From Default to Defended: Strengthening Zabbix Security

Genādijs Jeniceks, Zabbix Engineer

WHY?

Communication over the world wide web

Communications over the internet should always be secured

“Weakest link”

Because Zabbix can have access to many other systems, it needs to be properly defended

Protect valuable data

Zabbix can contain very valuable data which should not be exposed to others

Availability of monitoring

If Zabbix is disrupted, you lose visibility into your environment

Security focus points:

Communication

- Encryption between Zabbix components
- HTTPS

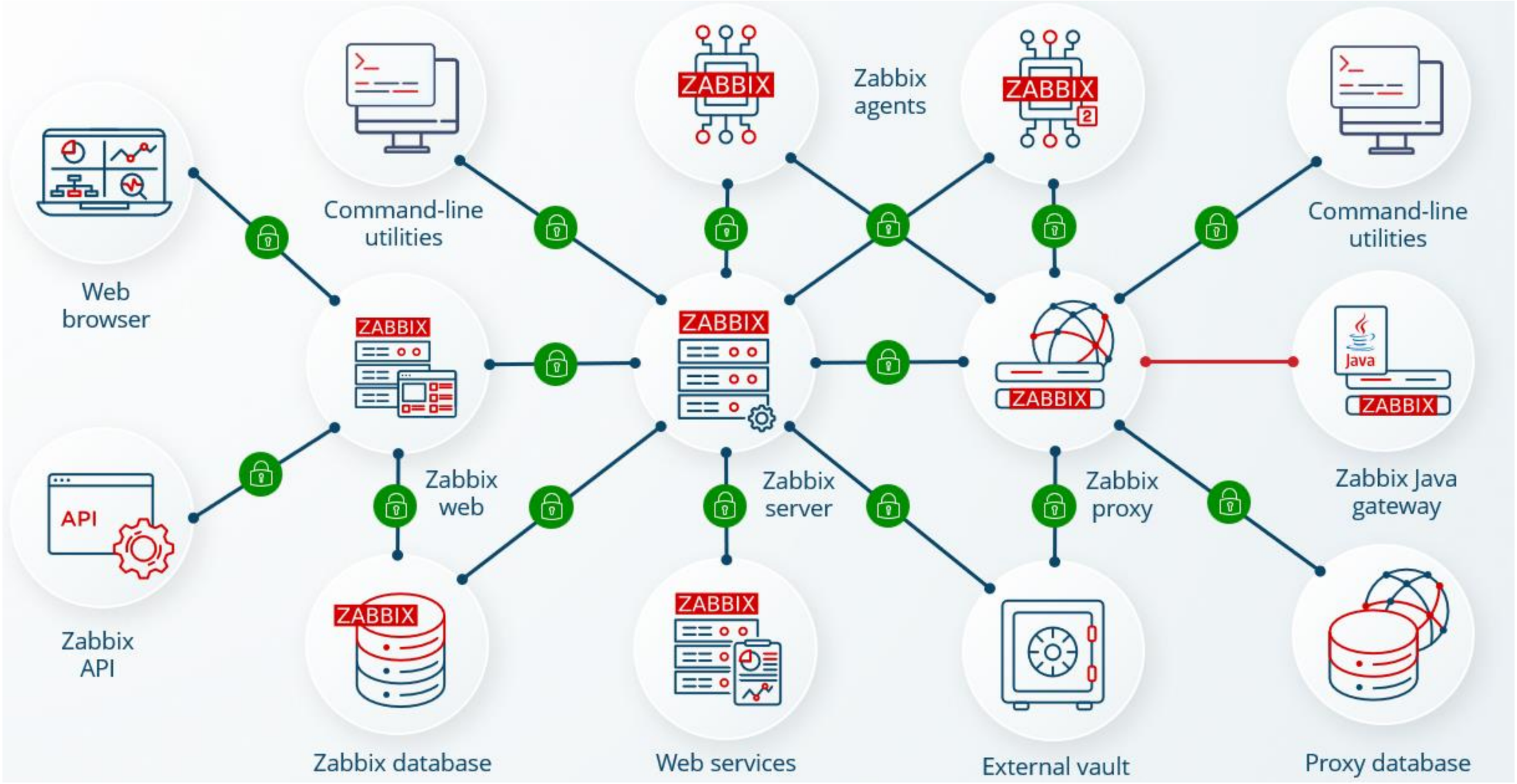
Permissions

- RBAC
- People see what they need to see

Other

- Product security
- Limiting script execution
- API security
- OS security

ZABBIX


Web browser
Command-line utilities
Zabbix agents
Command-line utilities
Zabbix API
Zabbix web
Zabbix server
Zabbix proxy
Zabbix Java gateway
Zabbix database
Web services
External vault
Proxy database
Java

Default Zabbix installation

The default Zabbix setup, although very convenient and running smoothly, does not offer any security-related settings. All of them need to be manually set up.

Í ŎĢi i Ĩ Ĭ

Frontend is not using HTTPS

No encryption between components

All communications between Zabbix components are in plain text

No permissions

Default Admin account without any restrictions and no other users

Where to start?

Where to start?

ĈĖ BĢİ İ Ĩ Ĭ

1B. DB encryption

2. Zabbix server and web server encryption

3. Zabbix server and proxy encryption

1A. HTTPS

HTTP is valuable, as it establishes a standardized protocol for communication between web browsers and servers, ensuring universal accessibility across the internet. Additionally, it is lightweight, adaptable, and easily extended to support secure transmission through HTTPS.

Encryption

HTTPS encrypts all data exchanged between client and server using SSL/TLS

Certificate-based

Confirms the website's legitimacy and ownership

Global standard

HTTPS is the worldwide standard for secure web communication.

1A. DB encryption

Database encryption is required to protect sensitive data from unauthorized access, ensuring confidentiality even if the database is compromised. It also helps organizations comply with regulatory standards and safeguards user trust by securing critical information at rest and in transit.

Protects data

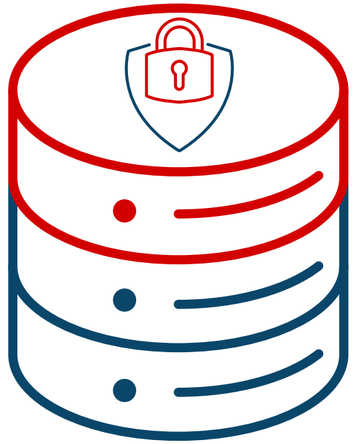
Prevents interception or tampering of queries and results

Secures communications

Encrypts data in transit between Zabbix and the database using TLS/SSL

Available out of the box and easy to implement

MySQL and PostgreSQL include built-in encryption features



2. Zabbix server and web server encryption

Zabbix supports encryption between Zabbix server and the web server

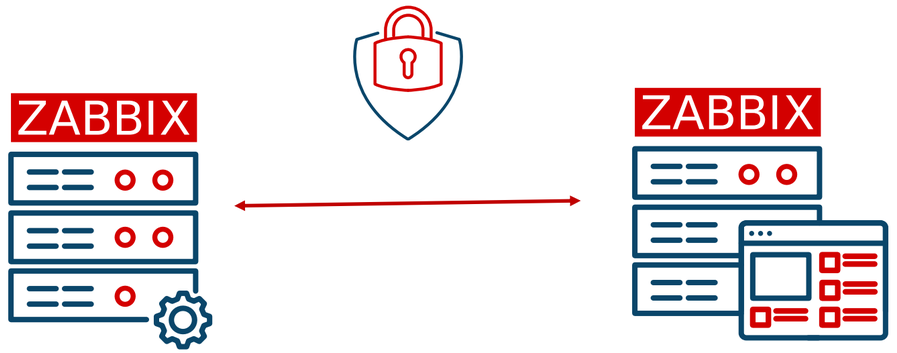
Implemented in 7.4

Regular TLS setup

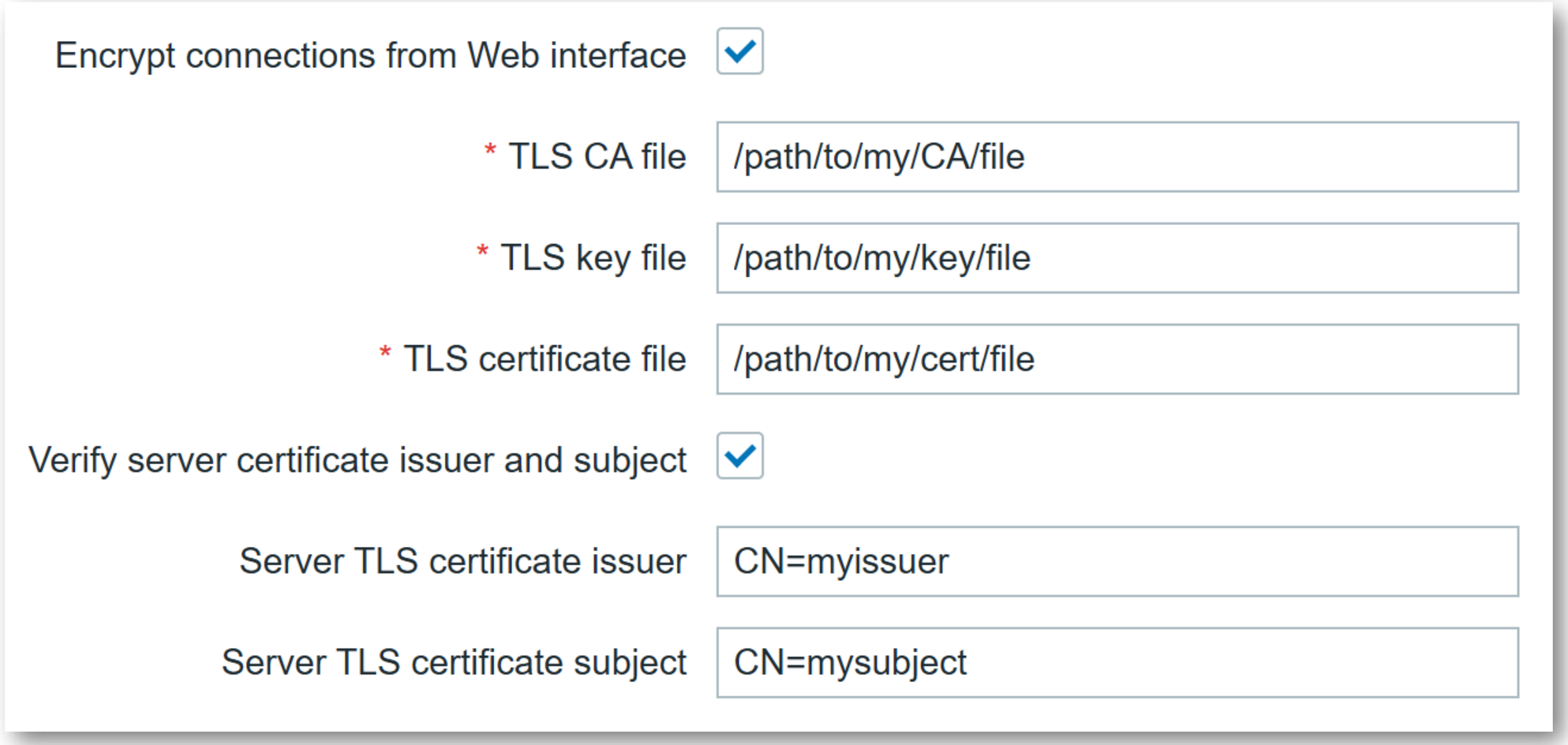
Se up is simple and follows any other TLS-based setup

Protects valuable data

Testing items, creating tasks, and other communication is encrypted



Encryption settings in web setup



Encryption settings in web configuration file

```
$ZBX_SERVER_TLS['ACTIVE'] = '1';
$ZBX_SERVER_TLS['CA_FILE'] = '/path/to/ca/file';
$ZBX_SERVER_TLS['KEY_FILE'] = '/path/to/key/file';
$ZBX_SERVER_TLS['CERT_FILE'] = '/path/to/cert/file';
$ZBX_SERVER_TLS['CERTIFICATE_ISSUER']  = 'issuer';
$ZBX_SERVER_TLS['CERTIFICATE_SUBJECT'] = 'subject';
```

3. Zabbix server and proxy encryption

Zabbix allows secure, encrypted communication between the server and its proxies to protect monitoring data as it travels across networks. This ensures data confidentiality, authenticity, and integrity in distributed monitoring setups.

Encrypted Communication

Certificate or PSK-based

Ę ÑCEPÓÓMÁPÒÓÑ

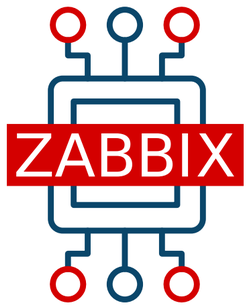
Strengthens the overall security posture of the Zabbix monitoring infrastructure

Simple configuration and compatibility

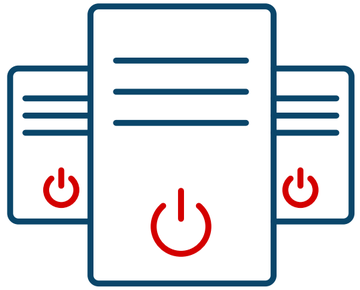
Fully compatible across versions and easy to roll out to multiple proxies

What to do next?

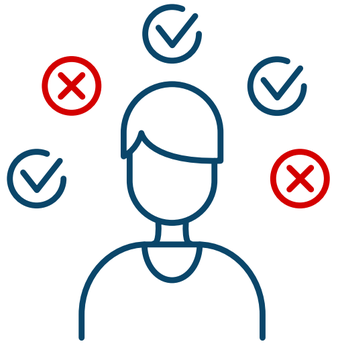
4. Agent encryption



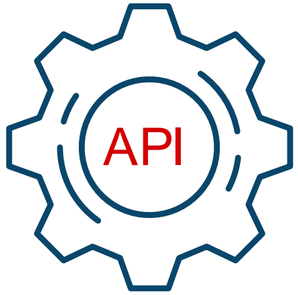
5. OS security



6. RBAC



7. API security



4. Agent encryption

Zabbix agent encryption safeguards communication between monitored hosts and the Zabbix server or proxy.

Encrypted communication

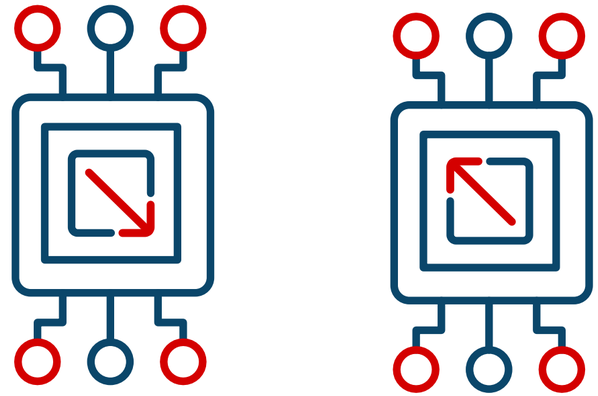
Certificate or PSK-based

Command line utility security

Zabbix sender can also use encryption

Enhanced security posture

Ideal for securing critical systems and production environments



5. OS security

Proper configuration of firewalls, SELinux, and system hardening ensures that Zabbix services run securely and are protected from unauthorized access or attacks.

Firewall Ccnfig uration

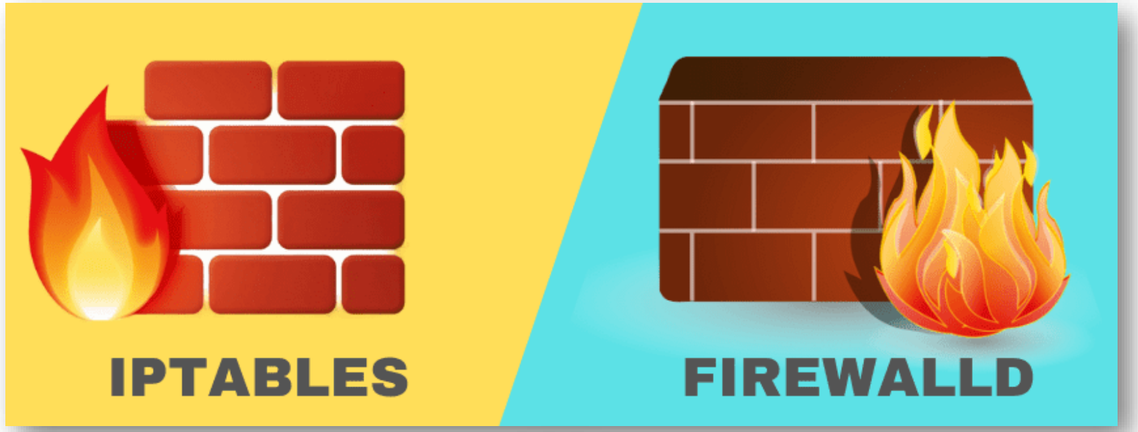
Typically, firewalld or iptables are used

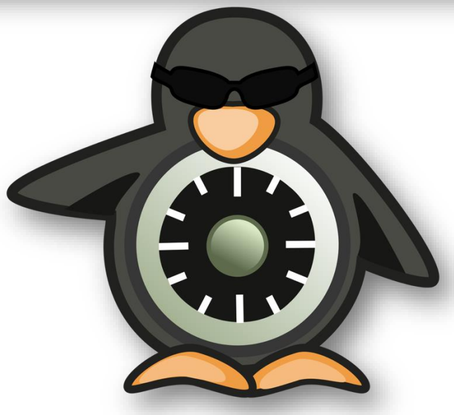
SELinux setup

Keep SELinux enabled in enforcing mode for added protection

User and permission management

All Zabbix components are run by limited “zabbix” user





6. RBAC

API security in Zabbix ensures that communication with the system's API endpoints is properly authenticated, authorized, and protected from misuse. Implementing strong security controls prevents unauthorized access, data leaks, and potential automation abuse.

Permissions

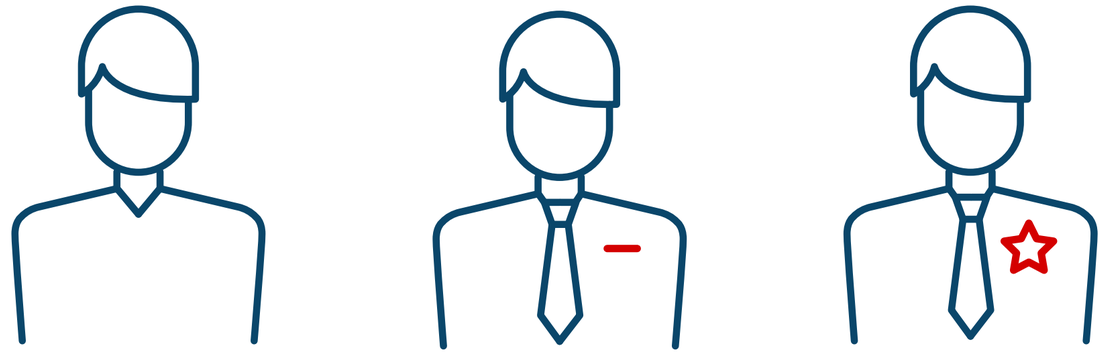
Permissions control what hosts/templates Users and Admins can see

Roles

Roles help control what users can do on the frontend

Two-factor authentication

Built in 2FA support for TOTP or Duo universal prompt



7. API security

Database encryption is required to protect sensitive data from unauthorized access, ensuring confidentiality even if the database is compromised. It also helps organizations comply with regulatory standards and safeguard user trust by securing critical information at rest and in transit.

Limited API user

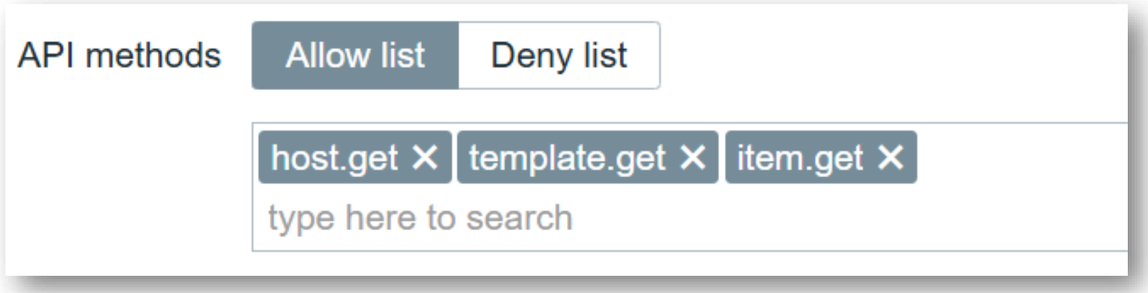
API user should only be able to access the API, not the frontend

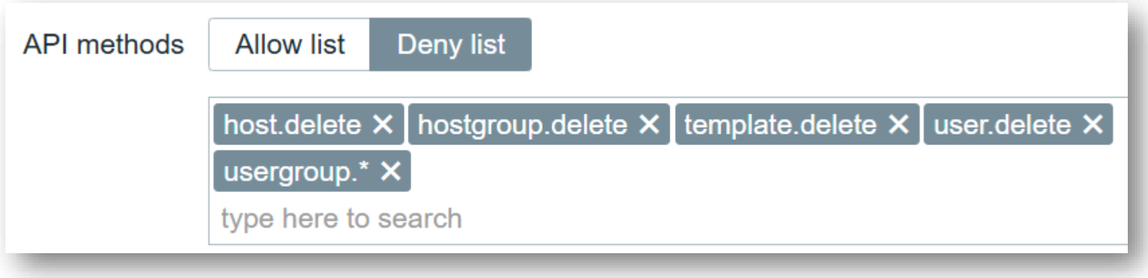
Allow and deny lists

Roles provide a possibility to create, allow, or deny lists for API methods

Secure the API token

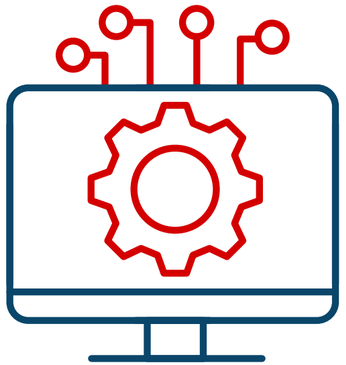
Make sure the API token has no unauthorized access





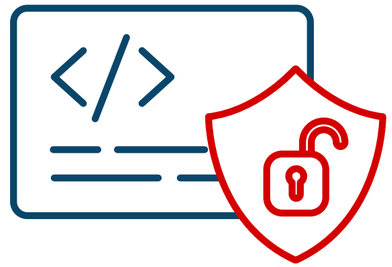
Further improvements

8. Securing macros



9. Using external vault

10. Restricting script running



8. Securing macros

Macros in Zabbix are powerful variables used to store configuration values such as credentials, paths, or parameters. Securing macros is essential to protect sensitive data from exposure and maintain the integrity of automated monitoring actions.

Secret macros

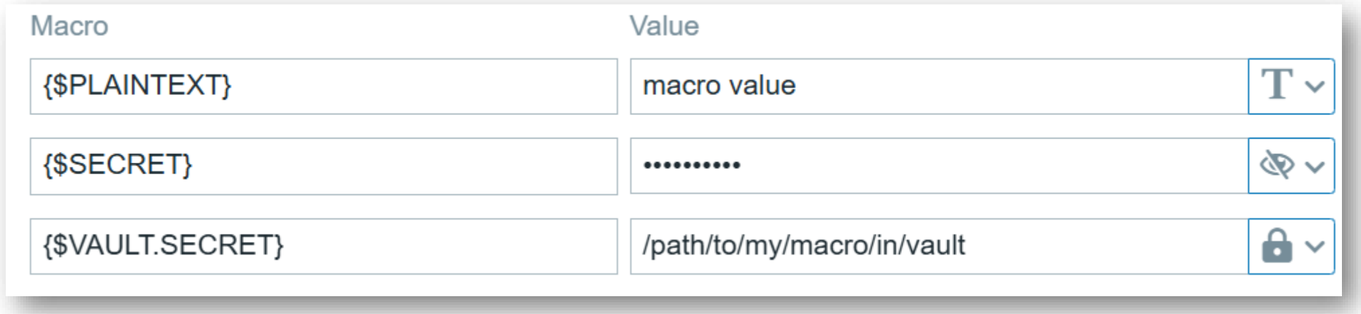
Prevents sensitive data such as passwords or tokens from being visible to users

Macro scope management

Host-level or template-level macros are preferable to global ones

Avoid storing plain text credentials

Do not store passwords, API keys, or tokens as plain text in regular macros. Think about a vault



9. Using an external vault

Integrating an external vault with Zabbix enhances security by storing sensitive credentials, tokens, and secrets outside the monitoring system. This approach minimizes the risk of credential leaks and ensures compliance with enterprise security standards.

Centralized secret management

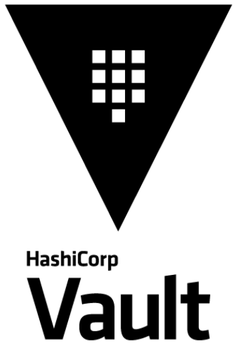
HashiCorp and Cyberark vaults are supported

Granular access control

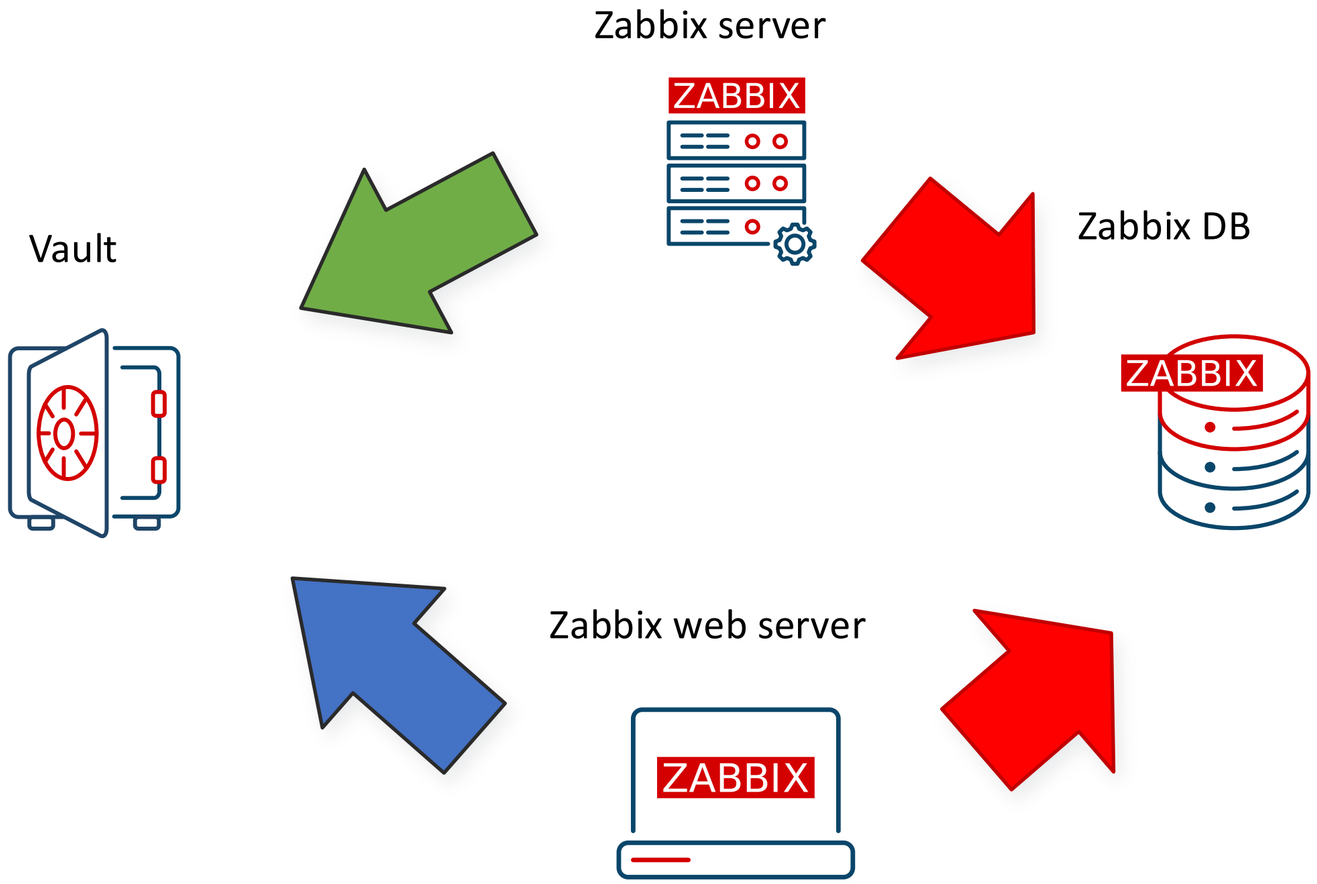
Different access level on vault data to Zabbix server, frontend, and proxy

Modern security measures

Aligns Zabbix credential management with current enterprise security best practices



Vault usage on Zabbix server

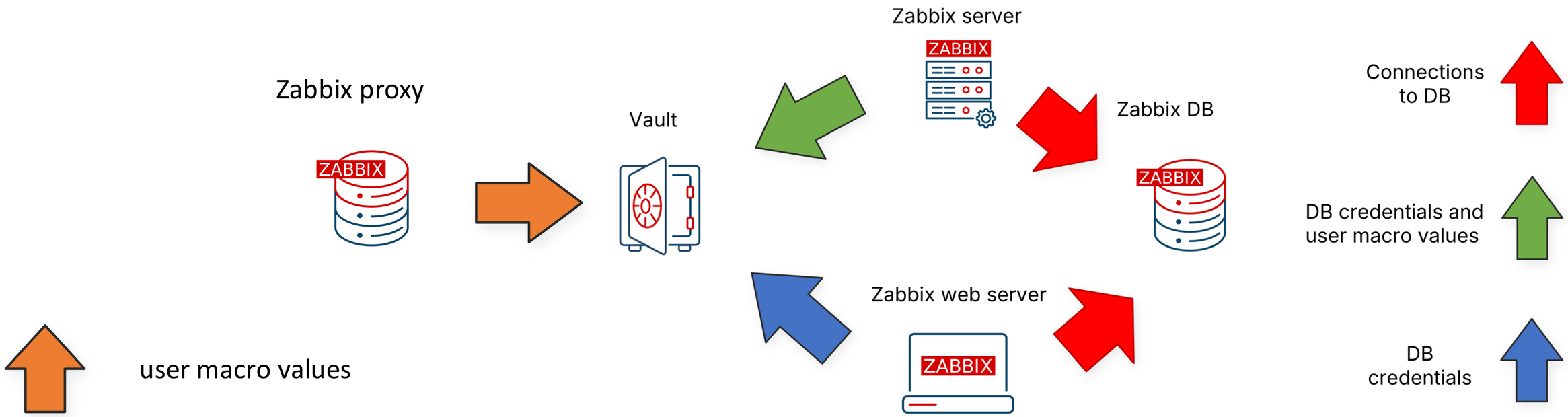


Connections to DB

DB credentials and user macro values

DB credentials

Vault usage on proxies



10. Restricting script running

Restricting script execution in Zabbix helps prevent unauthorized or unsafe commands from being run on monitored systems. Proper control over script locations, permissions, and user access ensures automation remains secure and compliant with organizational policies.

Zabbix server and proxy does not allow script running on new installation

If the configuration files are provided by Zabbix, the default setting for server is enabled

Agent script restriction

Use AllowKey and DenyKey parameters

RBAC control

L7–&‹ -⁄–Î› 7‹€ ‹ (7)‹ ›^€ RÎ)— (Á RÎ))‹R&–€– (
Î ¢€RÎ7 €)&7-ˆ)‹ } 7Î ˆ)‹ (Î7 (Á

Summary

1. HTTPS and DB security
2. Zabbix server and web server encryption
3. Zabbix server and proxy encryption
4. Agent encryption
5. OS security
6. RBAC
7. API security
8. Securing macros
9. Using external vault
10. Restricting script running

ZABBIX

ZABBIX

Acquired skills and knowledge

Secure your Zabbix data flows

with **Certificate** or **PSK encryption**

Restrict access to sensitive metrics

and learn how to define metric **allow and deny lists**

Add an extra layer of security to your sensitive data

by deploying **HashiCorp** or **CyberArk** vault

Make sure all your Zabbix components are perfectly secure

by following **latest** Zabbix security **advisories**

Security training

Thank you!