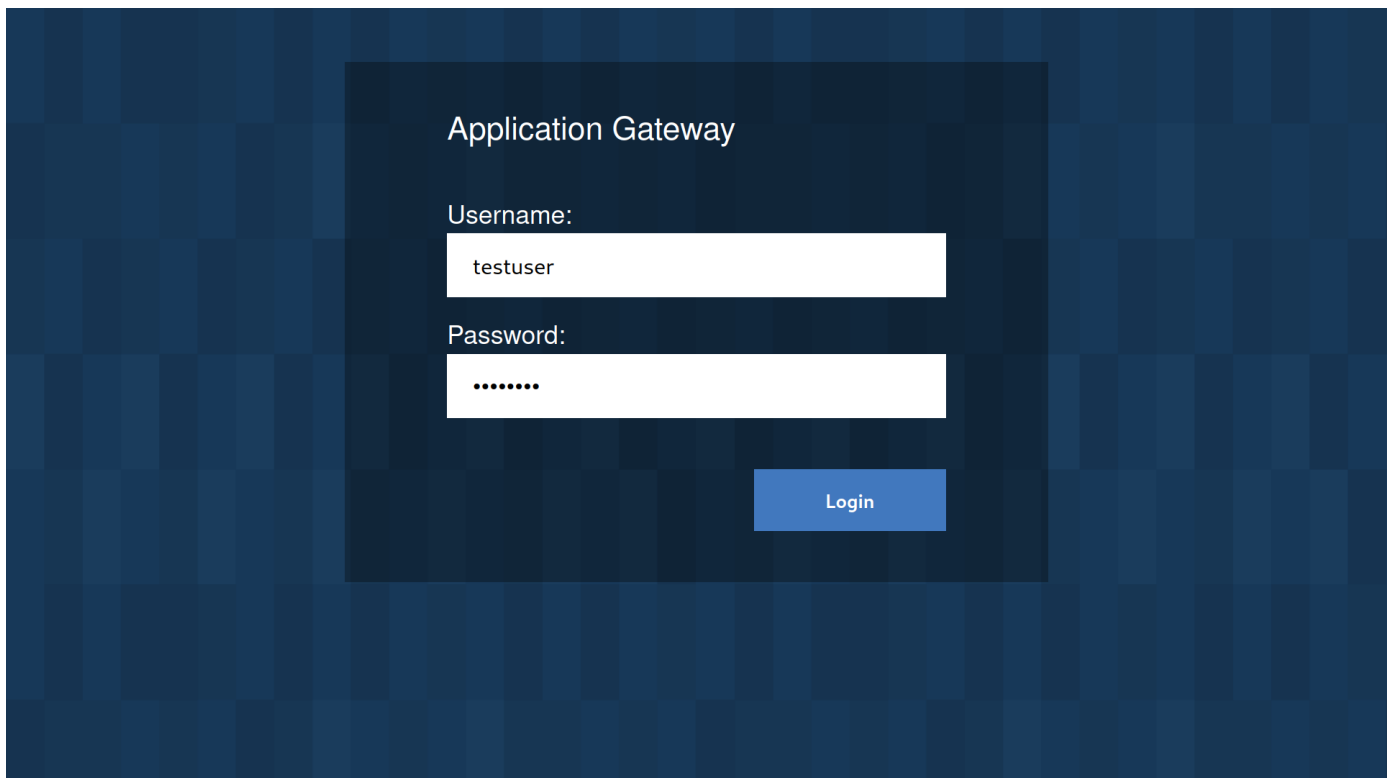
Users

The playbook does not create ISVA users. So on a clean ISVA appliance, you'd have to create a User in the local policy server (via Policy Administration)

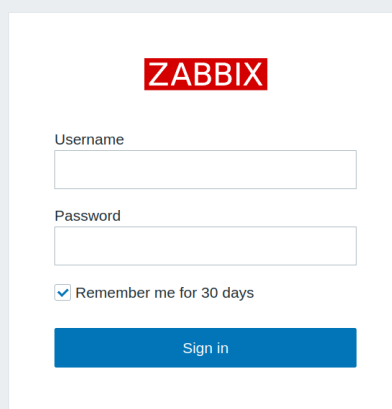
```
cd isva_playbooks
ansible-playbook 4_create_reverseproxy.yml
```

The Zabbix server is now available through WebSeal.

<https://192.168.151.201/zabbix>



There is no Single Sign on integration , so you'll have to login to Zabbix after authenticating to ISVA.



ZABBIX

Username

Password

Remember me for 30 days

[Sign in](#)

[Help](#) • [Support](#)

© 2001–2021, Zabbix SIA

Chapter 6. Zabbix Traps handling

The traps are the most difficult element to understand in the setup. ISVA will send traps to port 162/udp on the address that is configured (see the playbooks).

The Zabbix docker snmp trap container picks them up and processes them using a specific perl script. Then the Zabbix server processes the traps .

6.1. Zabbix logs

To look at the logs, you can use "podman logs". It may however be necessary to increase the loglevel. Increase the global log level:

```
sudo podman exec -ti zabbix-server-mysql zabbix_server -R log_level_increase
```

You can `tail` the logs to see if anything is wrong.

```
sudo podman logs -f zabbix-server-mysql
```

6.2. Make sure the zabbix server is enabled to handle snmp traps

This is an environment parameter for the container version, and a parameter in the zabbix configuration file for the system version of Zabbix.

6.3. Traps on the zabbix trap container

The Zabbix trap container receives the traps in the raw format, so this looks slightly different from the data that the

Zabbix server will process.

```
2021-12-05 13:03:52 UDP: [192.168.151.128]:55078->[10.88.0.11]:1162 [UDP:
[192.168.151.128]:55078->[10.88.0.11]:1162]:
DISMAN-EVENT-MIB::sysUpTimeInstance = 7736438    SNMPv2-MIB::snmpTrapOID.0 = OID:
SNMPv2-SMI::enterprises.2499.0.2    SNMPv2-SMI::enterprises.2499.1.4.1.1.1.1.0 =
STRING: "1638709431"    SNMPv2-SMI::enterprises.2499.1.4.1.1.1.2.0 = STRING: "2021-
12-05T14:03:51+0100"    SNMPv2-SMI::enterprises.2499.1.4.1.1.1.3.0 = STRING:
"WGAWA0665I"    SNMPv2-SMI::enterprises.2499.1.4.1.1.1.4.0 = INTEGER: 3654    SNMPv2-
SMI::enterprises.2499.1.4.1.1.1.5.0 = STRING: "WGAWA0665I The reverse proxy
instance 'test' was started by user 'admin@local'."    SNMPv2-
SMI::enterprises.2499.1.4.1.1.1.6.0 = STRING: "name=system,priority=low"
```

6.4. Unmatched traps

The snmp match is a bit annoying and expects an IP address to match the IP address sent in the actual trap. See [Zabbix host configuration](#) for details.

This requires the configuration option "Log unmatched SNMP traps" to be set (Administration/General/Other). This is active by default.

So if you see the following in the log files for the Zabbix server, the traps are not matched. This is most likely because the sending IP address is not in the host configuration's list of IP addresses.

```
205:20211204:150947.236 unmatched trap received from "192.168.151.200":
20211204.150945 UDP: [192.168.151.200]:16893->[10.88.0.10]:1162
DISMAN-EVENT-MIB::sysUpTimeInstance = 491692
SNMPv2-MIB::snmpTrapOID.0 = SNMPv2-SMI::enterprises.2499.0.2
SNMPv2-SMI::enterprises.2499.1.4.1.1.1.1.0 = "1638630585"
SNMPv2-SMI::enterprises.2499.1.4.1.1.1.2.0 = "2021-12-04T16:09:45+0100"
SNMPv2-SMI::enterprises.2499.1.4.1.1.1.3.0 = "GLGSY0103I"
SNMPv2-SMI::enterprises.2499.1.4.1.1.1.4.0 = 338
SNMPv2-SMI::enterprises.2499.1.4.1.1.1.5.0 = "GLGSY0103I The user,
'monitoring01@local' from '192.168.151.1', was successfully authenticated."
SNMPv2-SMI::enterprises.2499.1.4.1.1.1.6.0 = "name=system,priority=low"
```

6.5. Send test events

You can send test events to the Zabbix trapper, to verify if the trapper works:

```
yum install net-snmp-libs
mkdir -p /usr/local/share/snmp/mibs
snmptrap -v 1 -c public 127.0.0.1 '.1.3.6.1.6.3.1.1.5.3' '0.0.0.0' 6 33 '55'
.1.3.6.1.6.3.1.1.5.3 s "teststring000"
```

6.6. Polling SNMP

Likewise, to see what data is available in the ISVA SNMP polling data, you can use `snmpwalk` (also part of `net-snmp-libs`).

```
snmpwalk -m ALL -v 2c -c public <isam hostname> HOST-RESOURCES-MIB::host
```

6.7. MIB file

Make sure you've put the ISAM MIB File in the volume mount for the Trapper. This allows the trapper to process the raw incoming trap, and clean it up a bit before handing it over to the Zabbix Server.



MIB file

It's not strictly necessary to do that, because the Zabbix server will just as merrily process the raw traps. This does mean you need to adapt the rules a bit for the traps in the [ISVA OS Template](#). Specifically the following regular expression:

```
SNMPv2-SMI::enterprises\.2499\.1\.4\.1\.1\.1\.5\.0 = "(.*)" \1
```

Chapter 7. MIB MANAGER (Netcool)

The MIB file that is available at the time of writing for ISAM/ISVA, is a bit too simple. Zabbix does not care too much about MIB files, but other snmp trap tools do, for instance IBM's own Netcool/Omnibus.

The MIB Manager tool is a good way to validate the MIB file and possibly to adjust it.

7.1. Download MIB manager

As IBM'er, you can find the e-assembly on the internal systems.

Passport Advantage	Product or component
CJ8KCEN	IBM Tivoli Netcool/OMNIBus v8.1 for Netcool Operations Insight v1.6.3 Multiplatform English eAssembly (CJ8KCEN)

MIB Manager is part of this Core package.

7.2. Install MIB Manager

I've installed it on Linux. I'm sure you can figure out the installation on Windows. MIB Manager uses IBM Installation manager.



MIB manager

In the installer, only select "MIB Manager", leave the rest unchecked.



Recent linux kernels

Check out the technote below to enable MIB Manager to work on recent linux kernels:

<https://www.ibm.com/support/pages/apar/IJ29318>

7.3. Start the tool

On linux, start the tool using this command:

```
/opt/IBM/tivoli/netcool/omnibus/bin/nco_mibmanager
```

7.4. Import the MIB

MIB Manager should show that the import succeeded.

7.4.1. Export (generate traps)

Select the IBM MIB definition you just imported.

Click "Export"

Select Netcool

The screenshot shows the Netcool MIB Manager application window. The 'MIB Modules' pane on the left lists various MIB modules, with 'ISS-MIB (ISS-MIB (202111220000Z))' selected. The 'OID Tree' pane on the right shows a hierarchical tree structure, with 'iss' selected under the '3 org' branch. A dialog box titled 'Export Complete' is overlaid on the interface, displaying the message 'Export successful, 1 objects exported' and an 'OK' button. Below the dialog, a console window shows the command being executed: 'Processing template /opt/IBM/tivoli/netc...sgi/2/0/.cp/templates/nckl_3_0/snmp.tpl'. At the bottom, the 'Details' pane shows the following information:

Field	Value
MIB_MOD	ISS-MIB (ISS-MIB (202111220000Z))
MODULE	ISS-MIB

This generates the trap rules based on the MIB file, in the export directory (eg. /opt/IBM/tivoli/netcool/omnibus/var/mibmanager/export/20211124142846-nckl_3_0)

Chapter 8. Resources

List of provided attachments:

- [isam_new.mib](#)

All attachments and examples:

- [Attachments and examples all-in-one file](#)

Change Log

Revision	Date	Responsible	Description
Initial	01 Jun 2020	IBM Security Knowledge Exchange	Initial package template
0.1.0	10 Dec 2021	Tom Bosmans	Original author. First draft.
1.0.0	12 Dec 2021	Tom Bosmans (Content)	

Document Source

Package Author: Tom Bosmans tom.bosmans@be.ibm.com

Package Owner: IBM Security Expert Labs skesel@be.ibm.com

If you wish to suggest a change to this document, please do so. You have the following options:

- Email the owning author above or reach out on Slack at IBM Security #ske, or <https://ibm.biz/ske-slack>
- Raise an issue on the package's github repository at: <https://github.ibm.com/skelib/1890-isva-monitoring-with-zabbix/issues>
- Leave feedback comments where you have obtained access to this material. For example, SLA course feedback.

© Copyright IBM Corporation 2021

IBM Security
Route 100
Somers, NY 10589
U.S.A.

Produced in the United States of America
February 2018
All Rights Reserved

IBM, the IBM logo, ibm.com, and IBM X-Force are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol ® or ™, these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at: ibm.com/legal/copytrade.shtml

Other product, company or service names may be trademarks or service marks of others.

References in this publication to IBM products or services do not imply that IBM intends to make them available in all countries in which IBM operates.