initMAX

ZABBIX PREMIUM PARTNER

ZABBIX CERTIFIED TRAINER

Webinar

# SNMP and SNMP traps in Zabbix 7.0

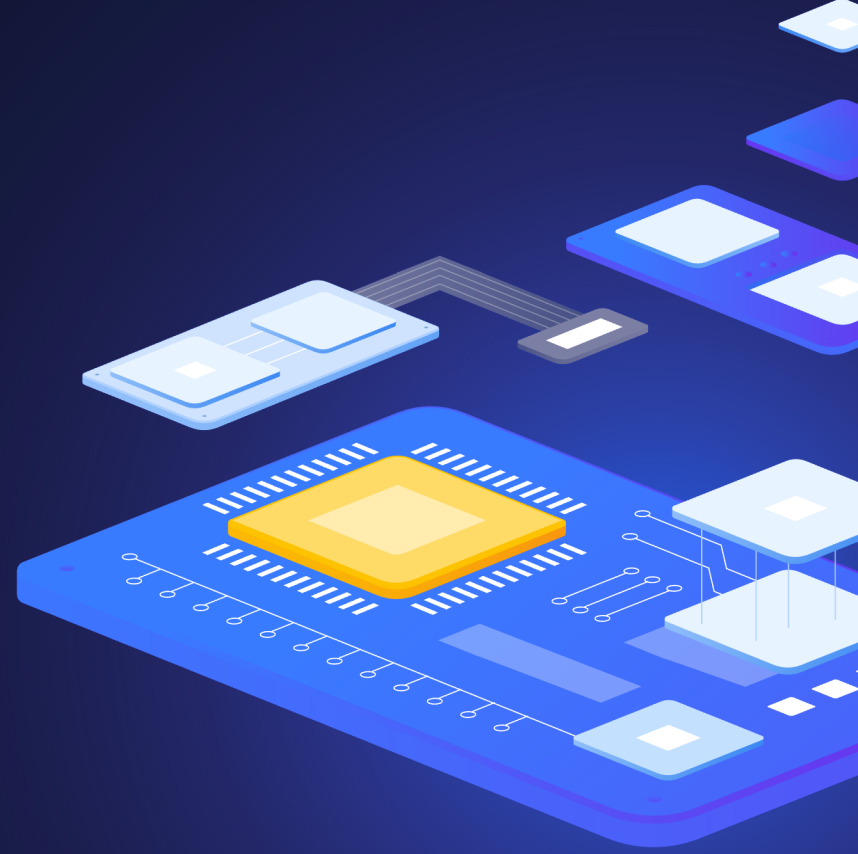
all our microphones are muted

ask your questions in Q&A, not in the Chat

use Chat for discussion, networking or applause

# 1

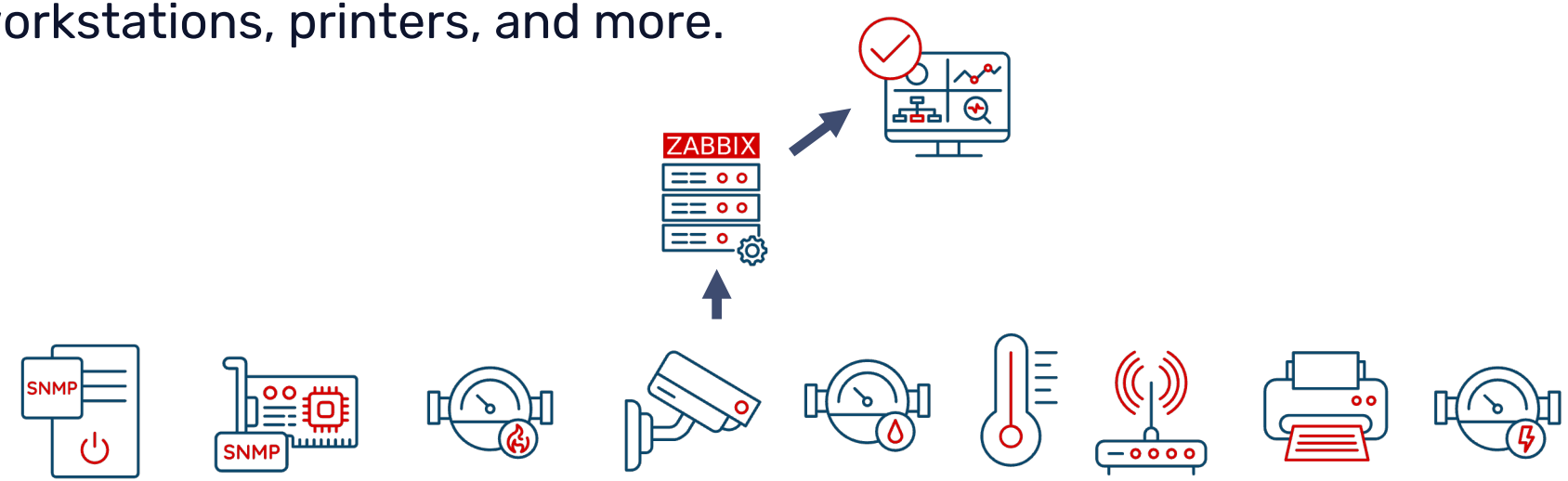
# SNMP protocol



initMAX

# Simple Network Management Protocol (SNMP)

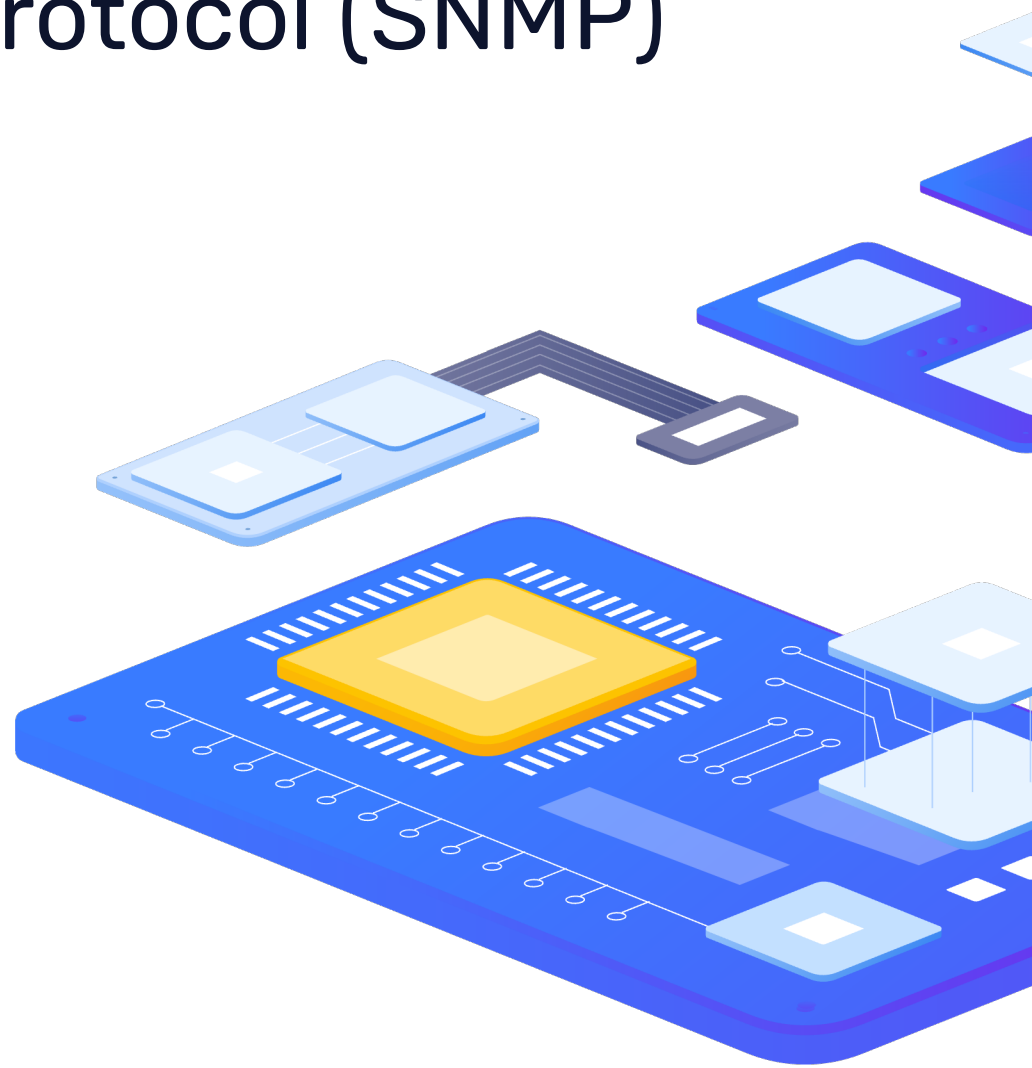
- Simple Network Management Protocol (SNMP) is an Internet Standard protocol for collecting and organizing information about managed devices on IP networks and for modifying that information to change device behavior.
- Devices that typically support SNMP include cable modems, routers, network switches, servers, workstations, printers, and more.



initMAX

# Simple Network Management Protocol (SNMP)

- The first version SNMPv1 was published in 1988
- SNMP has evolved over the years, and several versions of the protocol have been released, including SNMPv2c and SNMPv3



initMAX

# Simple Network Management Protocol (SNMP)

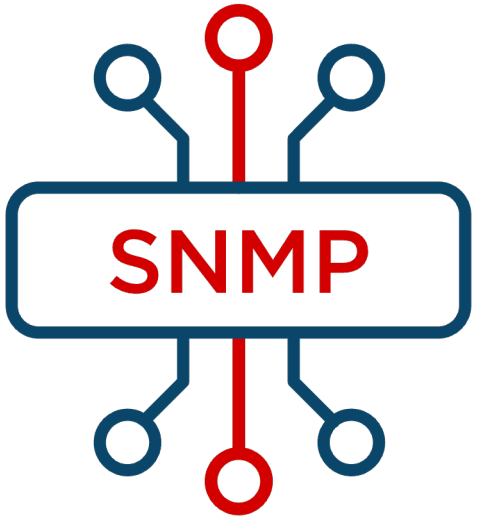
- SNMP agent checks- UDP port 161

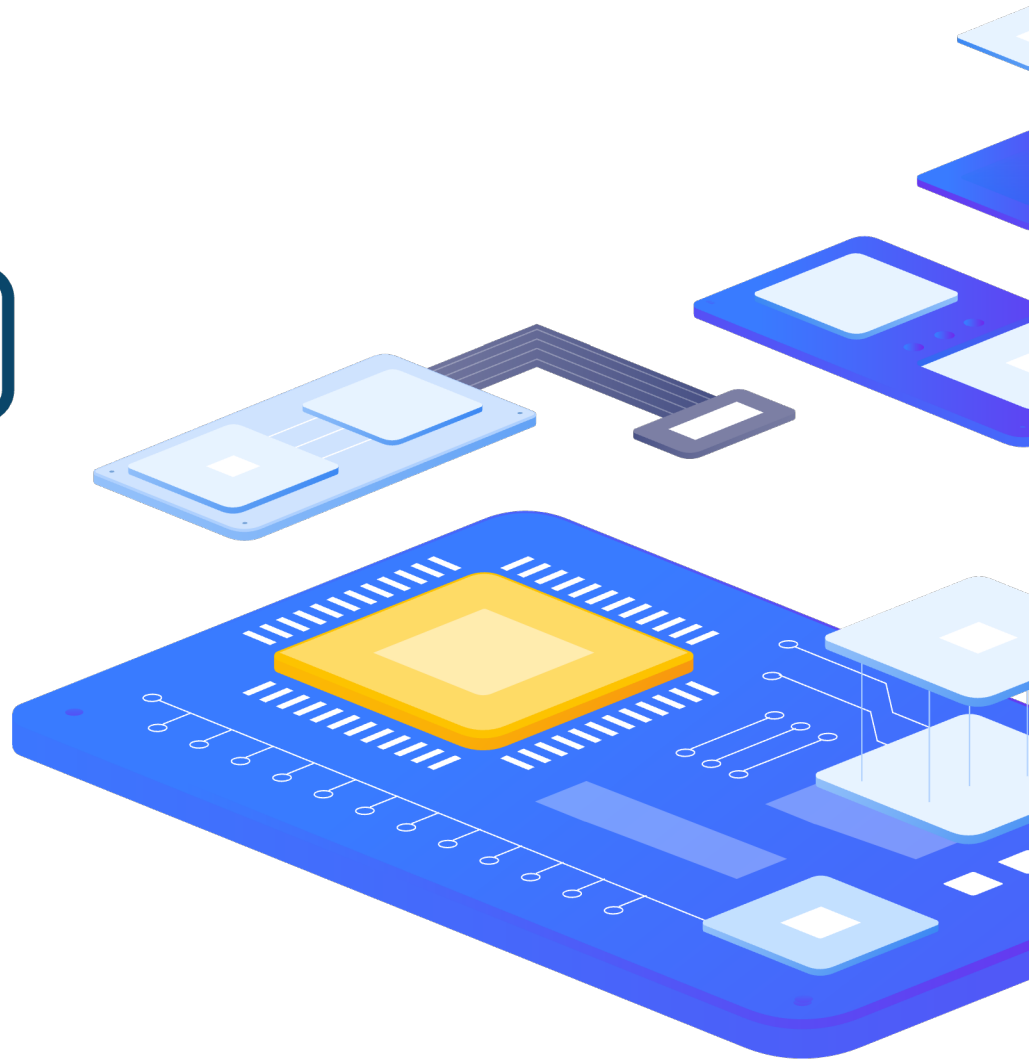

- SNMP Traps – UDP port 162

initMAX

# SNMP versions

- SNMP Version 1 - RFC 1213 - 1988
  - plain-text community string
  - only 32-bit counters supported
- SNMP Version 2c - RFC 1441 - 1993
  - plain-text community string
  - adds support for 64-bit counters
  - introduces GETBULK command
- SNMP Version 3 - RFC 2570 - 1999
  - adds authentication
  - adds encryption
  - improved error reporting and reliability
  - adds multiple SNMP contexts



initMAX

# Management information base (MIB)

- A management information base (MIB) is a database used for managing the entities in a communication network.
- Objects in the MIB are defined using a subset of Abstract Syntax Notation One (ASN.1) called "Structure of Management Information Version 2 (SMIv2)" RFC 2578.
  - The software that performs the parsing is a MIB compiler.
- The database is hierarchical (tree-structured) and each entry is addressed through an object identifier (OID).

initMAX

# Management information base (MIB)

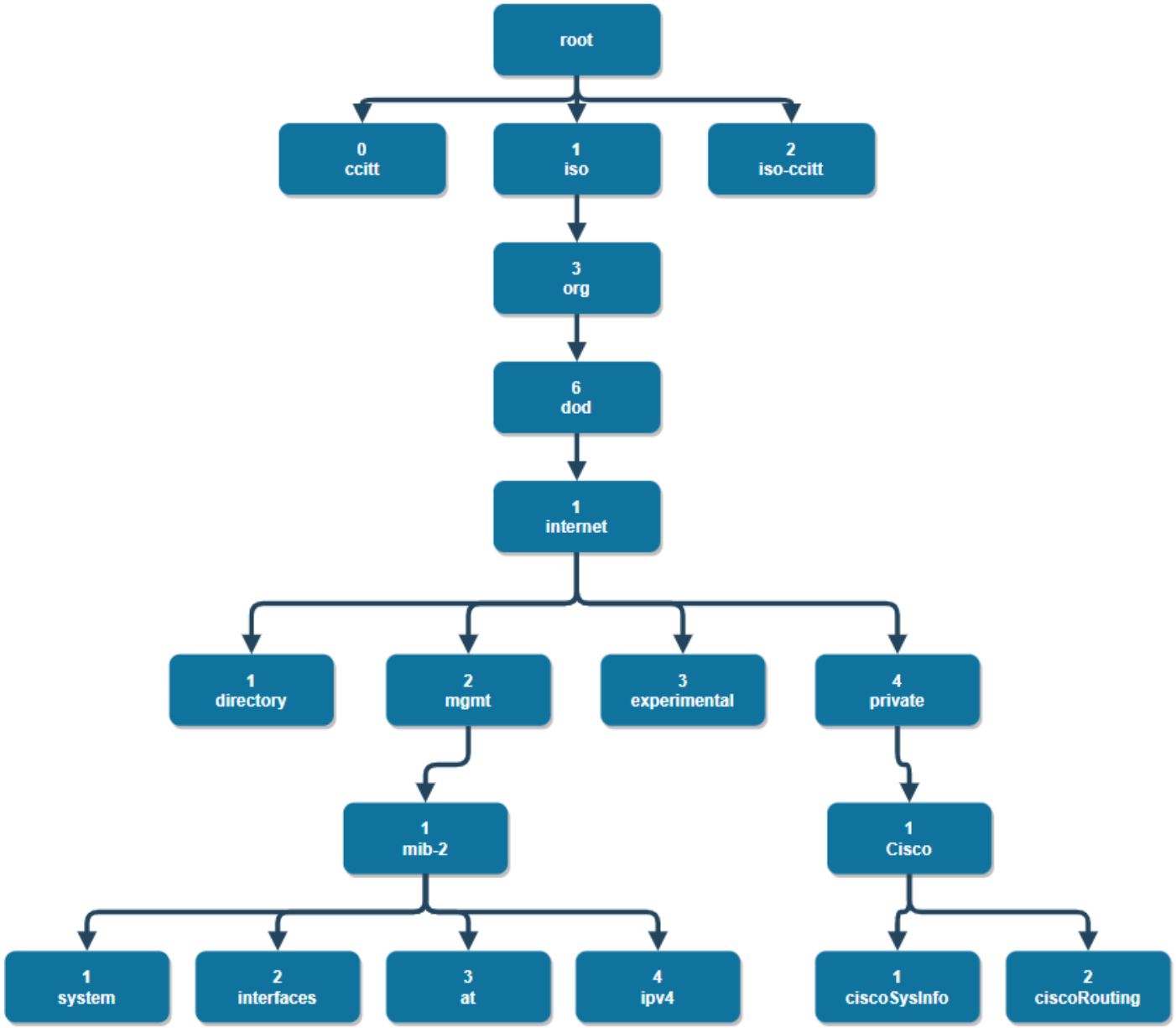
- RFC Definition:
- https://datatracker.ietf.org/doc/html/rfc2578

```
SNMPv2-SMI DEFINITIONS ::= BEGIN

-- the path to the root

org             OBJECT IDENTIFIER ::= { iso 3 } --"iso" = 1
dod             OBJECT IDENTIFIER ::= { org 6 }
internet        OBJECT IDENTIFIER ::= { dod 1 }

directory       OBJECT IDENTIFIER ::= { internet 1 }

mgmt            OBJECT IDENTIFIER ::= { internet 2 }
mib-2           OBJECT IDENTIFIER ::= { mgmt 1 }
transmission    OBJECT IDENTIFIER ::= { mib-2 10 }

experimental    OBJECT IDENTIFIER ::= { internet 3 }

private         OBJECT IDENTIFIER ::= { internet 4 }
enterprises     OBJECT IDENTIFIER ::= { private 1 }

security        OBJECT IDENTIFIER ::= { internet 5 }
```

initMAX

# MIB

- The database is hierarchical (tree-structured) and each entry is addressed through ar object identifier (OID).



initMAX

# Management information base (MIB)

- ASN.1 object notation

```
1 (iso)                                  -- ISO (International Organization for Standardization)
│
├── 3 (org)                              -- Organization (under ISO, organizations like DOD, IETF are defined here)
│   ├── 6 (dod)                          -- Department of Defense (U.S. Department of Defense)
│   │   └── 1 (internet)                 -- Internet OID (The Internet-related MIB objects, as defined by IETF)
│   │       ├── 2 (mgmt)                 -- Management Information (Standard MIB objects for network management)
│   │       │   └── 1 (mib-2)           -- MIB-2 (A set of standard MIB objects used in network management)
│   │       │       ├── 1 (system)      -- System-related information (e.g., device description, uptime)
│   │       │       ├── 2 (interfaces)  -- Interface-related information (e.g., network interfaces on the device)
│   │       │       ├── 3 (at)          -- Address Translation (ARP table or similar address/protocol management)
│   │       │       ├── 4 (ipv4)        -- IPv4 information (e.g., IP routing, address configuration)
│   │       │       └── 5 (icmp)        -- ICMP information (e.g., ICMP statistics, errors)
│   │       ├── 4 (private)              -- Private MIB (Manufacturer-specific MIBs, not standardized)
│   │       │   └── 1 (cisco)            -- Cisco-specific MIB (Cisco devices' private MIB objects)
│   │       │       ├── 1 (ciscoSysInfo)-- Cisco system information (e.g., system status, device model)
│   │       │       └── 2 (ciscoRouting)-- Cisco routing-related objects (e.g., routing protocols, interfaces)
│   │       └── 10 (enterprise)          -- Enterprise-specific MIB (Manufacturer-specific identifiers for custom MIBs)
│   │           └── 1 (hp)               -- HP-specific MIB (HP devices' private MIB objects)
│   │               └── 2 (hpSysInfo)   -- HP system-related information (e.g., system info, device stats)
└── 4 (iso)                              -- International Standard Organization (ISO-related information)
```

initMAX

# Management information base (MIB)

- Single object example 1:
  - Syntax
  - Access type
  - Status
  - Description

```
sysContact OBJECT-TYPE
        SYNTAX      DisplayString (SIZE (0..255))
        MAX-ACCESS  read-write
        STATUS      current
        DESCRIPTION
                "The textual identification of the contact person for
                this managed node, together with information on how
                to contact this person.  If no contact information is
                known, the value is the zero-length string."
        ::= { system 4 }
```

initMAX

# Management information base (MIB)

- Single object example 2:
  - Syntax – status definitions
  - Access type
  - Status
  - Description

```
ifOperStatus OBJECT-TYPE
    SYNTAX  INTEGER {
                up(1),        -- ready to pass packets
                down(2),
                testing(3),   -- in some test mode
                unknown(4),   -- status can not be determined
                              -- for some reason.
                dormant(5),
                notPresent(6),    -- some component is missing
                lowerLayerDown(7) -- down due to state of
                                  -- lower-layer interface(s)
            }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The current operational state of the interface.  The testing(3) state
indicates that no operational packets can be passed.  If ifAdminStatus is down(2)
then ifOperStatus should be down(2).  If ifAdminStatus is changed to up(1) then
ifOperStatus should change to up(1) if the interface is ready to transmit and
receive network traffic; it should change to dormant(5) if the interface is waiting
for external actions (such as a serial line waiting for an incoming connection); it
should remain in the down(2) state if and only if there is a fault that prevents it
from going to the up(1) state; it should remain in the notPresent(6)state if the
interface has missing (typically, hardware) components."
    ::= { ifEntry 8 }
```

initMAX

# Management information base (MIB) - Data types

- SNMPv2
  - support for 64-bit counters
- Working with counters

```
Integer    - Signed 32bit Integer (values between -2147483648 and 2147483647).
Integer32  - Same as Integer.
UInteger32 - Unsigned 32bit Integer (values between 0 and 4294967295).
Octet String - Arbitrary binary or textual data, typically limited to 255 characters
in length.
Object Identifier - An OID.
Bit String - Represents an enumeration of named bits. This is an unsigned datatype.
IpAddress  - An IP address.
Counter32  - Represents a non-negative integer which monotonically increases until it
reaches a maximum value of 32bits-1 (4294967295 dec), when it wraps around and starts
increasing again from zero.
Counter64  - Same as Counter32 but has a maximum value of 64bits-1.
Gauge32    - Represents an unsigned integer, which may increase or decrease, but
shall never exceed a maximum value.
TimeTicks  - Represents an unsigned integer which represents the time, modulo 232
(4294967296 dec), in hundredths of a second between two epochs.
Opaque - Provided solely for backward-compatibility, its no longer used.
NsapAddres - Represents an OSI address as a variable-length OCTET STRING.
```

initMAX

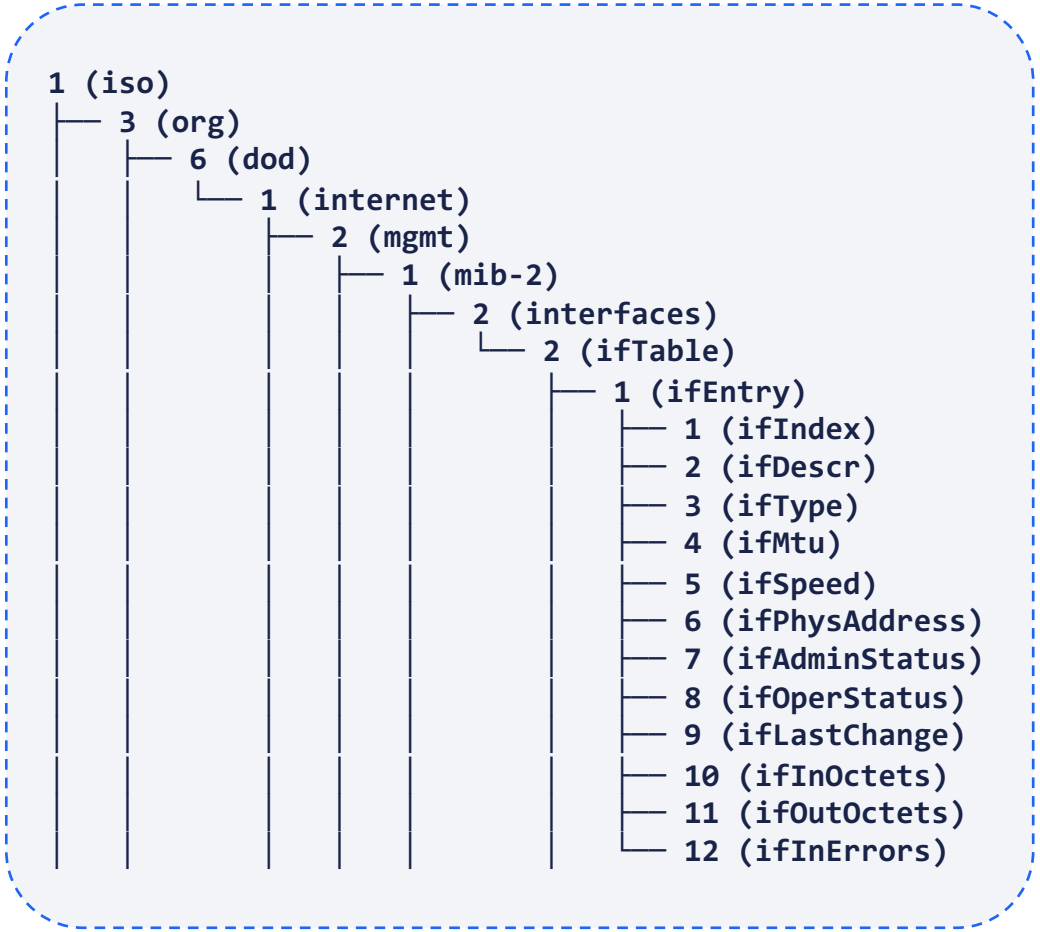
# Management information base (MIB) - Tables

```
1 (iso)                                                 -- ISO (International Organization for Standardization)
│
├── 3 (org)                                             -- Organization
│   ├── 6 (dod)                                         -- Department of Defense
│   │   └── 1 (internet)                                -- Internet OID (Standard MIB for Internet protocols)
│   │       ├── 2 (mgmt)                                -- Management Information (MIB-2, used for network management)
│   │       │   ├── 1 (mib-2)                           -- MIB-2 (Standard set of management objects for network devices)
│   │       │   │   ├── 2 (interfaces)                  -- Interfaces (part of MIB-2 for network interface management)
│   │       │   │   │   └── 2 (ifTable)                -- ifTable (Network if table, contains information about all interfaces)
│   │       │   │   │       ├── 1 (ifEntry)            -- ifEntry (Individual entry for each interface)
│   │       │   │   │       │   ├── 1 (ifIndex)       -- Interface index (unique identifier for each interface)
│   │       │   │   │       │   ├── 2 (ifDescr)       -- Interface description (e.g., "eth0", "eth1", etc.)
│   │       │   │   │       │   ├── 3 (ifType)        -- Interface type (e.g., Ethernet, Wi-Fi, etc.)
│   │       │   │   │       │   ├── 4 (ifMtu)         -- Maximum Transmission Unit (MTU) for the interface
│   │       │   │   │       │   ├── 5 (ifSpeed)       -- Speed of the interface in bits per second
│   │       │   │   │       │   ├── 6 (ifPhysAddress) -- Physical address (MAC address)
│   │       │   │   │       │   ├── 7 (ifAdminStatus) -- Administrative status (up, down, testing)
│   │       │   │   │       │   ├── 8 (ifOperStatus)  -- Operational status (up, down, etc.)
│   │       │   │   │       │   ├── 9 (ifLastChange)  -- Time of last status change
│   │       │   │   │       │   ├── 10 (ifInOctets)   -- Number of octets received on the interface
│   │       │   │   │       │   ├── 11 (ifOutOctets)  -- Number of octets sent on the interface
│   │       │   │   │       │   └── 12 (ifInErrors)   -- Number of input errors on the interface
```

initMAX

# Management information base (MIB) - Tables

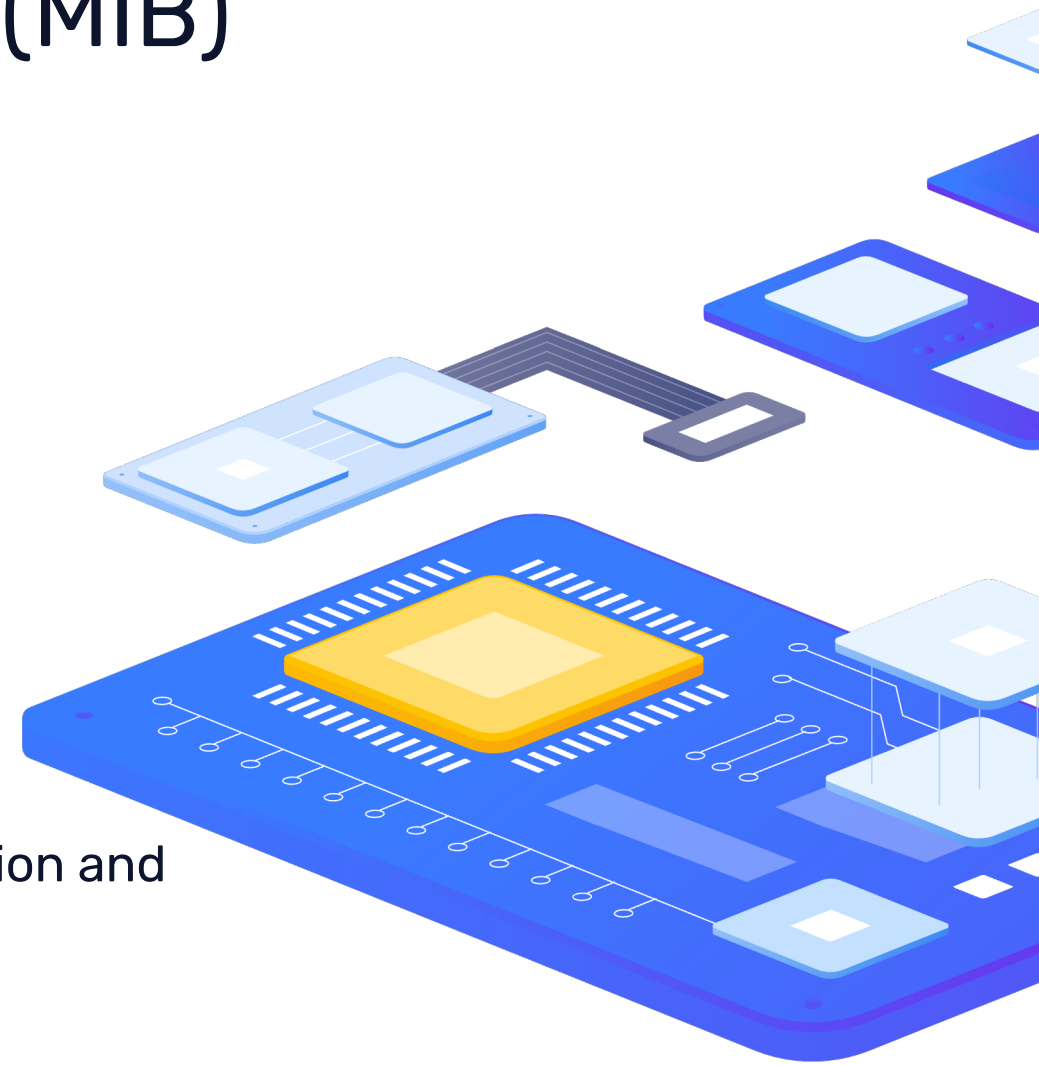
- Table Indexes
  - 1.3.6.1.2.1.2.2.1.2.1
  - IF-MIB::ifDescr.1
- Interface with index 2:
  - 1.3.6.1.2.1.2.2.1.1.2 - IF-MIB::ifIndex.2
  - 1.3.6.1.2.1.2.2.1.2.2 - IF-MIB::ifDescr.2
  - 1.3.6.1.2.1.2.2.1.3.2 - IF-MIB::ifType.2



initMAX

# Management information base (MIB)

- What to use?
- OID
  - .1.3.6.1.2.1.1.1.0
  - It will work everywhere
  - No aditional configuration needed
- Text notation
  - SNMPv2-MIB::sysDescr.0
  - You need to install MIB files – dependent on distribution and SNMP library
  - Better understanding of values



initMAX

# SNMP GET commands

- GET
  - GET - request from SNMP Manager to collect a single value from the device
    GETNEXT - get information from the next OID within the MIB tree
  - GETBULK - pull data tables by using lots of GETNEXT commands
  - WALK - sequentially retrieve values for a range of SNMP objects with GETNEXT
- RESPONSE - the agent sends a GetResponse when replying to a SNMP command

initMAX

# SNMP SET commands

- SET - message sent to the agent to change configurations and issue data
- RESPONSE - the agent sends a RESPONSE when replying to a SNMP command

```
snmpset -v2c -c private 10.1.1.48 SNMPv2-MIB::sysContact.0 = "Spravce windows“

11:21:01.925905 IP (tos 0x0, ttl 64, id 55936, offset 0, flags [DF], proto UDP (17), length 87)
    10.1.1.165.60506 > 10.1.1.48.161:  { SNMPv2c C="private" { SetRequest(43) R=1611696501  .1.3.6.1.2.1.1.4.0="Spravce windows" } }

11:21:01.926728 IP (tos 0x0, ttl 128, id 54551, offset 0, flags [none], proto UDP (17), length 87)
    10.1.1.48.161 > 10.1.1.165.60506:  { SNMPv2c C="private" { GetResponse(43) R=1611696501  .1.3.6.1.2.1.1.4.0="Spravce windows" } }
```

initMAX

# SNMP TRAPs

- TRAP – agent sends Asynchronous trap without confirmation
- Newer specifications of SNMP v2c allows usage of confirmation packets

```
11:43:05.343886 IP (tos 0x0, ttl 64, id 51693, offset 0, flags [DF], proto UDP (17), length 182)
    10.1.1.212.45196 > 10.1.1.165.162:  { SNMPv2c { V2Trap(137) R=85176881  .1.3.6.1.2.1.1.3.0=6710724
.1.3.6.1.6.3.1.1.4.1.0=.1.3.6.1.6.3.1.1.5.3 .1.3.6.1.6.3.18.1.3=00_00_00_00_00_00_00_00_00_00_ff_ff_0a_01_01_d4
.1.3.6.1.2.1.2.2.1.1.4=4 .1.3.6.1.2.1.2.2.1.7.4=1 .1.3.6.1.2.1.2.2.1.8.4=2 } }


11:43:44.068584 IP (tos 0x0, ttl 64, id 53712, offset 0, flags [DF], proto UDP (17), length 182)
    10.1.1.212.50235 > debian12zbx.snmp-trap:  { SNMPv2c { V2Trap(137) R=2064549253  system.sysUpTime.0=6714596
S:1.1.4.1.0=S:1.1.5.3 S:18.1.3=00_00_00_00_00_00_00_00_00_00_ff_ff_0a_01_01_d4 interfaces.ifTable.ifEntry.ifIndex.4=4
interfaces.ifTable.ifEntry.ifAdminStatus.4=1 interfaces.ifTable.ifEntry.ifOperStatus.4=2 } }
```

initMAX

# SNMP security

- Access rights in SNMP v1 and SNMP v2c are managed by defining communities:
  - Community is a type of shared password between the SNMP management and the device
  - Multiple communities can be defined on the same device with different access rights
  - Devices may have a built-in "public" community (which is well-known and insecure)
  - Community is sent over the network unencrypted

initMAX

# SNMPv3 security

- Context name - SNMP security context
- Security name - username
- Security level:
  - noAuthNoPriv - plain text authentication and no data encryption
  - authNoPriv - secure authentication, but no data encryption
  - authPriv - secure authentication and data encryption

initMAX

# SNMPv3 engineId

- SNMP v3 requires a few additional important settings:
- snmpEngineId, which is configured on the device itself, must be unique (RFC 2571)
  - There is a one-to-one association between SNMP engines and SNMP entities
  - Usually, the Engine ID is based on the IANA enterprise number and MAC or IP address of the device
- snmpEngineBoots must be persistent (RFC 3414)
  - Engine boots value is the number of times the SNMP engine has been started or initialized
- snmpEngineTime is used together with snmpEngineBoots for timeliness check
  - Engine time is the number of seconds since the last time the SNMP engine has been "booted"

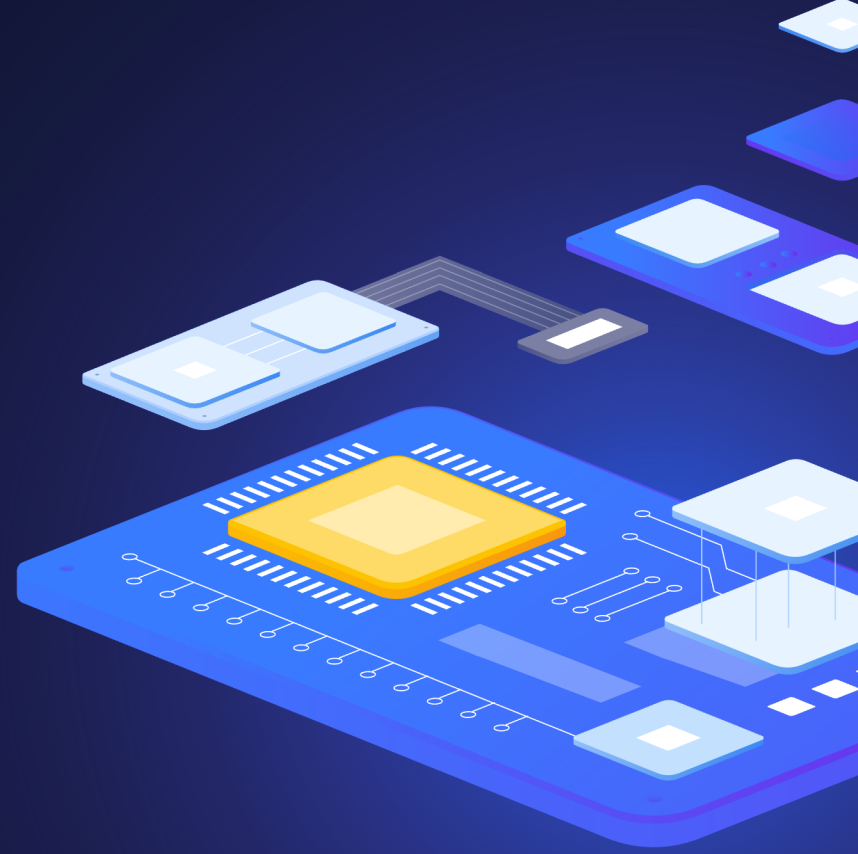
initMAX

# Command line utils

- RPM based distros: net-snmp-utils
- Deb based distros: snmp, snmp-mibs-downloader
  - snmpget
  - snmpgetnext
  - snmpwalk
  - snmpset
  - snmptrap
  - snmpbulkget
  - snmpbulkwalk

# 2

# Zabbix and SNMP

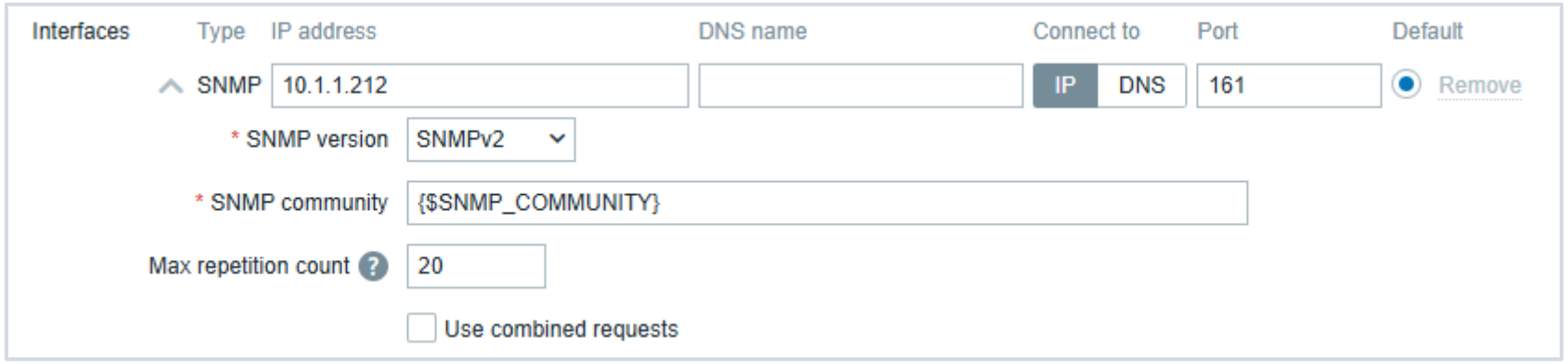


initMAX

# Zabbix SNMP agent - Host

Host SNMP interface

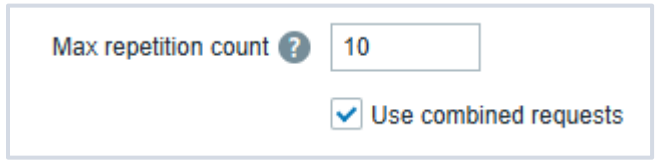
- Multiple Host interfaces
- Multiple SNMP versions: v1, v2c, v3
- Macro usage in interface parameters



initMAX

# Zabbix SNMP agent configuration

Host SNMP interface




- Use combined request:
  - More results in single query with GetBulk
- Max repetition counts:
  - Maximum number of results in single Bulk request

initMAX

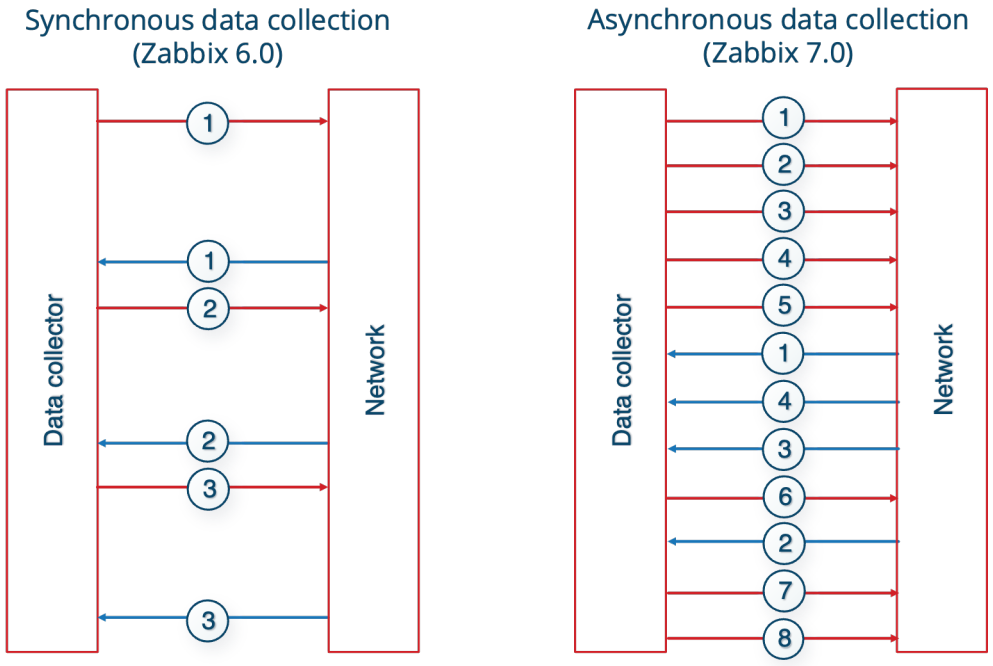
# Zabbix 7.0

- Asynchronous single-OID SNMP requests
  - A new get[OID] SNMP item has been added allowing to query for a single OID value asynchronously.
- Logging of duplicate SNMPv3 Engine IDs
- Engine IDs in SNMPv3 are used as unique identifiers of the device.
- Configurable timeouts per item
  - SNMP agent (only for SNMP walk[OID] and get[OID] items)

initMAX

# Zabbix 7.0 Synchronous x Asynchronous polling

- Asynchronous - snmp poller process
  - get[OID]
  - Walk[OID]
  - snmp poller (for walk[OID] and get[OID] items)
- Synchronous – poller process
  - regular SNMP items
  - SNMP items with dynamic indexes
  - SNMP low-level discovery rules



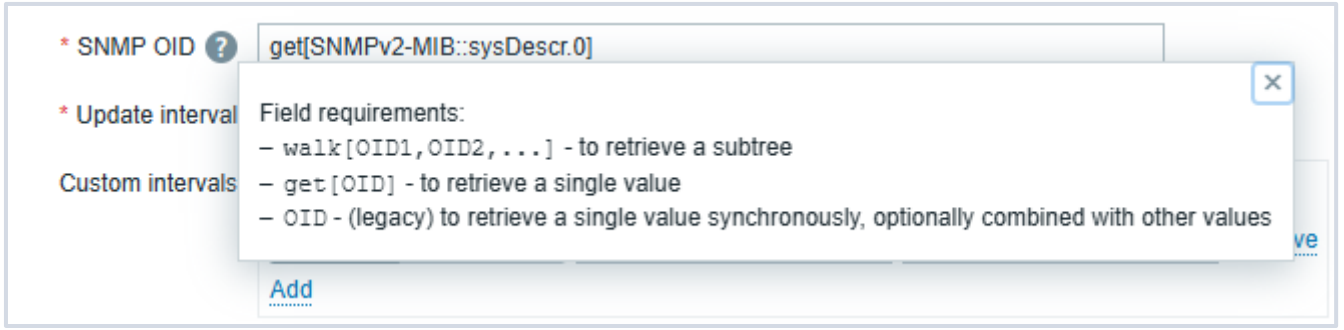
initMAX

# Zabbix 6.0 vs 7.0

initMAX

# Zabbix SNMP agent item key and SNMP OID

- Key
  - Something meaningful
- SNMP OID
  - OID - (legacy) enter a single textual or numeric OID to retrieve a single value synchronously, optionally combined with other values.
  - get[] - retrieve a single value asynchronously.
  - walk[] - retrieve a subtree of values. This option makes use of native SNMP bulk requests (GetBulkRequest-PDUs) asynchronously.

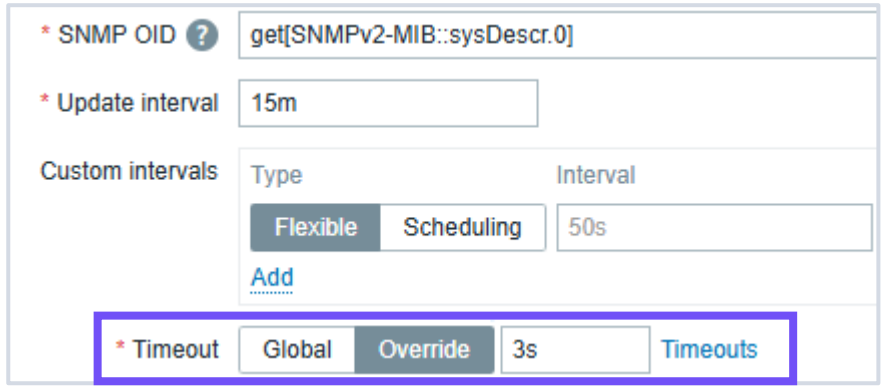


initMAX

# Zabbix SNMP agent timeout

- Zabbix 7.0 will introduce item level timeout for most checks:
  - Timeout is defined using Zabbix graphical user interface
  - Range is from 1 to 600 seconds (10 minutes)

- Timeout can be defined on multiple levels:
  - On Zabbix server globally for all items
  - Per proxy for items monitored by the proxy
  - On each item individually



initMAX

# Timeout in the configuration file

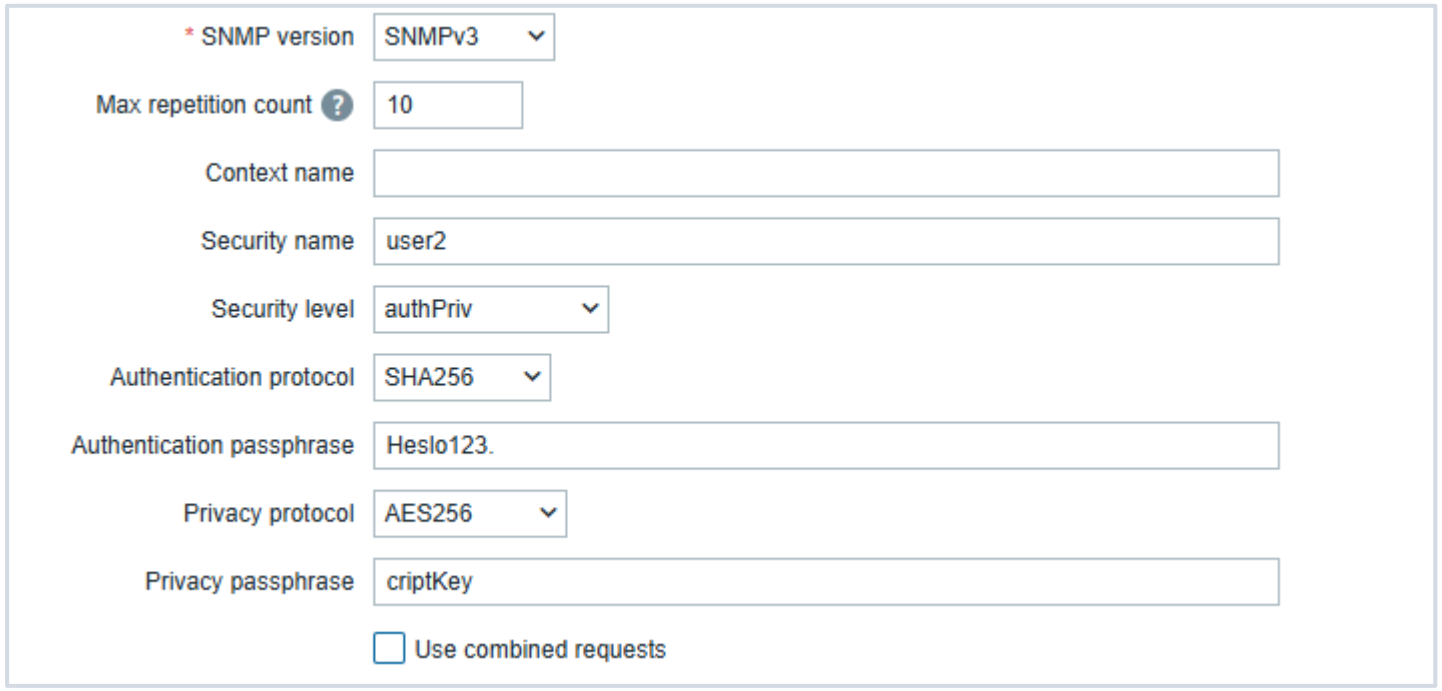
- The timeout setting from Zabbix server / proxy configuration file
  - Will become a default value for item timeout during the upgrade process
  - Will be used as a timeout for communication between server and proxy

```
### Option: Timeout
#       Specifies timeout for communications (in seconds).
# Mandatory: no
# Range: 1-30
# Default:

Timeout=4
```

initMAX

# Zabbix SNMP agent – SNMPv3

- Authentication protocol
- Privacy protocol



initMAX

# SNMPv3 security

- Authentication protocol
  - MD5, SHA1; with net-snmp 5.8 and newer SHA224, SHA256, SHA384 or SHA512.
- Privacy protocol
  - DES, AES128, AES192, AES256, AES192C (Cisco) or AES256C (Cisco).

initMAX

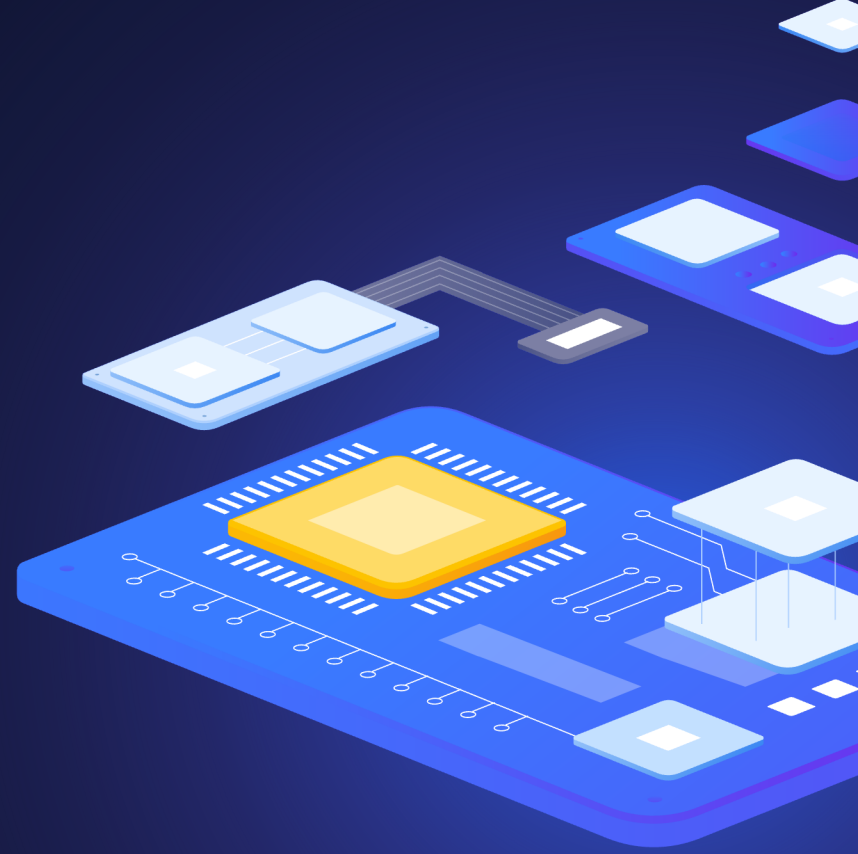
# SNMPv3 security

- Zabbix internally caches SNMPv3 authentication settings:
- The following changes will take effect only after the SNMP cache on a server/proxy is reloaded:
  - Authentication protocol, Authentication passphrase
  - Privacy protocol, Privacy passphrase
- In case the "Security name" is also changed, all parameters will be updated immediately
- There are multiple ways to clear SNMPv3 cache on Zabbix server/proxy
  - Use Zabbix command-line option -R snmp_cache_reload (recommended
  - Restart Zabbix server/proxy service
  - Move the monitored host to a different proxy

```
# zabbix_server -R snmp_cache_reload
zabbix_server [187522]: command sent successfully
```

# 3

# SNMP dynamic indexes



initMAX

# Zabbix SNMP - dynamic indexes

- While you may find the required index number (for example, of a network interface) among the SNMP OIDs, sometimes you may not completely rely on the index number always staying the same.
- Index numbers may be dynamic - they may change over time and your item may stop working as a consequence.

- https://www.zabbix.com/documentation/7.0/en/manual/config/items/itemtypes/snmp/dynamicindex

initMAX

# Zabbix SNMP - dynamic indexes

- A special syntax for OID is used:

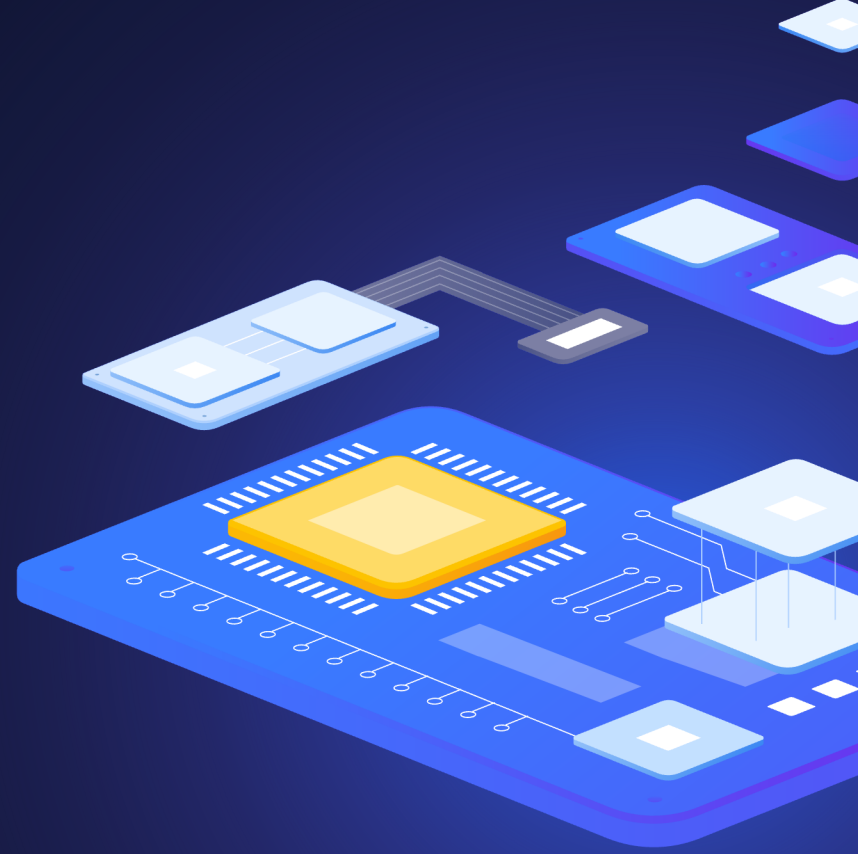
```
<OID of data>["index","<base OID of index>","<string to search for>"]
```

- Example:
- Getting status of Zabbix Agent2 windows service:

```
HOST-RESOURCES-MIB::hrSWRunStatus["index","HOST-RESOURCES-MIB::hrSWRunName","zabbix_agent2.exe"]
```

# 4

# SNMP LLD



initMAX

# LLD - Legacy

Legacy:

- Discovery[] -> SNMP Item prototypes

| | |
|---|---|
| * Name | Network interfaces discovery |
| Type | SNMP agent |
| * Key | net.if.discovery |
| * SNMP OID | discovery[{#IFOPERSTATUS},1.3.6.1.2.1.2.2.1.8,{#IFADMINSTATUS},1.3.6.1.2.1.2 |
| * Update interval | 1h |

## Item prototypes

All templates / Webinar - Legacy Discovery  Discovery list / Network interfaces discovery  Item prototypes 9  Trigger prototypes 4  Graph prototypes 1  Host prototypes

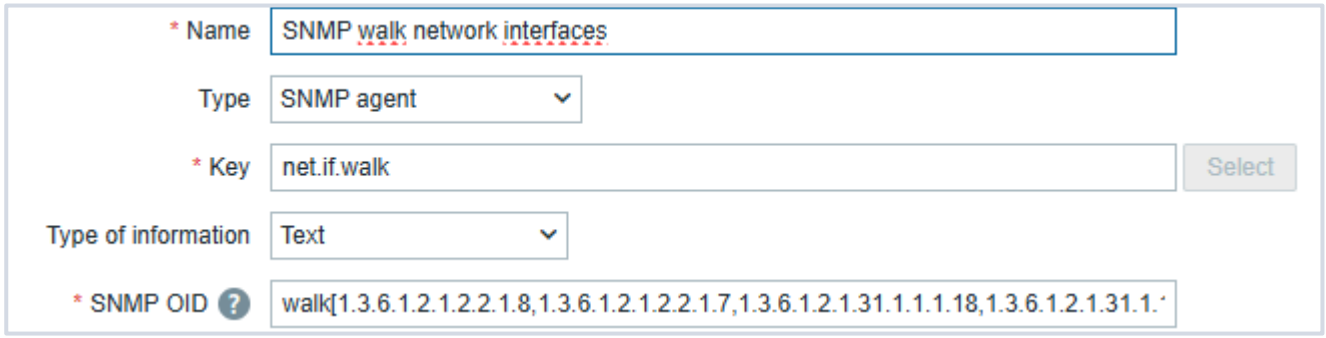
| Name ▲ | Key | Interval | History | Trends | Type |
|---|---|---|---|---|---|
| Interface {#IFDESCR}: Bits received | net.if.in[ifInOctets.{#SNMPINDEX}] | 3m | 7d | 365d | SNMP agent |
| Interface {#IFDESCR}: Bits sent | net.if.out[ifOutOctets.{#SNMPINDEX}] | 3m | 7d | 365d | SNMP agent |
| Interface {#IFDESCR}: Inbound packets discarded | net.if.in.discards[ifInDiscards.{#SNMPINDEX}] | 3m | 7d | 365d | SNMP agent |
| Interface {#IFDESCR}: Inbound packets with errors | net.if.in.errors[ifInErrors.{#SNMPINDEX}] | 3m | 7d | 365d | SNMP agent |

initMAX

# LLD - Modern

Modern:

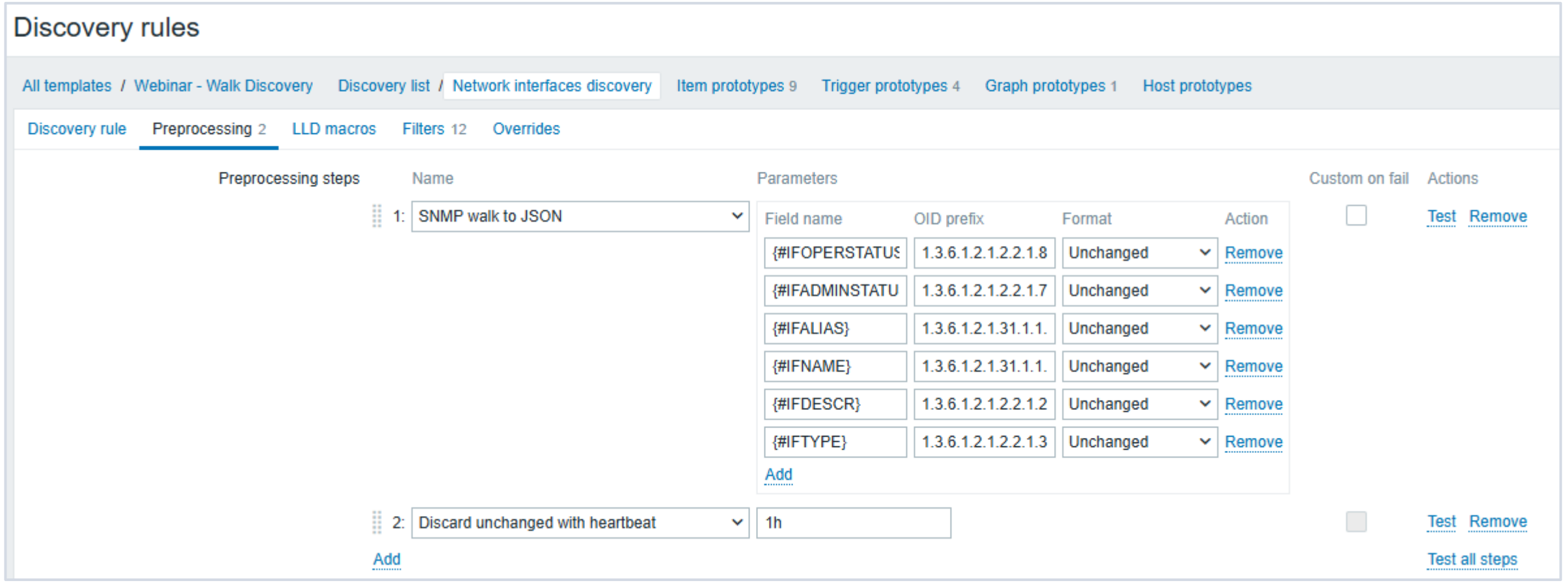
- Walk[] item -> dependent Discovery -> dependent Item prototypes




initMAX

# LLD - Modern

- Discovery preprocessing



initMAX

# LLD - Modern

- Item prototype preprocessing

Item prototypes

All templates / Webinar - Walk Discovery  Discovery list / Network interfaces discovery  Item prototypes 9  Trigger prototypes 4  Graph prototypes 1  Host prototypes

| Name ▲ | Key | Interval | History | Trends | Type |
|---|---|---|---|---|---|
| SNMP walk network interfaces: Interface {#IFNAME}({#IFALIAS}): Bits received | net.if.in[ifHCInOctets.{#SNMPINDEX}] | | 31d | 365d | Dependent item |
| SNMP walk network interfaces: Interface {#IFNAME}({#IFALIAS}): Bits sent | net.if.out[ifHCOutOctets.{#SNMPINDEX}] | | 31d | 365d | Dependent item |
| SNMP walk network interfaces: Interface {#IFNAME}({#IFALIAS}): Inbound packets discarded | net.if.in.discards[ifInDiscards.{#SNMPINDEX}] | | 31d | 365d | Dependent item |
| SNMP walk network interfaces: Interface {#IFNAME}({#IFALIAS}): Inbound packets with errors | net.if.in.errors[ifInErrors.{#SNMPINDEX}] | | 31d | 365d | Dependent item |

Item prototype

Item prototype  Tags 3  Preprocessing 3

Preprocessing steps

| | Name | Parameters | |
|---|---|---|---|
| 1: | SNMP walk value | 1.3.6.1.2.1.31.1.1.1.6.{# | Unchanged |
| 2: | Change per second | | |
| 3: | Custom multiplier | 8 | |

initMAX

# Legacy x Modern

Hybrid setup:

- Walk[] item -> dependent Discovery -> dependent Item prototypes
- Walk[] item -> dependent Discovery -> get[] Item prototypes
- Walk[] item1 -> dependent Discovery ->
  - Walk[] item2 -> dependent Item prototypes
- And combination of both..

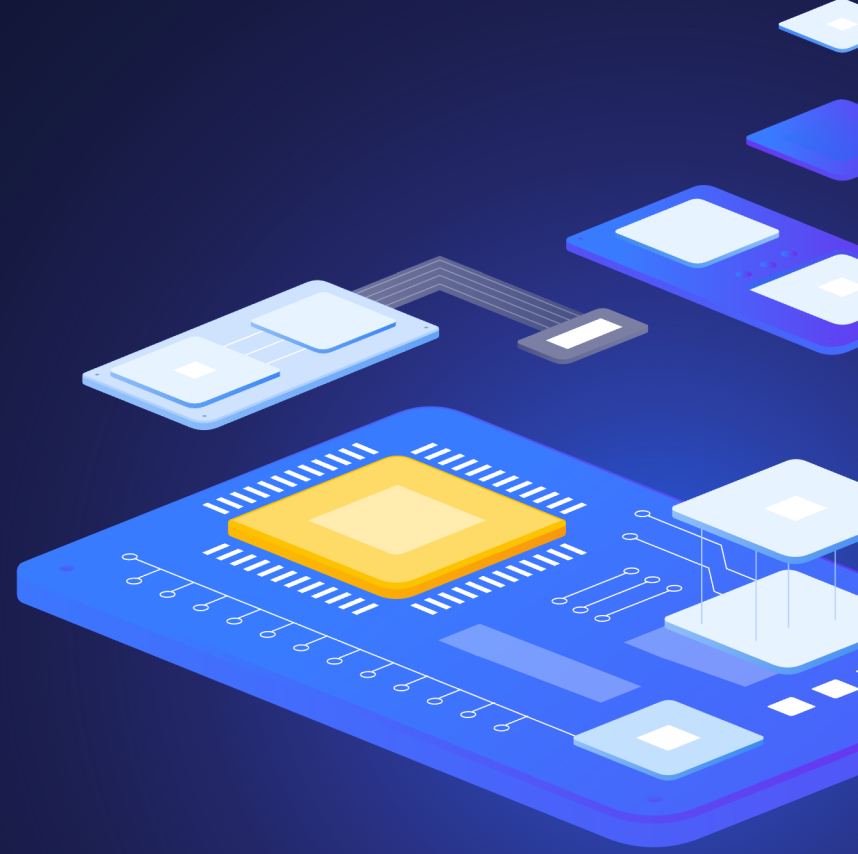
initMAX

# walk[] with multiple table indexes

- Joining Tables together with Javascript preprocessing
- Walk[] item -> dependent Discovery -> dependent Item prototypes

5

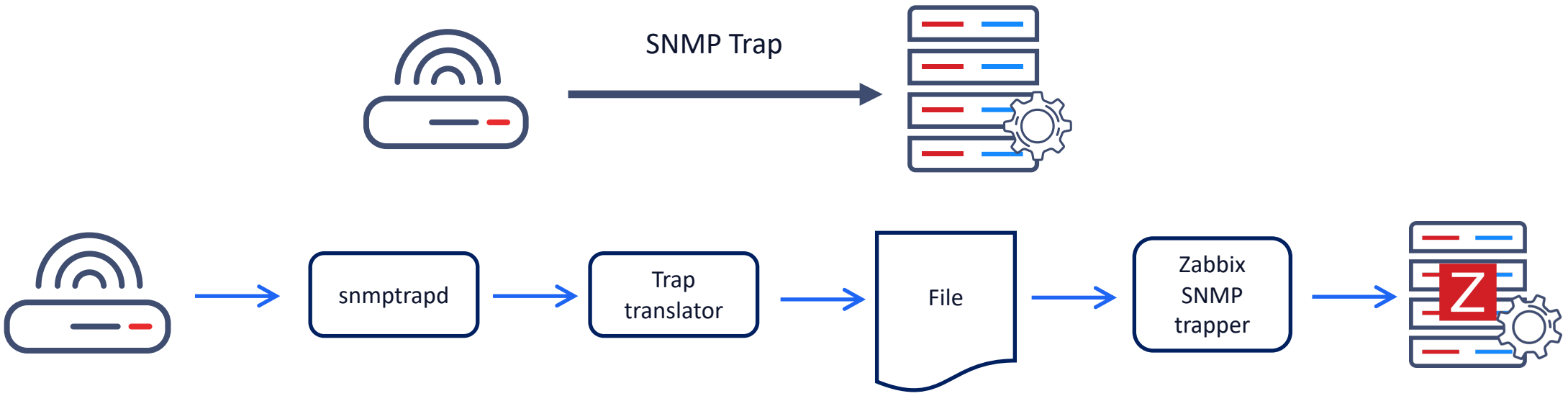
# SNMP traps



initMAX

# Trap flow

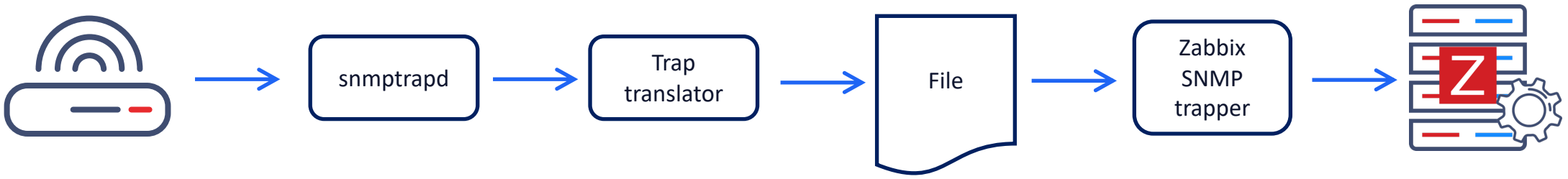
- Asynchronous notification from agent to manager. While in other SNMP communication, the manager actively requests information from the agent, these are PDUs that are sent from the agent to the manager without being explicitly requested.



initMAX

# Trap flow

- snmptrapd daemon receives a trap and passes the trap to the trap receiver ( translator )
- The trap receiver parses, formats, and writes the trap to a file
- Zabbix SNMP trapper process reads and parses the trap file
- Zabbix checks all SNMP trap items with an SNMP interface address matching the trap address
- If the address cannot be matched with any host, the trap is logged in the Zabbix server log file



initMAX

# SNMP trap receiver Installation

- EN https://www.initmax.com/wiki/how-to-set-up-snmp-trap-in-zabbix/
- CZ https://www.initmax.cz/wiki/jak-nastavit-snmp-trap-v-zabbixu/

```
apt install snmptrapd snmp
curl -o /usr/bin/zabbix_trap_receiver.pl
https://git.zabbix.com/projects/ZBX/repos/zabbix/raw/misc/snmptrap/zabbix_trap_receiver.pl
chmod +x /usr/bin/zabbix_trap_receiver.pl
vi /usr/bin/zabbix_trap_receiver.pl
mkdir /var/log/snmptrap
chown Debian-snmp /var/log/snmptrap
vi /etc/snmp/snmptrapd.conf
# change setup
vi /etc/logrotate.d/snmptrap
# insert logrotate settings
systemctl restart snmptrapd
systemctl enable snmptrapd
vi /etc/zabbix/zabbix_server.conf
# enable trap configuration
```

initMAX

# SNMP trap receiver Installation

- SNMP MIB files in debian
- Install package: **snmp-mibs-downloader**

```
# change sources-list – non-free
vi /etc/apt/sources.list
apt install snmp-mibs-downloader
# comment out line: mibs :
vi /etc/snmp/snmp.conf
```

initMAX

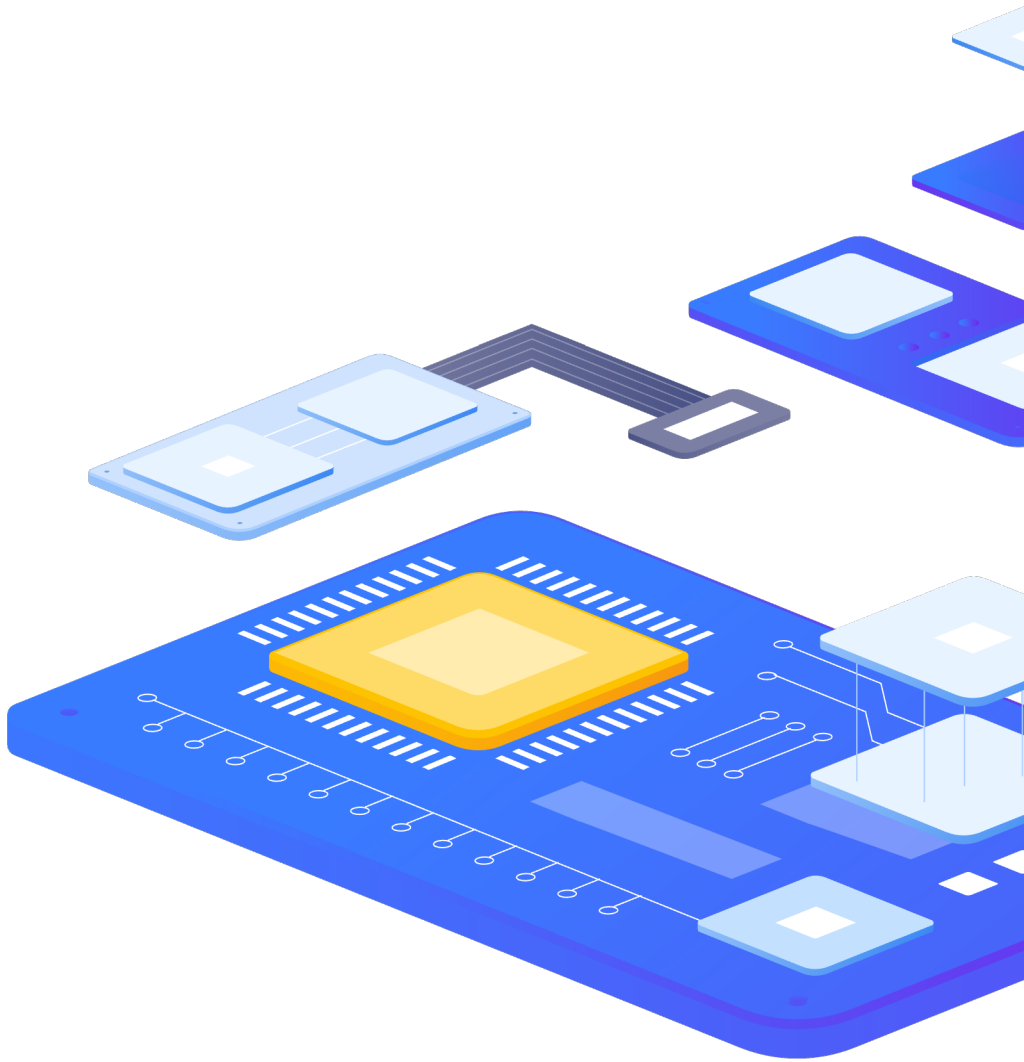
# Zabbix 7.0

- SNMP traps with Zabbix high availability
  - Zabbix now can read SNMP trap files from the correct place in case the active node is switched in a high-availability setup.
  - However, for this functionality to work it is required to update the time format in any bash, perl and SNMPTT scripts to "%Y-%m-%dT%H:%M:%S%z" (i.e. 2024-01-10T11:56:14+0300).

initMAX

# SNMP trap Items in Zabbix

- Zabbix Item Type:
  - SNMP trap
- Keys:
  - snmptrap[<regex>]
  - snmptrap.fallback



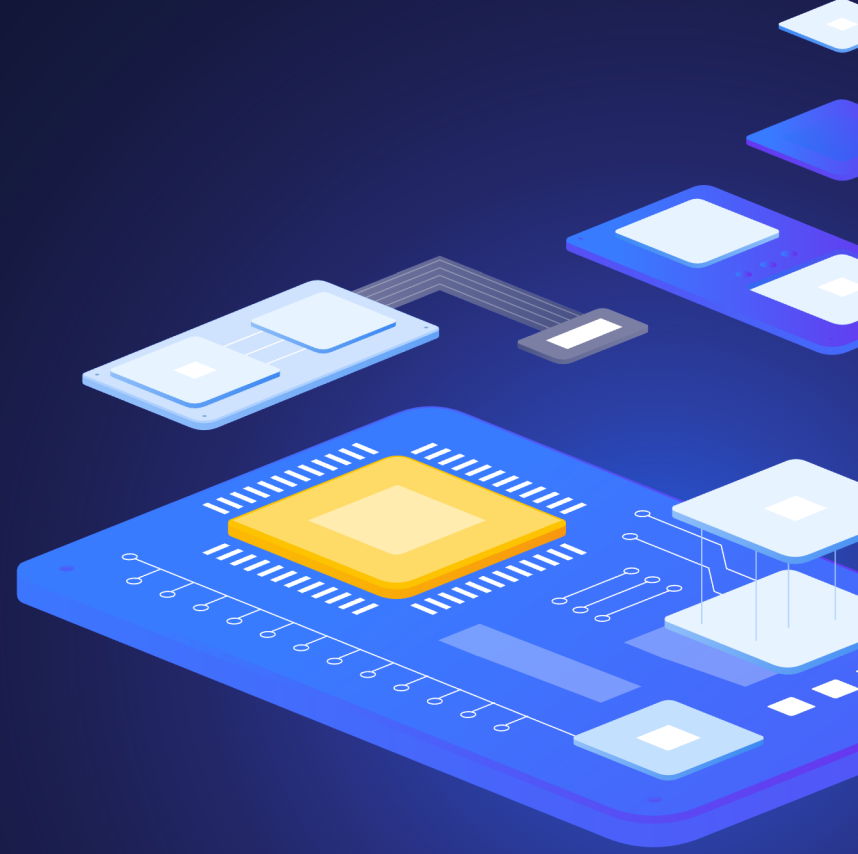
initMAX

# SNMP trap Items in Zabbix

- Templates
- Multiple Items -> Single Triggers per Item
- Single Item -> multiple event generation mode Trigger
  - Event corellation based on Tag value

# 6

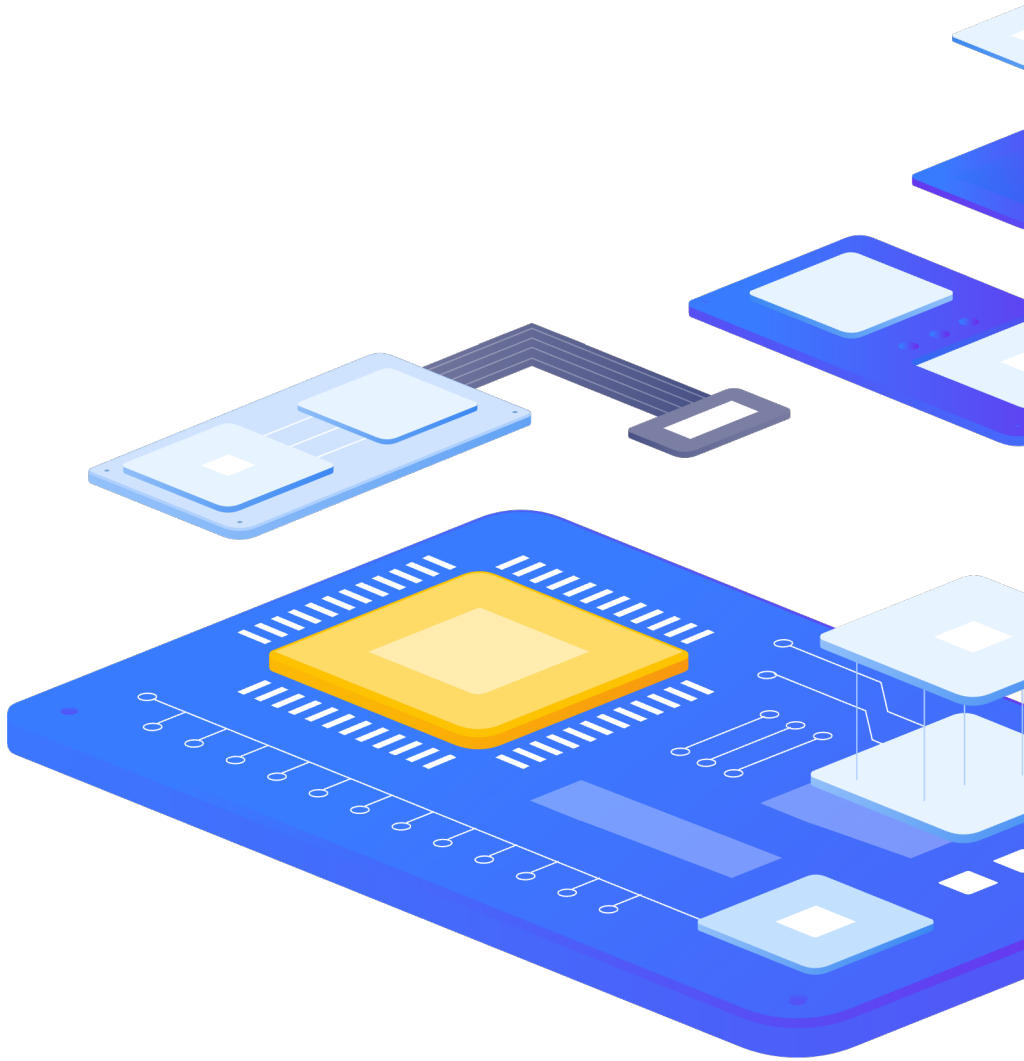
# SNMP templates



initMAX

# Out-Of-The-Box Templates

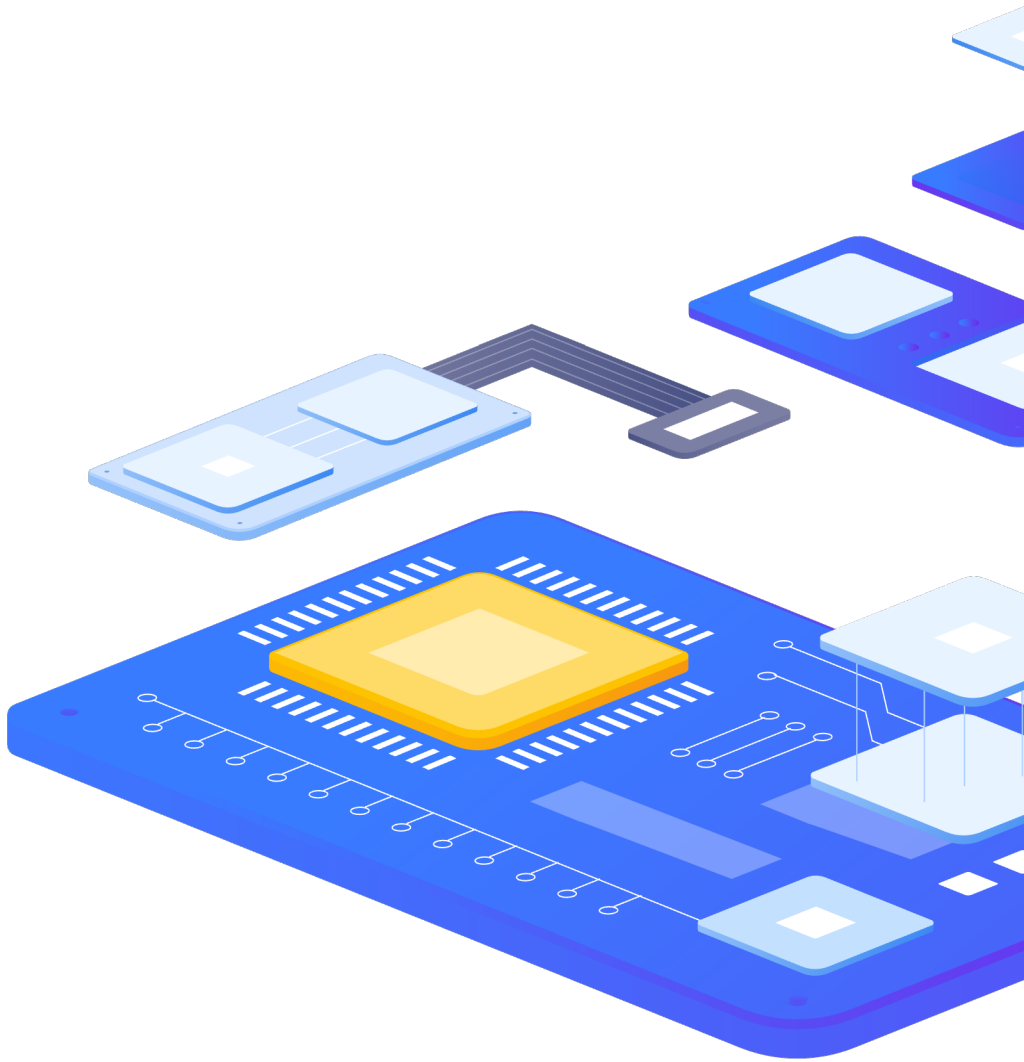
- Network devices
  - Cisco, Mikrotik, D-Link, F5, HP, …
- Hardware devices
  - APC, Dell, HPE, NetApp, …
- OS
  - Windows, Linux



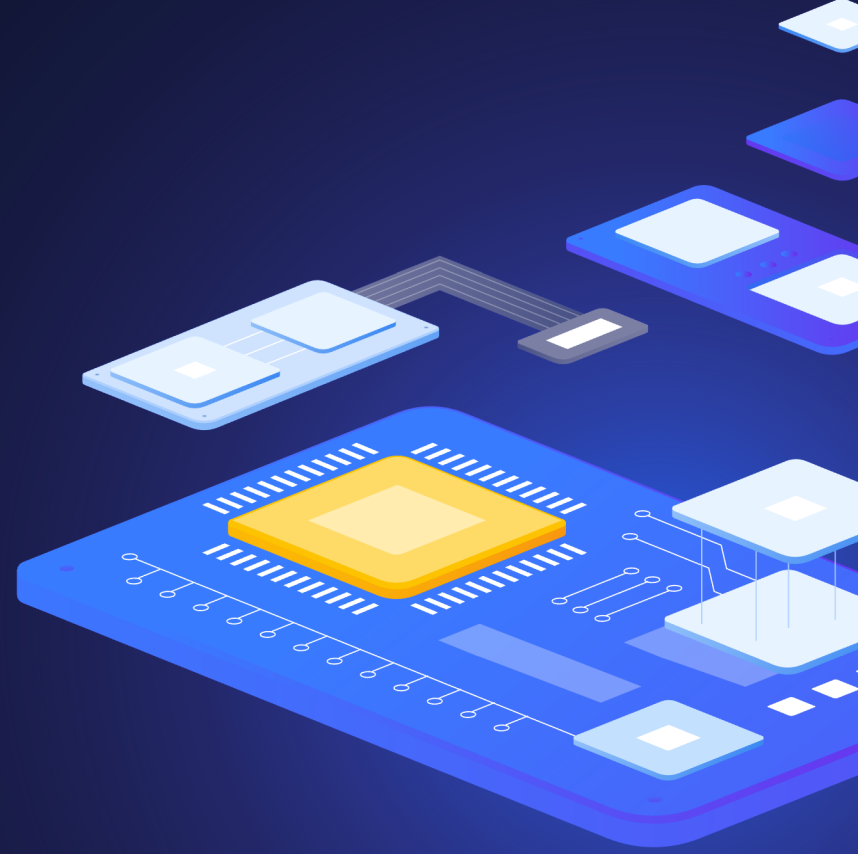
initMAX

# Custom Template Hints

- Use Modern walk[], get[] type
- Be carefull with community templates
- Convert Synchronous checks to Asynchronous



# Questions?



initMAX

# Contact us:


| | |
|---|---|
| Phone: | +420 800 244 442 |
| Web: | https://www.initmax.cz |
| Email: | tomas.hermanek@initmax.cz |
| LinkedIn: | https://www.linkedin.com/company/initmax |
| Twitter: | https://twitter.com/initmax |
| Tomáš Heřmánek: | +420 732 447 184 |