ZABBIX

Audit

Artjoms Rimdjonoks

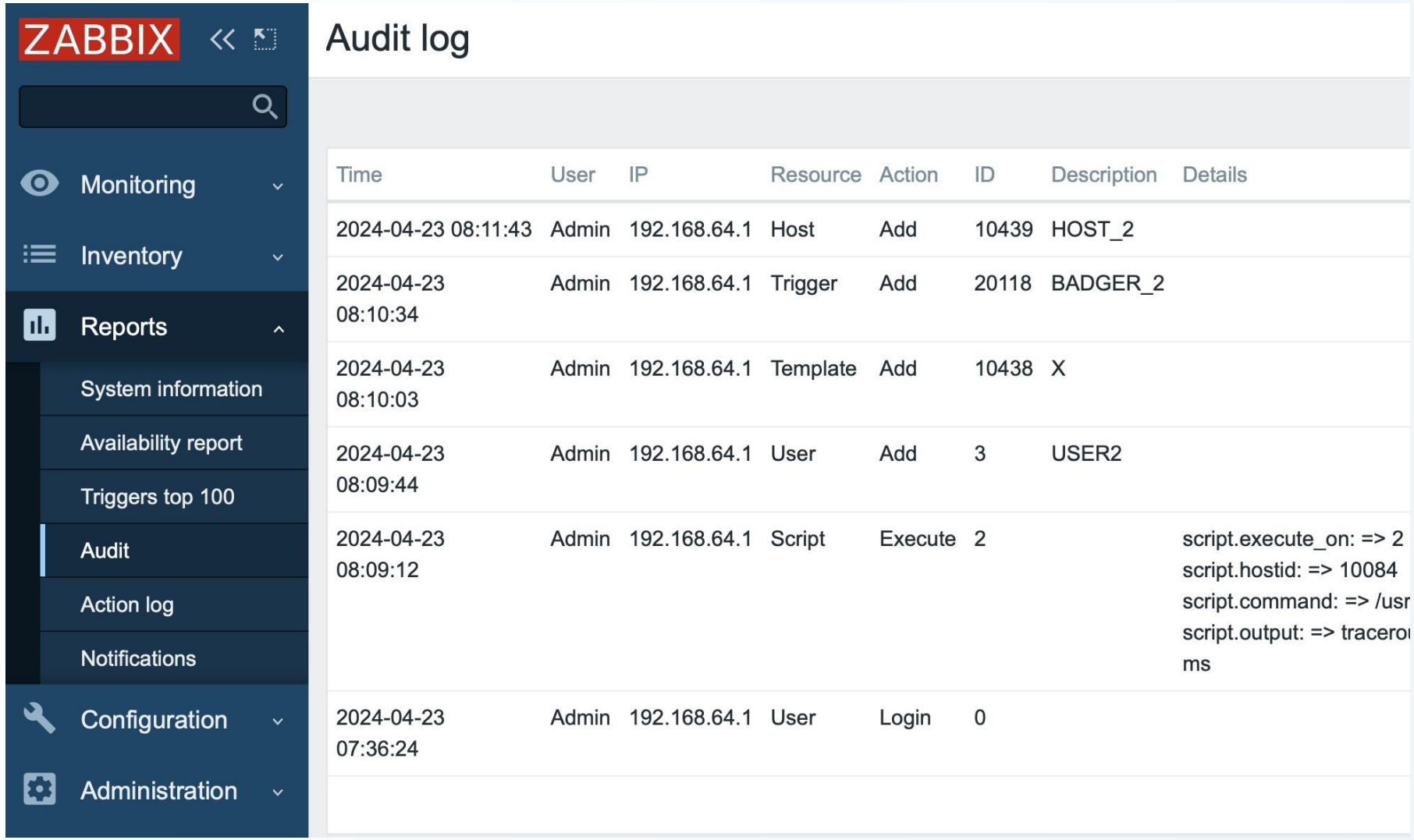
C Developer

Overview

In 5.0 there is already an Audit

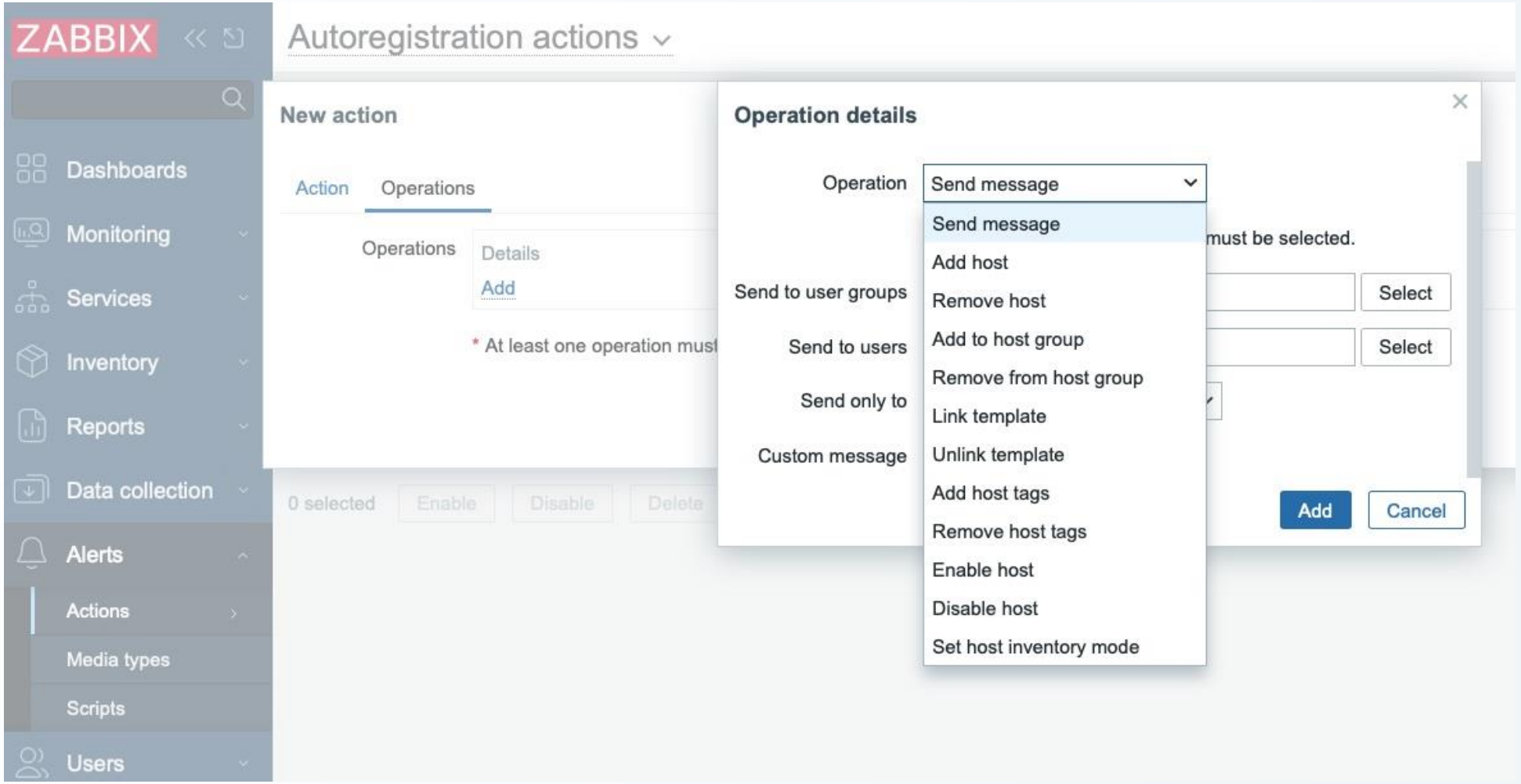
- ZBXNEXT-6470 improves it
- Goal - to audit all configuration and settings changes.
- Who, when and what.
- Enterprise-level requirement.
- Front-end development ongoing ....
- Server side is mostly done in 6.0
- 7.0 has improvements

Before 6.0…

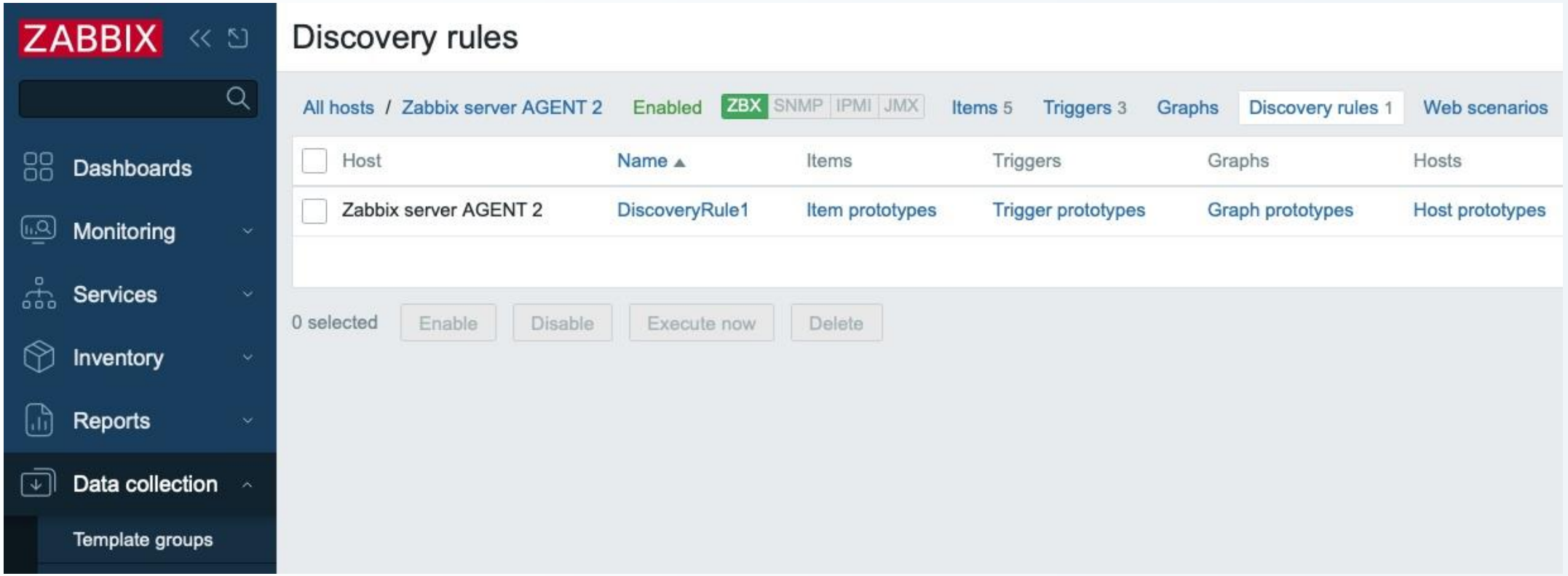


- Zabbix Server may do a lot of configuration yet there is no any audit of that. For example, if host is created on server – nothing is recorded.
- Adding of template on a host – is audited, but no info of items, triggers, tags etc. is present.

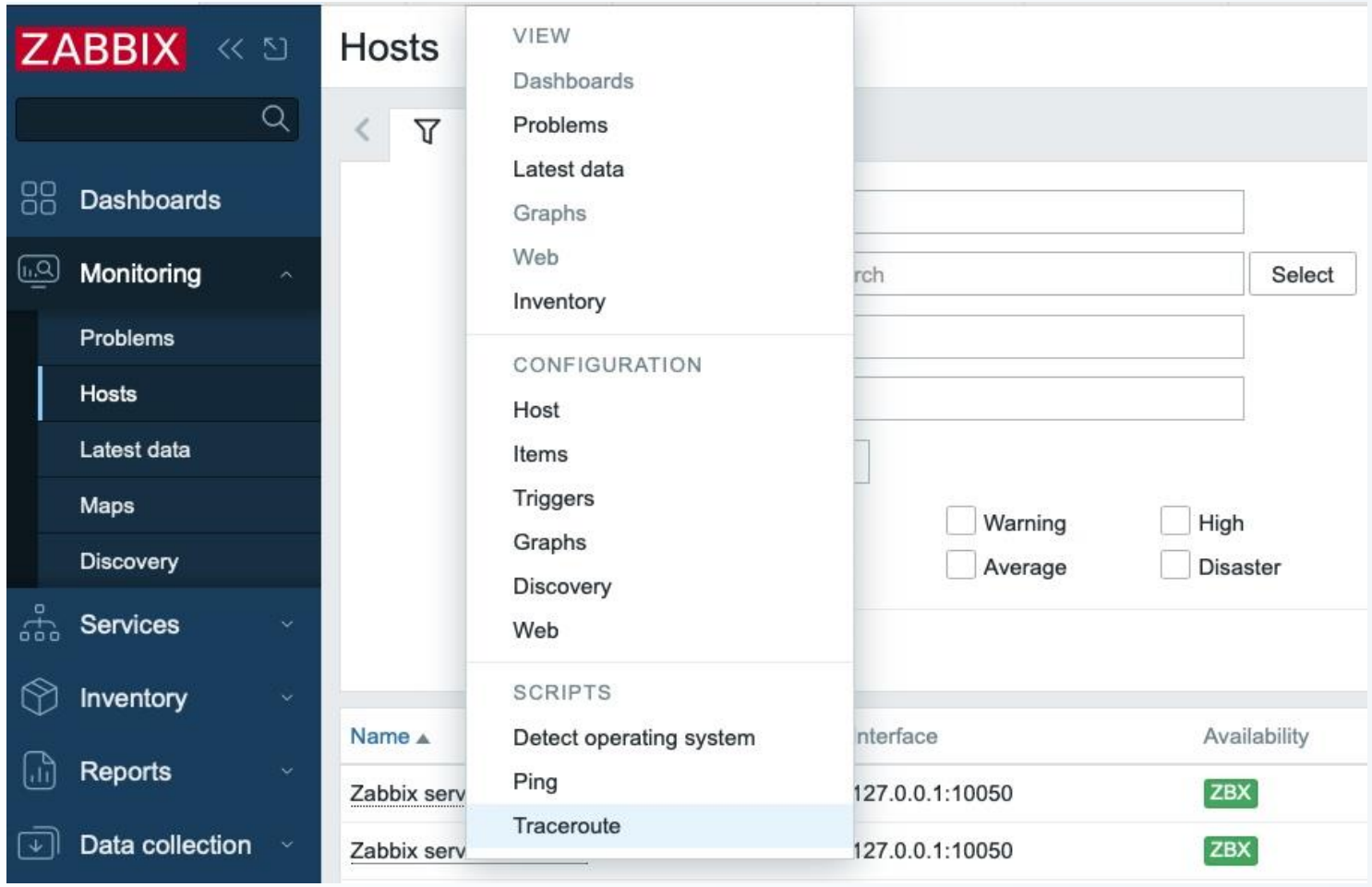
Autoregistration and Network Discovery



LLD

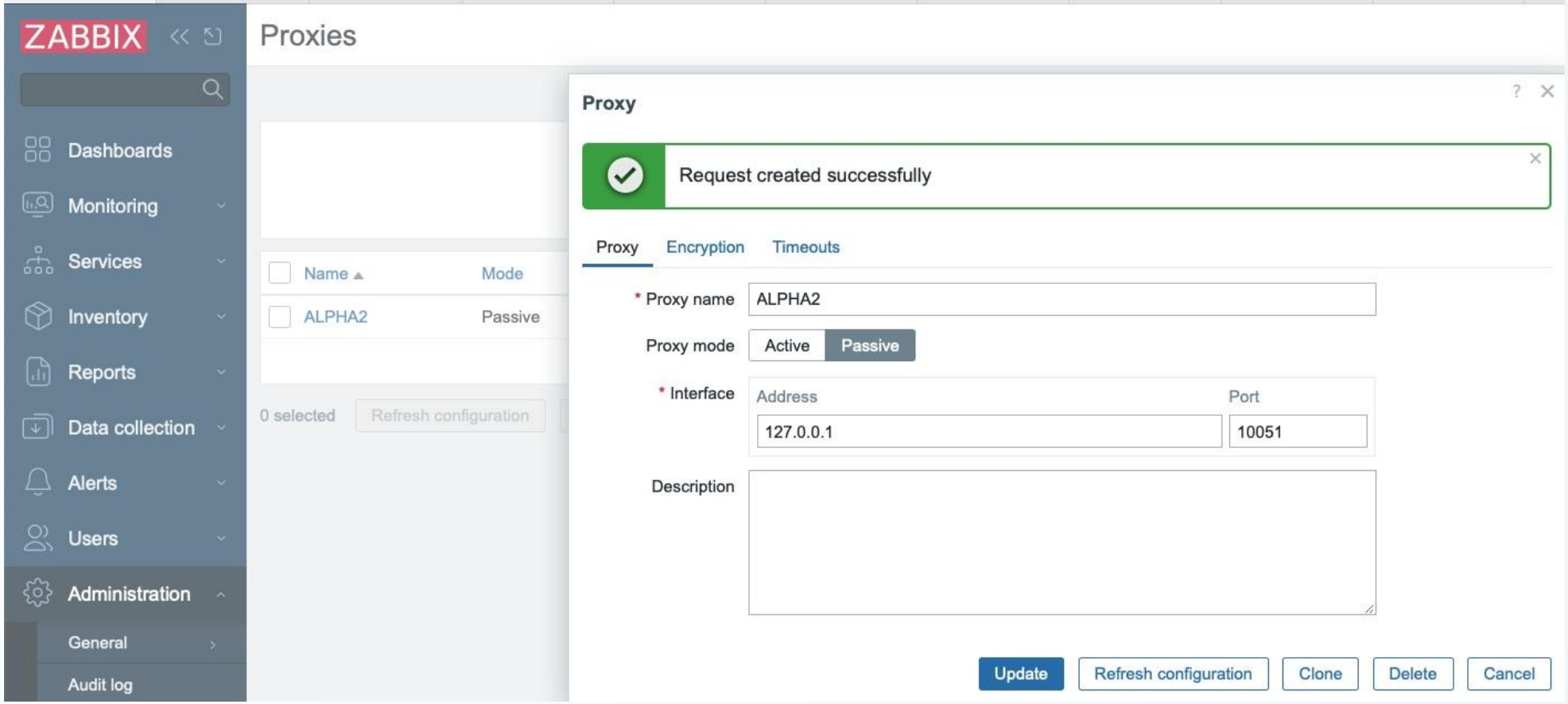

ZABBIX
Dashboards
Monitoring
Services
Inventory
Reports
Data collection
Template groups
Discovery rules
All hosts / Zabbix server AGENT 2
Enabled
ZBX
SNMP
IPMI
JMX
Items 5
Triggers 3
Graphs
Discovery rules 1
Web scenarios
Host
Name ▲
Items
Triggers
Graphs
Hosts
Zabbix server AGENT 2
DiscoveryRule1
Item prototypes
Trigger prototypes
Graph prototypes
Host prototypes
0 selected
Enable
Disable
Execute now
Delete

Script execution (non-configuration)



Tasks reload (non-configuration)

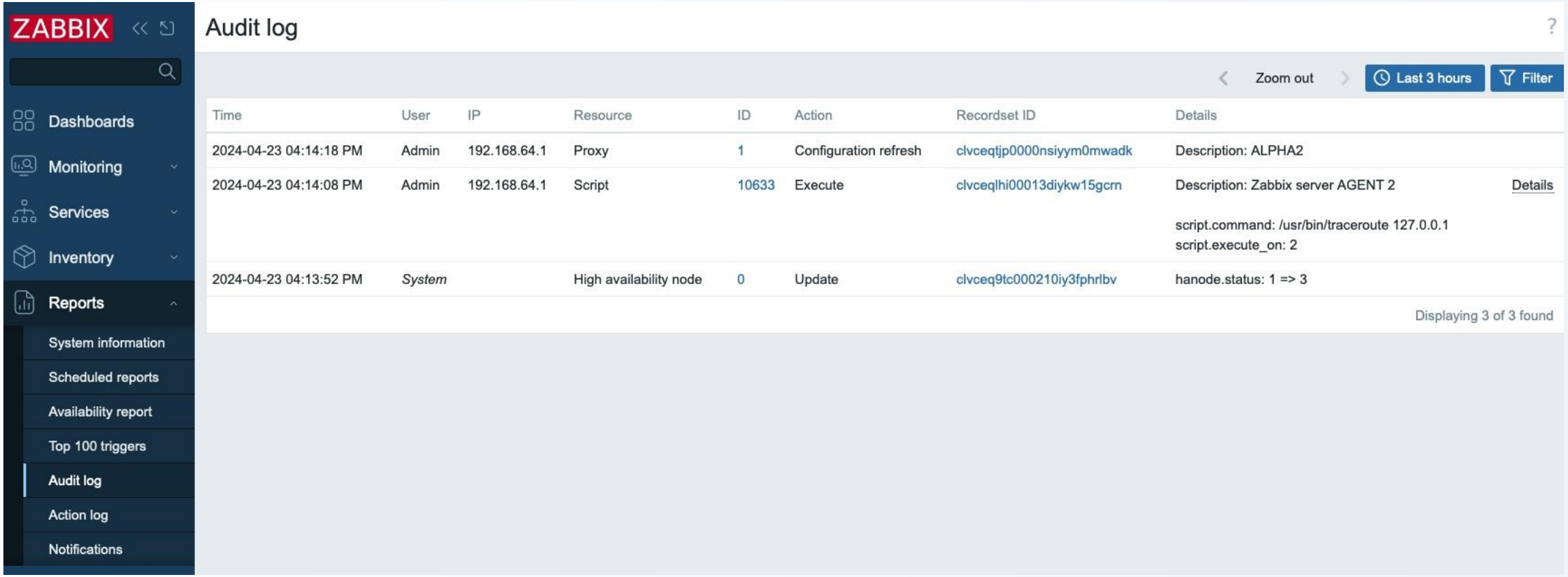
reloading passive proxy config data (ZBXNEXT-1580), added in 6.2



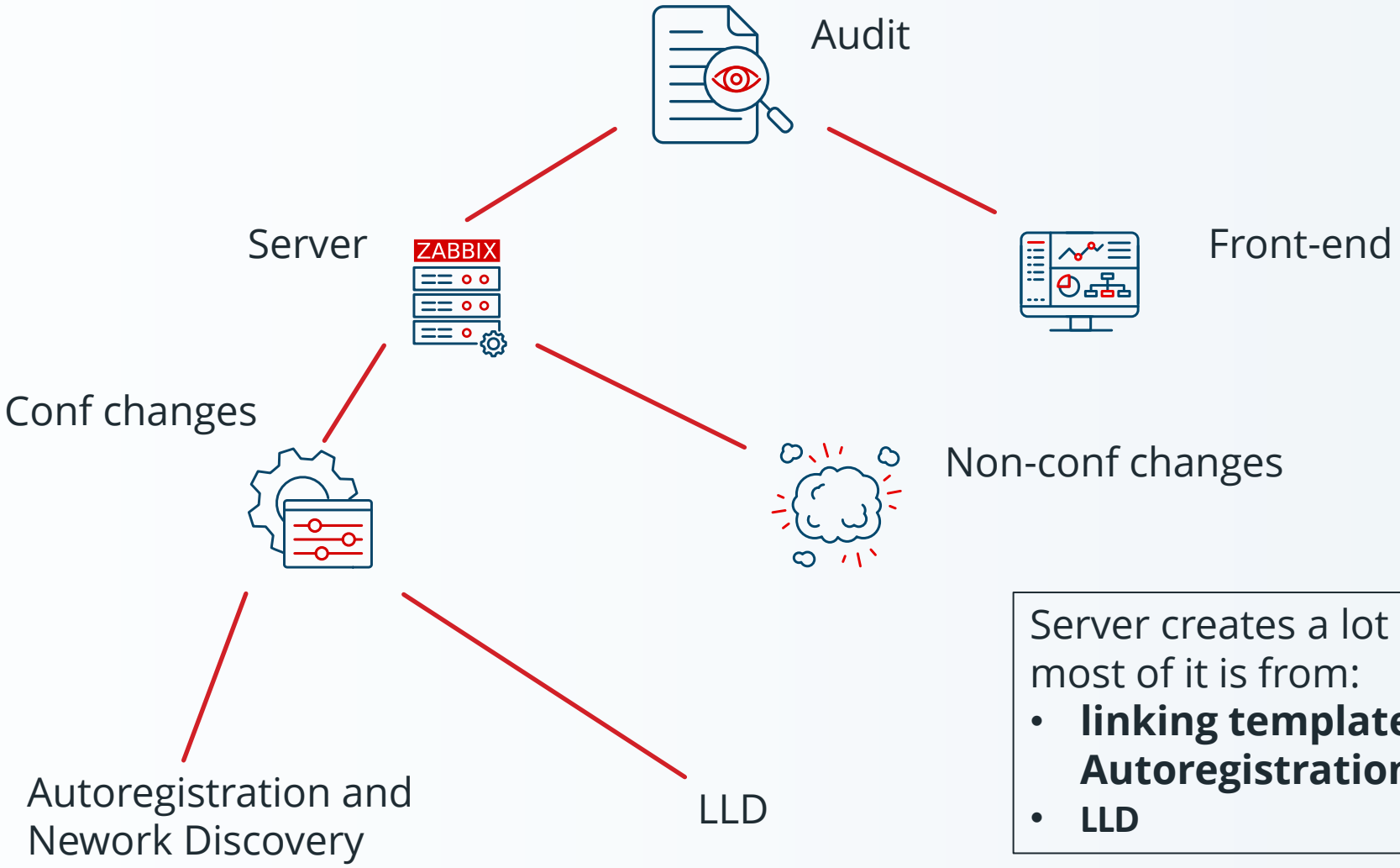
Other non-configuration...

HA node status changes (ZBXNEXT-6923), added in 6.0

History push API request, sending data to Zabbix server via API (ZBXNEXT-8541), added in 7.0



New Audit Scope Summary



Server creates a lot of configuration changes, most of it is from:

- **linking templates** during **Autoregistration** and **Network Discovery**
- **LLD**

New Audit Server Scope Summary

Most Zabbix server audit logic is in:

a) Linking of templates (as a result of **Autoregistration** or **Network Discovery**) with updates to:

- Hosts
- Items
- Triggers
- Graphs
- Discovery Rules (and prototypes of everything above)
- Web Scenarios

2) LLD, with the following entities created from prototypes:

- Hosts
- Items
- Triggers
- Graphs

New Audit Goals

ZABBIX

- Simple to manage and fast.
- All Audit is stored in single table (Simpler and faster SQL queries).
- Bulk SQL inserts and efficient ids generation.

- Audit of particular entity stays longer than this entity. If entity - (host or user) is deleted – audit for it stays. Audit has independent housekeeping schedule.
- Can be disabled.

IDs for New Audit

Ids table:

```
zabbix=> select * from ids;
   table_name    |   field_name    | nextid
-----------------+-----------------+--------
 actions         | actionid        |      7
 operations      | operationid     |     12
 optemplate      | optemplateid    |      3
 module          | moduleid        |     29
 profiles        | profileid       |     65
 housekeeper     | housekeeperid   |   1154
 hosts           | hostid          |  10632
 interface       | interfaceid     |     35
 hosts_groups    | hostgroupid     |    637
 hgset           | hgsetid         |     17
 hosts_templates | hosttemplateid  |    460
 items           | itemid          |  47084
 triggers        | triggerid       |  23685
 functions       | functionid      |  33628
(14 rows)
```

New audit could use ids table, but..

IDs for new Audit

CUID

Collision resistant id for horizontal scaling

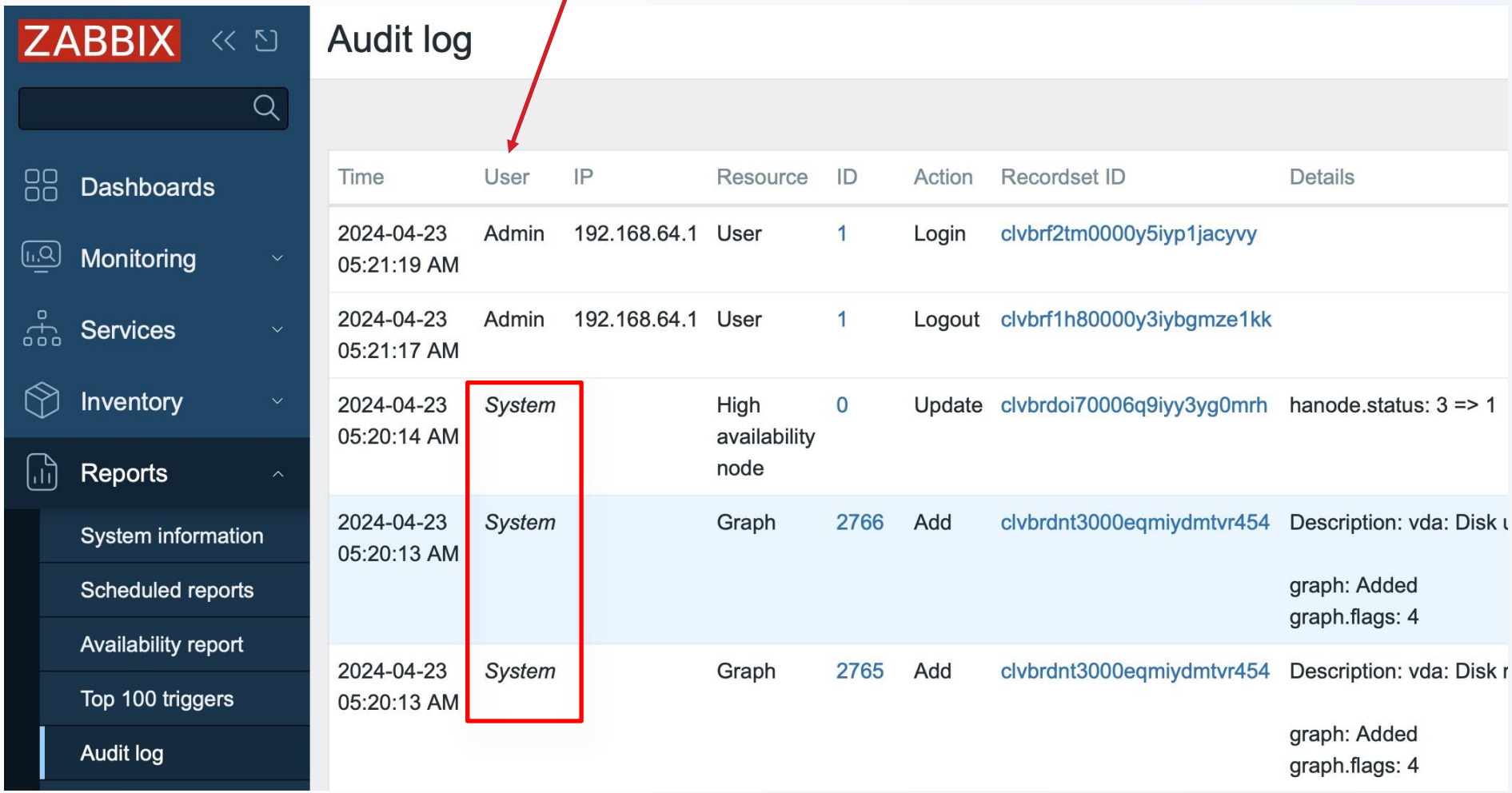
clvc7m4ik0009e9iy2t4dpmja

c - lvc7m4ik - 0009 - e9iy - 2t4dpmja

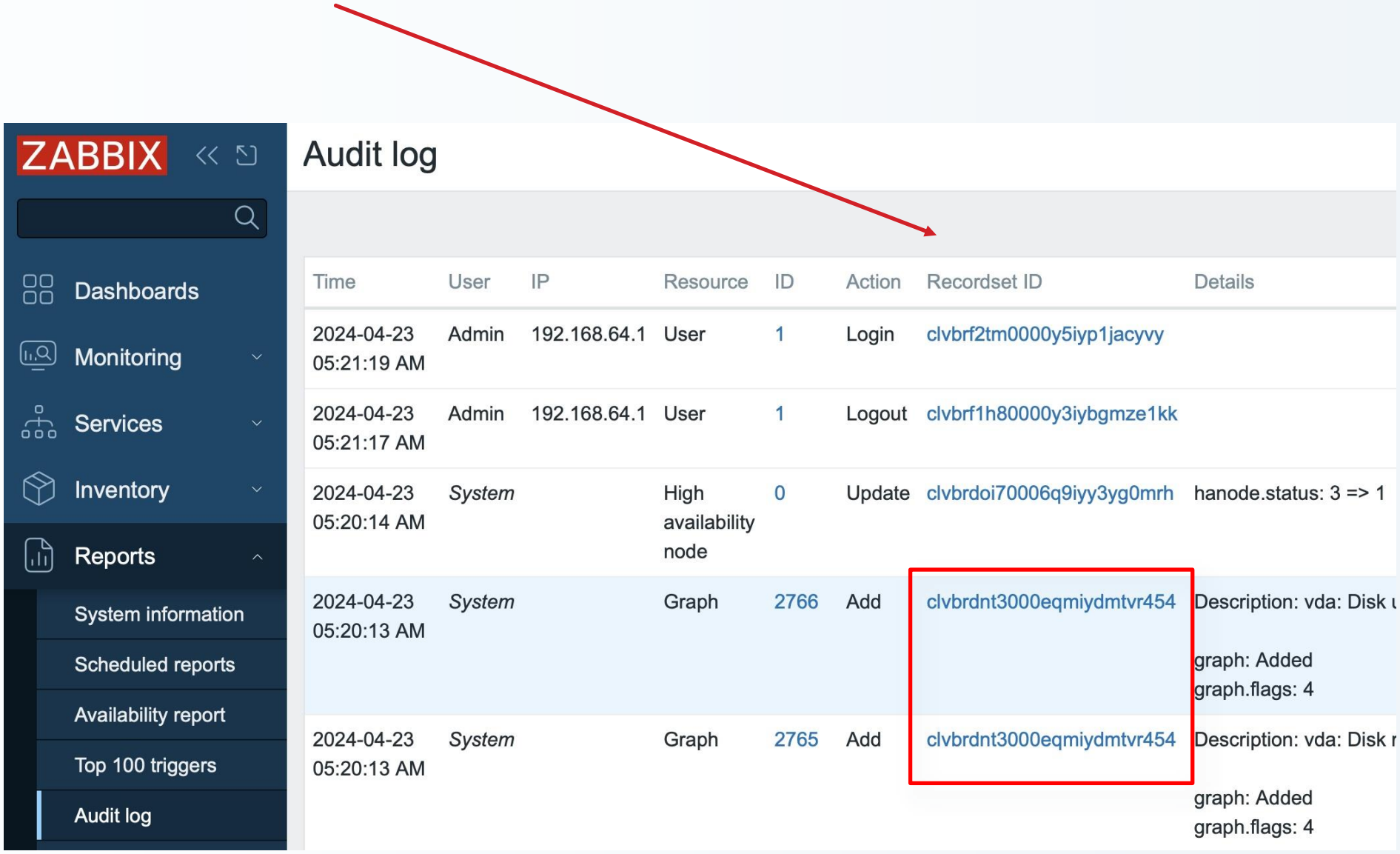
timestamp - counter - client fingerprint - random string

UI

New 'System' User



UI Recordset ID



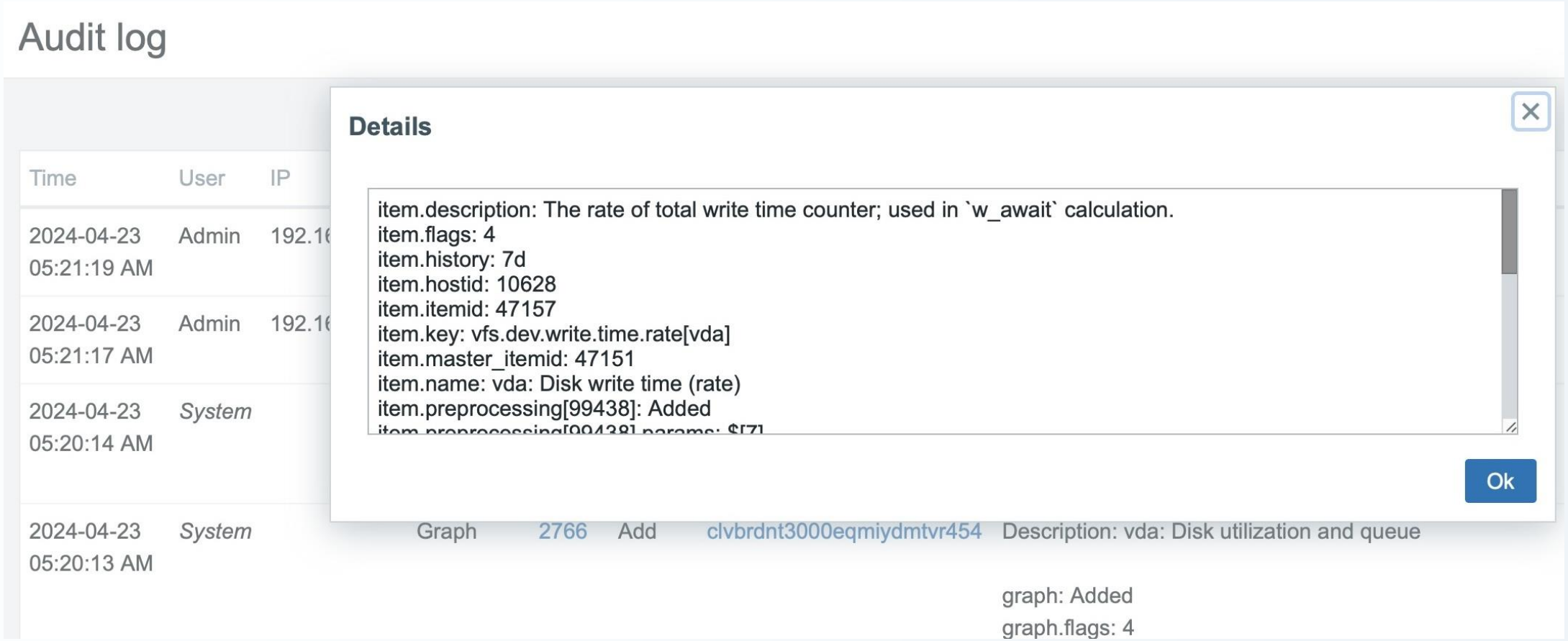
Recordset ID

From the spec:

“To have the ability to recognize that some set of audit log records was created during the processing of separate operation, a new column "Recordset ID" for audit log records will be provided. Each audit log record of separate operation will have the same recordset ID. The recordset ID will be generated using CUID algorithm.”

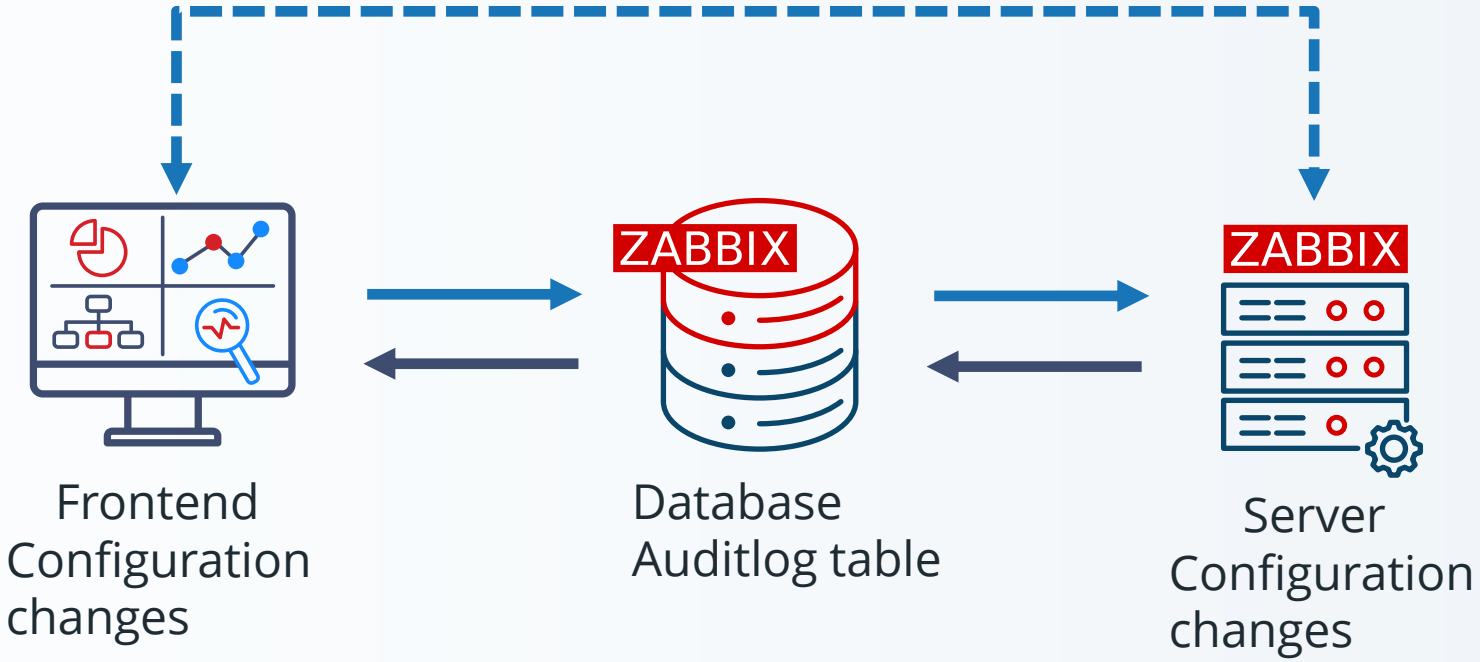
- Script execution has one single recordset ID.
- During the linking - all newly created audit entries are saved with the same recordset ID.

UI, Audit Details



Technical Implementation

Server audit has independent implementation from front-end, but they produce the same entries for same operations.



Technical Implementation

Database changes

Old auditlog and auditlog_details tables are removed during upgrade patch to 6.0.

New auditlog table is created.

```
TABLE|auditlog|auditid|0
FIELD           |auditid          |t_id           |       |NOT NULL       |0
FIELD           |userid           |t_id           |       |NOT NULL       |0

FIELD           |clock            |t_time         |'0'    |NOT NULL       |0

FIELD           |action           |t_integer      |'0'    |NOT NULL       |0
FIELD           |resourcetype     |t_integer      |'0'    |NOT NULL       |0
FIELD           |note             |t_varchar(128) |''     |NOT NULL       |0
FIELD           |ip               |t_varchar(39)  |''     |NOT NULL       |0
FIELD           |resourceid       |t_id           |       |NULL           |0

FIELD           |resourcename     |t_varchar(255) |''     |NOT NULL       |0


INDEX           |1                |userid,clock
INDEX           |2                |clock
INDEX           |3                |resourcetype,resourceid
```

```
TABLE|auditlog|auditid|0
FIELD           |auditid          |t_cuid         |       |NOT NULL       |0
FIELD           |userid           |t_id           |       |NULL           |0
FIELD           |username         |t_varchar(100) |''     |NOT NULL       |0
FIELD           |clock            |t_time         |'0'    |NOT NULL       |0
FIELD           |ip               |t_varchar(39)  |''     |NOT NULL       |0
FIELD           |action           |t_integer      |'0'    |NOT NULL       |0
FIELD           |resourcetype     |t_integer      |'0'    |NOT NULL       |0


FIELD           |resourceid       |t_id           |       |NULL           |0
FIELD           |resource_cuid    |t_cuid         |       |NULL           |0
FIELD           |resourcename     |t_varchar(255) |''     |NOT NULL       |0
FIELD           |recordsetid      |t_cuid         |       |NOT NULL       |0
FIELD           |details          |t_longtext     |''     |NOT NULL       |0
INDEX           |1                |userid,clock
INDEX           |2                |clock
INDEX           |3                |resourcetype,resourceid
```

Schema update

ZABBIX

- auditid is now CUID
- userid can be NULL (no more foreign reference on users table)
- username is added
- resource_cuid is added(alternative to resource, only for HA)
- recordsetid is added
- note and other auditlog_details table data now is in details (JSON)

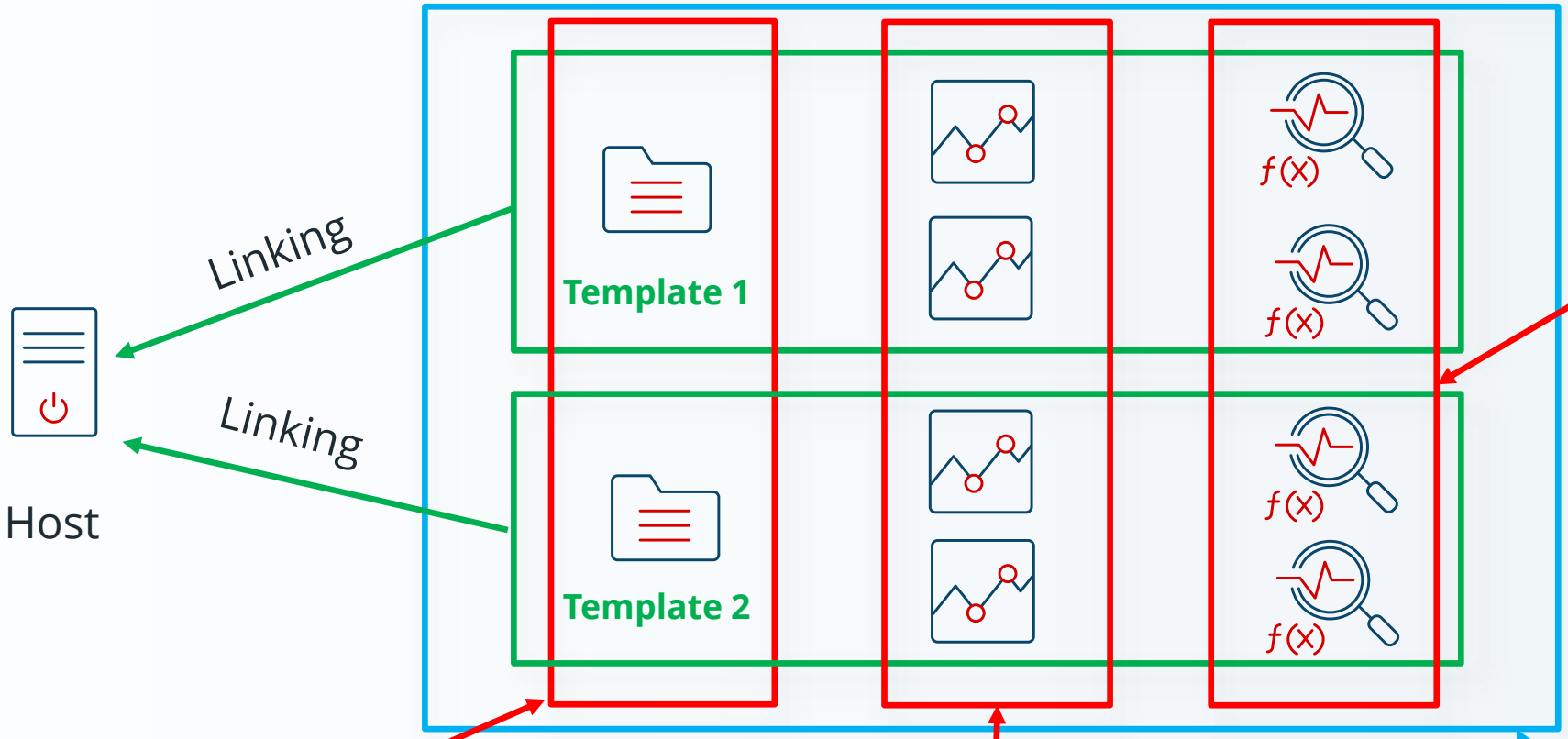
Technical Implementation

```
TABLE|auditlog|auditid|0
FIELD           |auditid          |t_id           |       |NOT NULL       |0
FIELD           |userid           |t_id           |       |NOT NULL       |0

FIELD           |clock            |t_time         |'0'    |NOT NULL       |0

FIELD           |action           |t_integer      |'0'    |NOT NULL       |0
FIELD           |resourcetype     |t_integer      |'0'    |NOT NULL       |0
FIELD           |note             |t_varchar(128) |''     |NOT NULL       |0
FIELD           |ip               |t_varchar(39)  |''     |NOT NULL       |0
FIELD           |resourceid       |t_id           |       |NULL           |0

FIELD           |resourcename     |t_varchar(255) |''     |NOT NULL       |0


INDEX           |1                |userid,clock
INDEX           |2                |clock
INDEX           |3                |resourcetype,resourceid
```

```
TABLE|auditlog|auditid|0
FIELD           |auditid          |t_cuid         |       |NOT NULL       |0
FIELD           |userid           |t_id           |       |NULL           |0
FIELD           |username         |t_varchar(100) |''     |NOT NULL       |0
FIELD           |clock            |t_time         |'0'    |NOT NULL       |0
FIELD           |ip               |t_varchar(39)  |''     |NOT NULL       |0
FIELD           |action           |t_integer      |'0'    |NOT NULL       |0
FIELD           |resourcetype     |t_integer      |'0'    |NOT NULL       |0


FIELD           |resourceid       |t_id           |       |NULL           |0
FIELD           |resource_cuid    |t_cuid         |       |NULL           |0
FIELD           |resourcename     |t_varchar(255) |''     |NOT NULL       |0
FIELD           |recordsetid      |t_cuid         |       |NOT NULL       |0
FIELD           |details          |t_longtext     |''     |NOT NULL       |0
INDEX           |1                |userid,clock
INDEX           |2                |clock
INDEX           |3                |resourcetype,resourceid
```

Why there is no index on RecordSet ID ?

Bulk SQL



insert into triggers (...) values (trigger1, trigger2, etc...)

insert into trigger_depends (...) values (trigger1_depends, trigger2_depends, etc...)

insert into trigger_tags (...) values (trigger_tag1, trigger_tag_2, etc...)

insert into hosts_templates (...) values (template 1, template 2, etc...)

insert into items (...) values (item1, item2, etc...)

insert into auditlog (...) values (item 1 data, item2 data, trigger1 data, trigger2 data) etc.

Performance impact

LINKING with audit	
MASTER	
1 000 triggers	00:00:0.19900000000052387
5 000 triggers	00:00:0.8909999999959837
10 000 triggers	00:00:1.743000000002212
20 000 triggers	00:00:3.754000000000815
40 000 triggers	00:00:7.335999999995693
68 000 triggers	00:00:12.494000000006054

LINKING without audit	
MASTER	
1 000 triggers	00:00:0.2139999999999418
5 000 triggers	00:00:0.908000000003085
10 000 triggers	00:00:1.7699999999967986
20 000 triggers	00:00:3.518000000003667
40 000 triggers	00:00:7.495999999999185
68 000 triggers	00:00:11.953999999997905

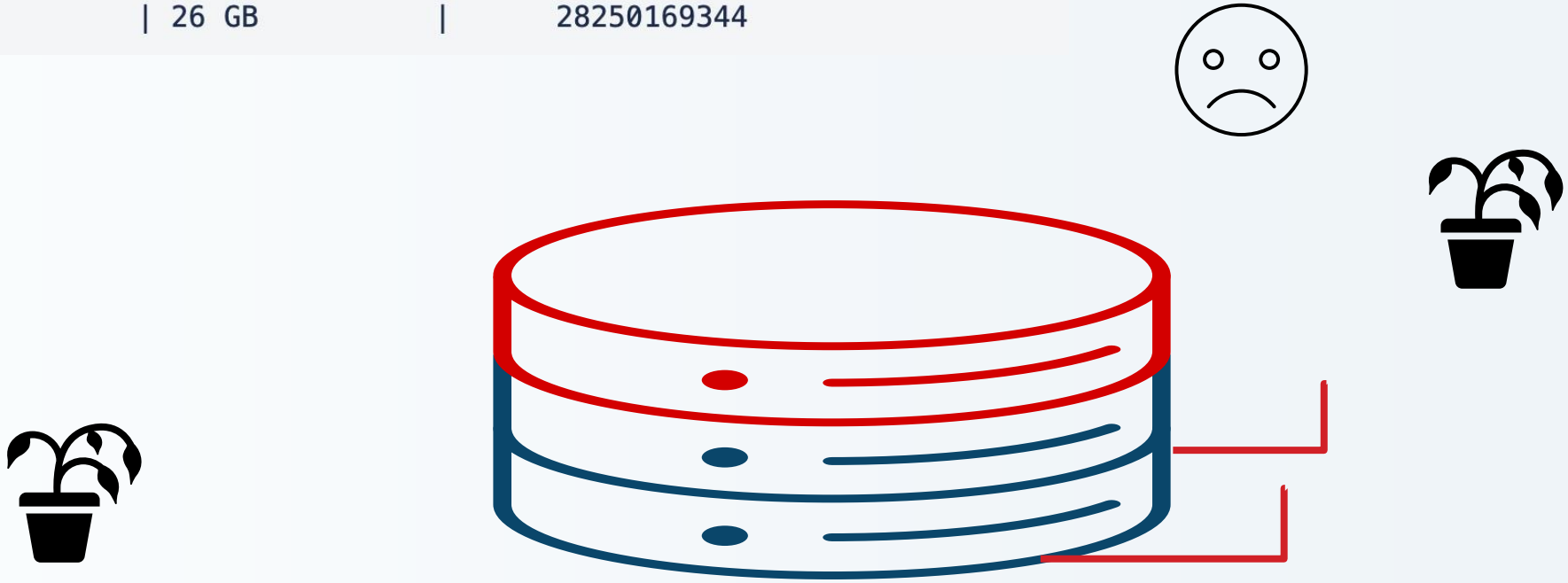
~4-5%

Data storage impact

ZABBIX

Impact on data storage requirements... (if forgotten about)

table_name	pg_size_pretty	pg_relation_size
auditlog	635 GB	681780338688
item_discovery	26 GB	28250169344



TimescaleDB

- Audit looks like a time-series data.
- Auditlog table has clocks column.
- Audit records are saved once and then only are read, never modified (and deleted).

- Can it be turned it into a hypertable ?

- Automated partitioning by time.
- + compression (Community edition license).

TimescaleDB

ZABBIX

table, dimension

```
SELECT create_hypertable('history', 'clock', chunk_time_interval => 86400, migrate_data => true);
SELECT create_hypertable('history_uint', 'clock', chunk_time_interval => 86400, migrate_data => true);
SELECT create_hypertable('history_log', 'clock', chunk_time_interval => 86400, migrate_data => true);
SELECT create_hypertable('history_text', 'clock', chunk_time_interval => 86400, migrate_data => true);
SELECT create_hypertable('history_str', 'clock', chunk_time_interval => 86400, migrate_data => true);
SELECT create_hypertable('trends', 'clock', chunk_time_interval => 2592000, migrate_data => true);
SELECT create_hypertable('trends_uint', 'clock', chunk_time_interval => 2592000, migrate_data => true);
```

So, why not just add:

```
SELECT create_hypertable('auditlog', 'clock', chunk_time_interval => 86400, migrate_data => true) ?
```

TimescaleDB

ZABBIX

ERROR: cannot create a unique index without the column "clock" (used in partitioning ..

History uint indexes:

"history_uint_pkey" PRIMARY KEY, btree (itemid, **clock**, ns)

Auditlog Indexes:

"auditlog_pkey" PRIMARY KEY, btree (auditid)

"auditlog_1" btree (userid, clock)

"auditlog_2" btree (clock)

"auditlog_3" btree (resourcetype, resourceid)

Primary key has no clock...

TimescaleDB

ZABBIX

Every unique index needs to contain clock dimension against which we partition by.

In Auditlog Primary key has no clock, but it not a regular ID...

It is **CUID**:

c**lvb1wfub**00027viy6mphm75s

Timestamp

So, we can extract it and use for partitioning by time.

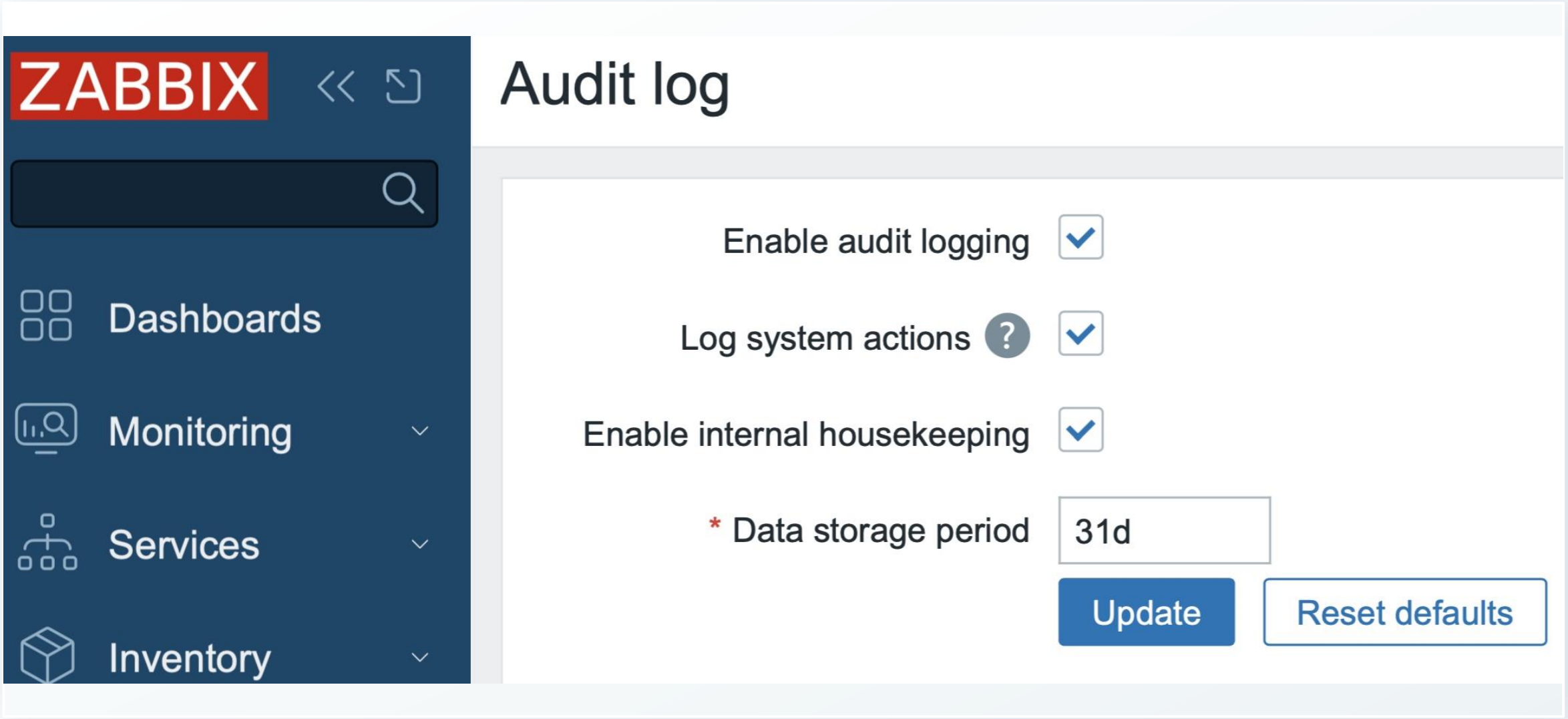
TimescaleDB

```
DROP FUNCTION IF EXISTS cuid_timestamp(cuid varchar(25));
CREATE OR REPLACE FUNCTION cuid_timestamp(cuid varchar(25)) RETURNS
integer AS $$
BEGIN
      RETURN CAST(base36_decode(substring(cuid FROM 2 FOR 8))/1000 AS
integer);
END;
$$ LANGUAGE 'plpgsql' IMMUTABLE;



PERFORM create_hypertable('auditlog', 'auditid', chunk_time_interval => 604800,
    time_partitioning_func => 'cuid_timestamp', migrate_data => true, if_not_exists
=> true);
```

ZBXNEXT-8520, added in 7.0

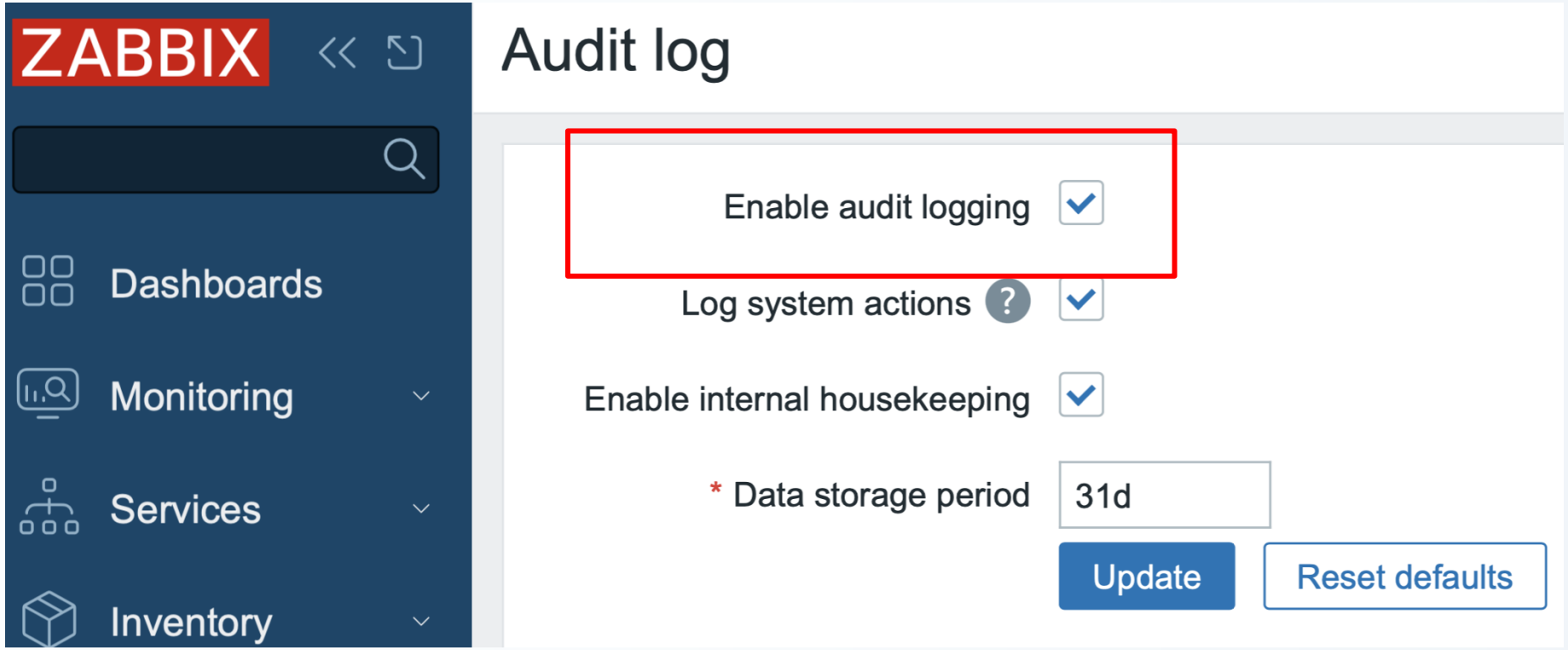
May take some time migrate existing data..

Administration



Administration

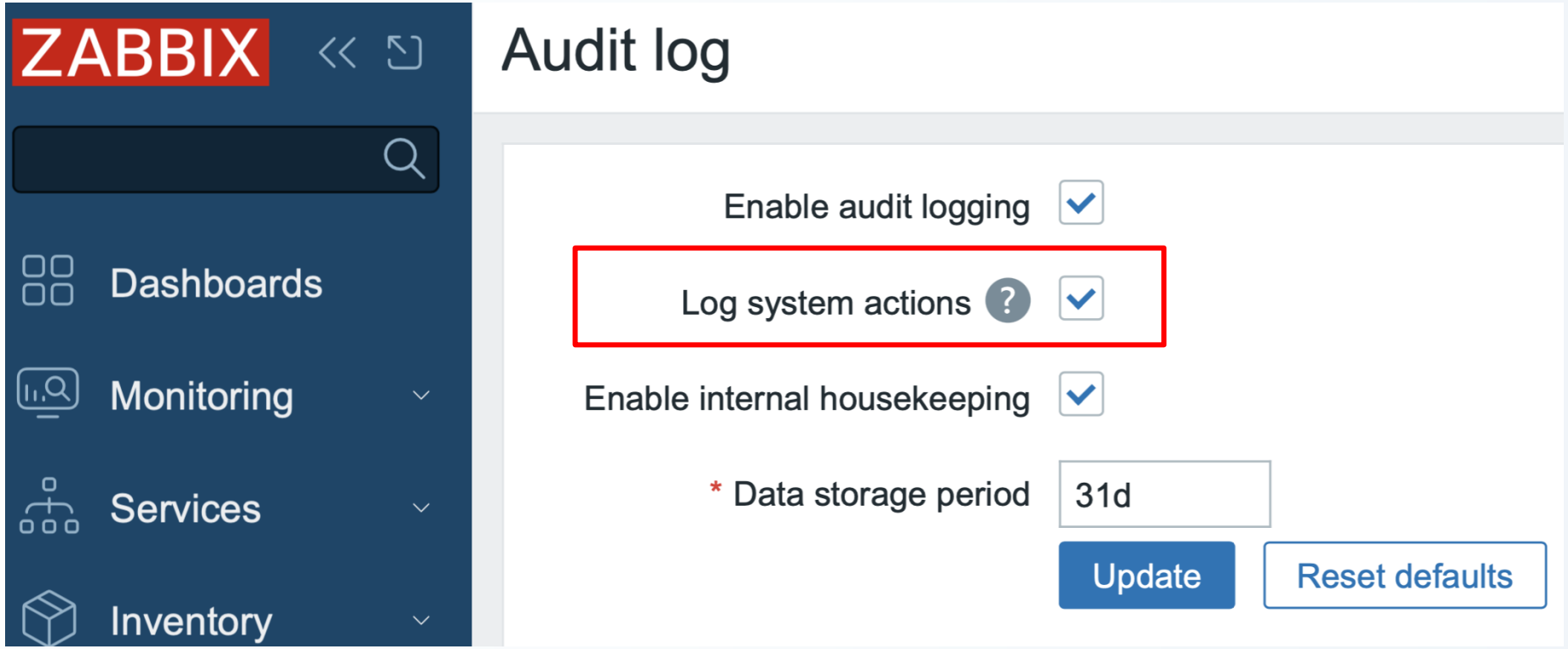
Enable audit logging



Disables ALL audit – including front-end.

Administration

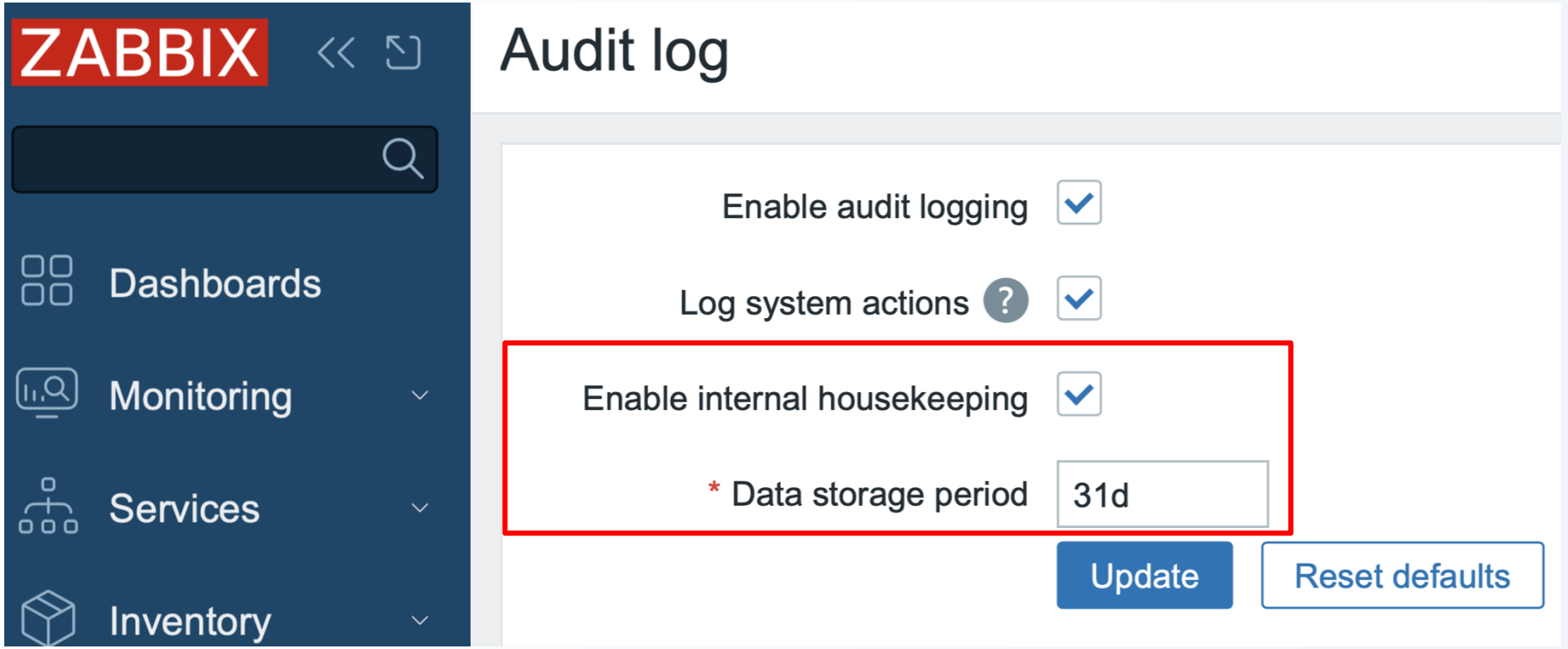
Log system actions (7.0)



Disables audit done by Zabbix server during Autoregistration, Network Discovery and LLD.

Administration

Audit has its own Housekeeping Schedule



If user, host, trigger, graph is deleted (including housekeeper) - all audit related to it stays.

Artjoms Rimdjonoks

C Developer