Tirocinanti

- **Salvatore**
- **Maria**
- **Domenico**

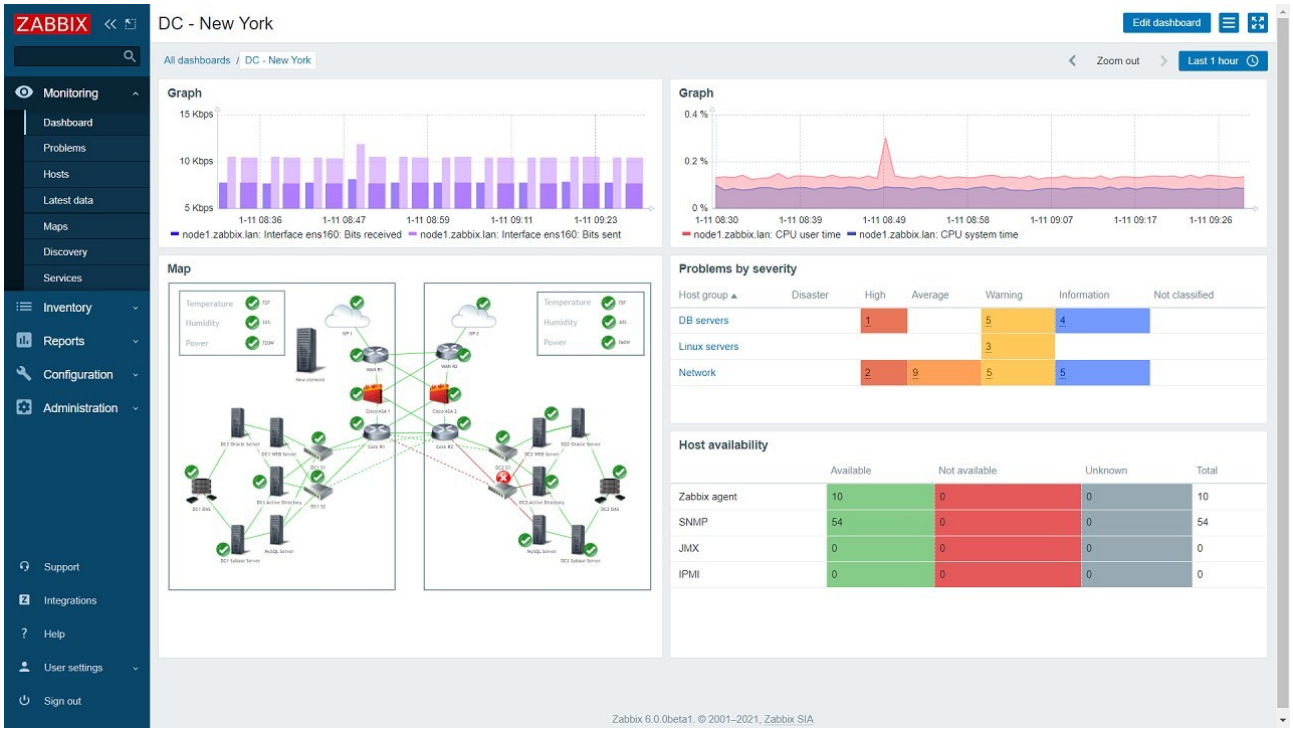
Presso azienda Novanex SRL

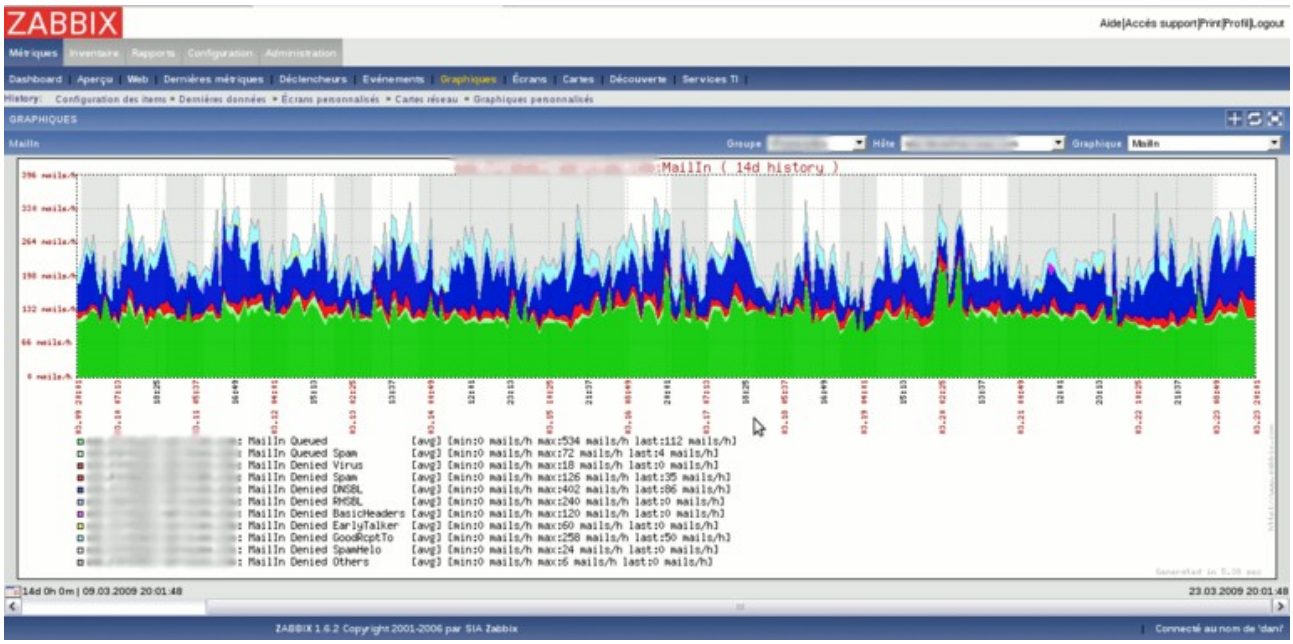
Programmatore sviluppo script Python

Zabbix è un software libero per il monitoraggio di reti e vari sistemi informatici, atto a tracciare lo stato di server e servizi di rete correlati. Ideato dal programmatore russo Alexei Vladishev, è rilasciato sotto i termini della *GNU General Public License* versione 2.

ZABBIX


ZABBIX
Monitoring
Inventory
Reports
Configuration
Administration
Dashboard
Problems
Overview
Web
Latest data
Graphs
Screens
Maps
Discovery
Services
Zabbix Global View
Edit dashboard
All dashboards / Zabbix Global View
Zoom out
Last 1 hour
Zabbix Cluster
Amazon Cloud
Azure Cloud
Zabbix server
Detected problems
Host group
Disaster
High
Average
Warning
Information
Not classified
Cloud/AWS
Cloud/Azure
End user services
HPC Cluster
Internal infrastructure
R&D Lab1
R&D Lab2
Region/Australia
Region/Brazil
Region/China
Region/Europe
Region/Japan
Region/USA
SAP HANA Infra
Zabbix infrastructure
Storage IOPs
UTC time
API calls/s
CPU usage
Collected values/s
Value cache misses
Zabbix server: Zabbix value cache hits: 1.83 Kvps


ZABBIX
DC - New York
Edit dashboard
All dashboards / DC - New York
Zoom out
Last 1 hour
Monitoring
Dashboard
Problems
Hosts
Latest data
Maps
Discovery
Services
Inventory
Reports
Configuration
Administration
Support
Integrations
Help
User settings
Sign out
Graph
node1.zabbix.lan: Interface ens160: Bits received
node1.zabbix.lan: Interface ens160: Bits sent
Graph
node1.zabbix.lan: CPU user time
node1.zabbix.lan: CPU system time
Map
Problems by severity
Host group
Disaster
High
Average
Warning
Information
Not classified
DB servers
Linux servers
Network
Host availability
Available
Not available
Unknown
Total
Zabbix agent
SNMP
JMX
IPMI
Zabbix 6.0.0beta1. © 2001–2021, Zabbix SIA



Esempio Installazione

Installa e configura Zabbix per la tua piattaforma

Install Zabbix repository

```
# wget
https://repo.zabbix.com/zabbix/6.4/ubuntu-arm64/pool/main/z/zabbix-
release/zabbix-release_6.4-1+ubuntu22.04_all.deb
# dpkg -i zabbix-release_6.4-1+ubuntu22.04_all.deb
# apt update
```

Install Zabbix server, frontend, agent

```
# apt install zabbix-server-mysql zabbix-frontend-php zabbix-apache-
conf zabbix-sql-scripts zabbix-agent
```

```
Create initial database
```

```
# mysql -uroot -p
password
mysql> create database zabbix character set utf8mb4 collate
utf8mb4_bin;
mysql> create user zabbix@localhost identified by 'password';
```

```
mysql> grant all privileges on zabbix.* to zabbix@localhost;
mysql> set global log_bin_trust_function_creators = 1;
mysql> quit;
```

On Zabbix server host import initial schema and data. You will be prompted to enter your newly created password.

```
# zcat /usr/share/zabbix-sql-scripts/mysql/server.sql.gz | mysql --
default-character-set=utf8mb4 -uzabbix -p zabbix
```

Disable log_bin_trust_function_creators option after importing database schema.

```
# mysql -uroot -p
password
mysql> set global log_bin_trust_function_creators = 0;
mysql> quit;
```

Configure the database for Zabbix server

Start Zabbix server and agent processes and make it start at system boot.

Edit file /etc/zabbix/zabbix_server.conf

```
DBPassword=password
```

Start Zabbix server and agent processes

Start Zabbix server and agent processes and make it start at system boot.

```
# systemctl restart zabbix-server zabbix-agent apache2
# systemctl enable zabbix-server zabbix-agent apache2
```

Abbiamo sviluppato le api di zabbix per il monitoraggio delle reti informatiche presso le infrastrutture server di vari clienti Novanext SRL.Il seguente linguaggio PYTHON
installazione api
pip install pyzabbix

codice di sviluppo

```
import requests
import json
```

```
ZABBIX_API_URL = "https://example.com/zabbix/api_jsonrpc.php"
UNAME = "Admin"
PWORD = "password"

r = requests.post(ZABBIX_API_URL,
                  json={
                      "jsonrpc": "2.0",
                      "method": "user.login",
                      "params": {
                          "user": UNAME,
                          "password": PWORD},
                      "id": 1
                  })

print(json.dumps(r.json(), indent=4, sort_keys=True))

AUTHTOKEN = r.json()["result"]

# Retrieve a list of problems
print("\nRetrieve a list of problems")
r = requests.post(ZABBIX_API_URL,
                  json={
                      "jsonrpc": "2.0",
                      "method": "problem.get",
                      "params": {
                          "output": "extend",
                          "selectAcknowledges": "extend",
                          "recent": "true",
                          "sortfield": ["eventid"],
                          "sortorder": "DESC"
                      },
                      "id": 2,
                      "auth": AUTHTOKEN
                  })

print(json.dumps(r.json(), indent=4, sort_keys=True))

#Logout user
print("\nLogout user")
r = requests.post(ZABBIX_API_URL,
                  json={
                      "jsonrpc": "2.0",
                      "method": "user.logout",
                      "params": {},
                      "id": 2,
                      "auth": AUTHTOKEN
                  })

print(json.dumps(r.json(), indent=4, sort_keys=True))


--------------------------------------------------------------------------------
------------------------------------------------


from pyzabbix.api import ZabbixAPI

    # Create ZabbixAPI class instance
    zapi = ZabbixAPI(url='https://localhost/zabbix/', user='Admin',
password='zabbix')
```

```
    # Get all monitored hosts
    result1 = zapi.host.get(monitored_hosts=1, output='extend')

    # Get all disabled hosts
    result2 = zapi.do_request('host.get',
                              {
                                  'filter': {'status': 1},
                                  'output': 'extend'
                              })

    # Filter results
    hostnames1 = [host['host'] for host in result1]
    hostnames2 = [host['host'] for host in result2['result']]

    # Logout from Zabbix
    zapi.user.logout()



    from pyzabbix.api import ZabbixAPI

    # Create ZabbixAPI class instance
    with ZabbixAPI(url='https://localhost/zabbix/', user='Admin',
password='zabbix') as zapi:

        # Get all monitored hosts
        result1 = zapi.host.get(monitored_hosts=1, output='extend')




    import sys
    import logging
    from pyzabbix.api import ZabbixAPI

    # Create ZabbixAPI class instance
    logger = logging.getLogger("pyzabbix")
    logger.setLevel(logging.DEBUG)
    handler = logging.StreamHandler(sys.stdout)
    logger.addHandler(handler)

    zapi = ZabbixAPI(url='http://localhost', user='Admin', password='zabbix')



    ZabbixAPI.login(Admin,********)
    Call user.login method
    urllib2.Request(http://localhost/api_jsonrpc.php, {"jsonrpc": "2.0",
"method": "user.login", "params": {"user": "Admin", "password": "********"},
"id": "1"})
    Response Body: {
        "jsonrpc": "2.0",
        "result": "********",
        "id": "1"
    }
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
def getMetricData(config, parameters, hosts, startTime, endTime):
    """Get metric data from Zabbix API"""
    # connection to zabbix
```

```
    logger.info(
        "Begin to connect to zabbix:{}
User/Pwd:{}/{}".format(str(config['ZABBIX_URL']), str(config['ZABBIX_USER']),

str(config['ZABBIX_PASSWORD'])))
    zapi = ZabbixAPI(url=config['ZABBIX_URL'], user=config['ZABBIX_USER'],
password=config['ZABBIX_PASSWORD'])
    logger.info("Get connection from zabbix success")

    # get hosts
    hosts_map = {}
    hosts_ids = []
    hosts_res = zapi.do_request('host.get', {
        'filter': {
            "host": hosts,
        },
        "sortfield": "name",
        'output': 'extend'
    })
    for item in hosts_res['result']:
        host_id = item['hostid']
        host = item['host']
        hosts_ids.append(host_id)
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-
pip install -U pip wheel

pip install pyzabbix
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
---

>>> import pyzabbix
>>> zapi = pyzabbix.ZabbixAPI("https://zabbixtest.lein.io/zabbix")
>>> zapi.auth
''
>>> zapi.api_version()

-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
----
>>> import getpass
>>> zapi.login(user="Admin", password=getpass.getpass())
Password:
>>> zapi.auth
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
---
>>> from pprint import pprint
>>> hosts = zapi.host.get()
>>> for host in hosts:
    print(host)
-------------------------------------------------------------------------------
-------------------------------------------------------------------------
#zapi.user.logout()
# for host in zapi.host.get():
     print(host["host"])
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
---
```

```
>>> hosts = zapi.host.get(filter={"host":"Zabbix server"})
>>> len(hosts)

>>> host = hosts[0]
>>> host["host"]
'Zabbix server'
>>> host["hostid"]
'10084'
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
--


>>> interfaces = zapi.hostinterface.get(hostids=[host["hostid"]])
>>> for intf in interfaces:
...     print(intf)

--------------------------------------------------------------------------------
----------------------------------------------------------------------------
>>> for intf in zapi.hostinterface.get(hostids=[host["hostid"]]):
...     print(intf["ip"])

--------------------------------------------------------------------------------
----------------------------------------------------------------------------
>>> for hostgroup in zapi.hostgroup.get(hostids=[host["hostid"]]):
...     pprint(hostgroup)
--------------------------------------------------------------------------------
----------------------------------------------------------------------------
from pyzabbix import ZabbixAPI

zapi = ZabbixAPI("http://zabbixserver.example.com")
zapi.login("zabbix user", "zabbix pass")
print("Connected to Zabbix API Version %s" % zapi.api_version())

for h in zapi.host.get(output="extend"):
    print(h['hostid'])


--------------------------------------------------------------------------------
----------------------------------------------------------------------------

from pyzabbix import ZabbixAPI

zapi = ZabbixAPI("http://zabbixserver.example.com")


zapi.session.auth = ("http user", "http password")


zapi.session.verify = False


zapi.timeout = 5.1

zapi.login("http user", "http password")
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
----
import sys
import logging
from pyzabbix import ZabbixAPI
```

```
stream = logging.StreamHandler(sys.stdout)
stream.setLevel(logging.DEBUG)
log = logging.getLogger('pyzabbix')
log.addHandler(stream)
log.setLevel(logging.DEBUG)


zapi = ZabbixAPI("http://zabbixserver.example.com")
zapi.login('admin','password')
-------------------------------------------------python
dataframe-------------------------------------------------------------------------
-----------------------------------------
for host in hosts:
cpu_item = zapi.item.get(hostids=host['hostid'],search={'name': 'CPU Busy'})
if len(cpu_item) == 0:
# print("No item for", host['name'])
continue
cpu_trend = zapi.trend.get(itemids=cpu_item[0]
['itemid'],limit=168,output=["clock", "value_avg", "value_max"])
if len(cpu_trend) == 0:
# print("No trend for", host['name'])
continue
df = pd.DataFrame(cpu_trend)
cpu_avg_max = df['value_max'].astype(float).mean()
cpu_max_avg = df['value_avg'].astype(float).max()
cpu_avg_std = df['value_avg'].astype(float).std()
cpu_up_var = (cpu_max_avg + cpu_avg_std) * 1.25

if cpu_avg_max > 50:
print(host['name']+','+str(cpu_max_avg)+','+str(cpu_avg_max)+','+str
(cpu_avg_std)+','+str(cpu_up_var))

-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
----
ZABBIX_SERVER = '<zabbix server url>'

zapi = ZabbixAPI(ZABBIX_SERVER)

# Login to the Zabbix API
zapi.login(user, pwd)

hosts = zapi.host.get(monitored_hosts=1, output='extend')
hosts = [{k:r[k] for k in ["host", "hostid"]} for r in hosts]

for host in hosts:
    print(host["host"])
    data = get_metric(host["hostid"], "Total CPU Utilization", (2020,5,1),
(2020,6,1))




def get_metric(host_id, metric, dfrom, dto):
    dfrom = int(datetime.datetime(*dfrom).timestamp())
    dto = int(datetime.datetime(*dto).timestamp())
    hostitems = zapi.item.get(filter={"hostid": host_id, "name": metric})
```

```
    t_metric = [item for item in hostitems if metric == item["name"] and
item["hostid"] == host_id]
    data = {"values": [], "timestamps": []}
    if len(t_metric):
        # Create a time range
        item = t_metric.pop()
        # Query item's history (integer) data
        history = zapi.history.get(hostids=[host_id],
                                   itemids=[item["itemid"]],
                                   time_from=dfrom,
                                   time_till=dto,
                                   output='extend',
                                   #limit='5000',
                                   history=item["value_type"]
                                   )
        if len(history) == 0:
            return data
        df = pd.DataFrame(history)
        df = df[["clock", "value"]]
        df["value"] = df["value"].astype(float)
        df["clock"] = df["clock"].astype(int)
        df["clock"] = df["clock"]*1000
        return {"values": df["value"].tolist(), "timestamps":
df["clock"].tolist()}
    return data


--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
-----
df = pd.DataFrame(cpu_trend)
cpu_avg_max = df['value_max'].astype(float).mean()
freq = df['clock'][df['value_max'].astype(float) > 80].count()
if freq > 5:
print(host['name']+','+str(freq)+','+str(cpu_avg_max))
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
----



from ZabbixAPI_py import Zabbix
from datetime import datetime
from datetime import timedelta




zabbix = Zabbix('host')
zabbix.login('id','password')




tenMin = datetime.today() - timedelta(minutes=10)
```

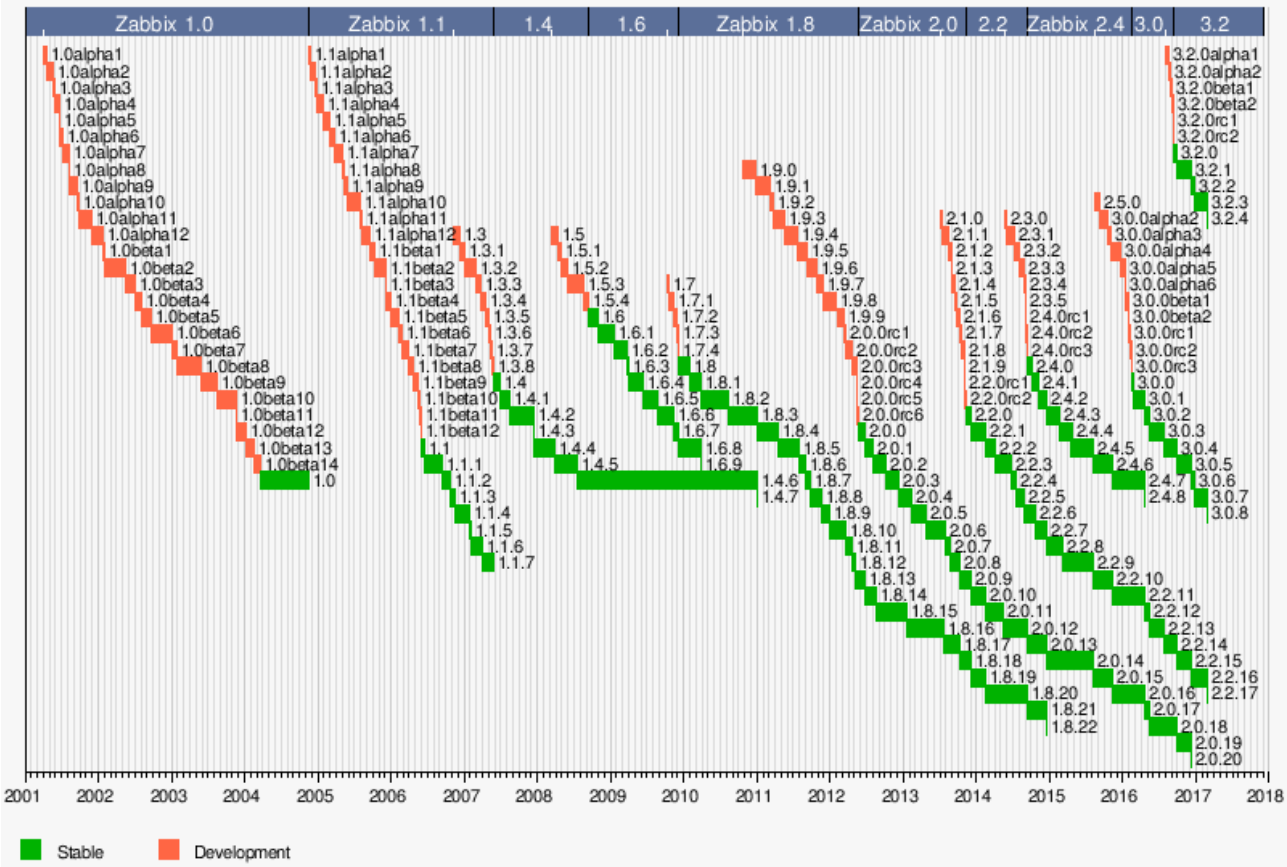
```
for x in zabbix.problem(method='get'):
    eventTime = datetime.fromtimestamp(int(x['clock'])) # Convert Unix Timestamp

    if tenMin < eventTime:
        msg = str(eventTime)+x['name']
        print(msg)
```

Nel 1998 Zabbix nacque come strumento proprietario utilizzato come progetto interno di un istituto bancario.Dopo tre anni, nel 2001, fu rilasciato al pubblico sotto i termini di una licenza di software libero. Ci sono voluti altri tre anni per il rilascio della versione stabile 1.0.

Dal 2005 fu fondata l'omonima società multinazionale Zabbix LLC con sedi negli Stati Uniti d'America, in Europa e Giappone.

Versioni pubblicate



Zabbix consta in diversi moduli separati:

•Server
•Agenti (*agent*)
•Frontend
•Proxy

Caratteristiche

Zabbix per la raccolta dei dati si interfaccia a vari database quali MySQL, PostgreSQL, SQLite, Oracle o IBM DB2.

Per verificare la disponibilità e la prontezza di alcuni servizi standard come SMTP o HTTP è sufficiente un *simple check*, ovvero non occorre installare alcun software sull'*host* da monitorare. Per monitorare altri tipologie di risorse quali il carico della CPU, la congestione di rete, lo spazio su disco, ecc. occorre installare il software *Zabbix agent*.

Fra i protocolli supportati vi sono SNMP, TCP e ICMP, così come IPMI, JMX, SSH, Telnet, o usando comandi personalizzati (*user parameters*).

Zabbix supporta vari meccanismi di notifica in *sistema real-time* fra i quali la posta elettronica, il protocollo di messaggistica istantanea XMPP, e via SMS.

Una delle caratteristiche di Zabbix è l'utilizzo di *agent* implementati evitando linguaggi ad alto livello. Questo permette alte prestazioni e la possibilità di monitorare di centinaia di migliaia di dispositivi.

Fra le funzionalità supportate dai vari agent:

- **Auto rilevamento di server e dispositivi di rete**
- **Low-level discovery**
- **Monitoraggio distribuito con gestione centralizzata via interfaccia web**
- **Supporto per polling e trapping**
- **Permessi utente flessibili**
- **Notifiche su eventi flessibili**
- **Monitoraggio**
- **Dati monitorati raccolti in grafici definiti dall'utente**
- **Audit dei log**

Storia.Nel 1998 Zabbix nacque come strumento proprietario utilizzato come progetto interno di un istituto bancario. Dopo tre anni, nel 2001, fu rilasciato al pubblico sotto i termini di una licenza di software libero. Ci sono voluti altri tre anni per il rilascio della versione stabile 1.0