Vulnerability Assessment Report

LG Security Specialist Course
Team2

1. Executive Summary

The purpose of this security assessment is to conduct a security evaluation of our classmate's systems (Team3 AACS).

Documentation was reviewed to ensure all the functions, interfaces, executables, libraries were documented as a part of the process.

Our team sourced any additional documentation/information necessary for assessment via email.

To ensure the security protocols mapped to the product documentation, the device and the software package were subjected to testing and analysis.

The project under test was found to be not in compliance with some of the security policy and the reasons are described in this report.

Module	Role	Environment
ACS	Attendance Check / Client	Windows 10
FRS	Face Recognition / Server	Jetson Nano (Ubuntu)

A total of 33 unique vulnerabilities were found during this assessment. Critical, High, Medium and Low severity vulnerabilities were found to exist across the system. Also, Vulnerabilities are classified into categories as follows.

Severity	Count
Critical	5
High	9
Medium	16
Low	3
Total	**33**

Category	Count
Spoofing	5
Tampering	6
Repudiation	1
Information Disclosure	13
DoS	6
Elevation of Privilege	0
etc.	2
Total	**33**

It is our recommendation that immediate action be taken to resolve these vulnerabilities by applying patches and adjusting system configurations as necessary. In addition, a patch and configuration management process should be implemented to continually assess system risk level as vulnerabilities are discovered. This will ensure relevant security patches and configurations are applied in a timely manner.

2. Methodology

1) Information Gathering

We scanned for known vulnerabilities that can be exploited using Trommel, Flawfinder, and Nmap.
Other sources of known vulnerabilities that can be exploited were from the document and design review, source code review, firmware review, and fuzzing.

a) Review architecture design and source code. (Code Review result is attached)

b) Static Analysis Tool

Tool	Version
Flawfinder	2.0.18
Trommel	2019-08-02

c) Network Scanning

Tool	Version
nmap	7.91

d) Review 3rd party library/source codes

2) Exploitation

a) Fuzz Testing Tool

Tool	Version
MiniFuzz	1.5.5
zzuf	0.15

b) Penetration Test

Tool	Version
Burp Suite	2020-12-01
SSLScan	1.11.5
Metasploit	6.0.48-dev
Customized Program (For MITM)	Coding by Visual Studio 2019

3) Risk Assessment

a) Establish the scope of assessment and identify assets.
b) Identify the possible security risks through the analysis of assets, threats and vulnerabilities.
c) Categorize vulnerabilities by STRIDE.

Category	
S	Spoofing
T	Tampering
R	Repudiation
I	Information Disclosure
D	Denial of Service
E	Elevation of privilege

d) Rating vulnerabilities considering their likelihood and impact.

Severity	Description
Critical	They are relatively easy for attackers to exploit or may provide them with full control of the affected systems.
High	High severity vulnerabilities may not provide the security policy of sensitive data to affected systems.
Medium	These vulnerabilities often provide information to attackers that may assist them in mounting subsequent attacks on your network.
Low	Low severity vulnerabilities do not need to be patch immediately and can be resolved during the next updates maintenance window.

3. Findings

1) Information Gathering

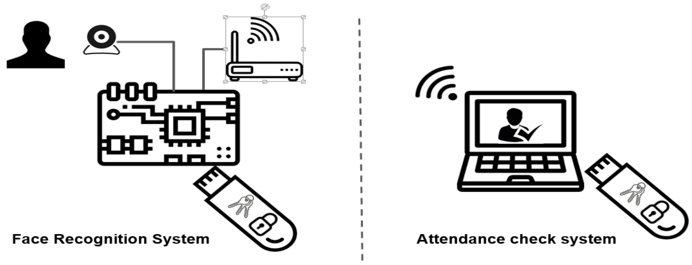
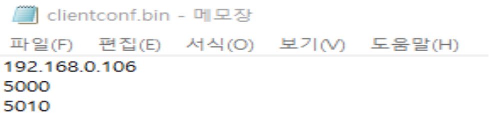
As a first step, we reviewed a number of documents describing the product under evaluation, these are documents received from Team 3.

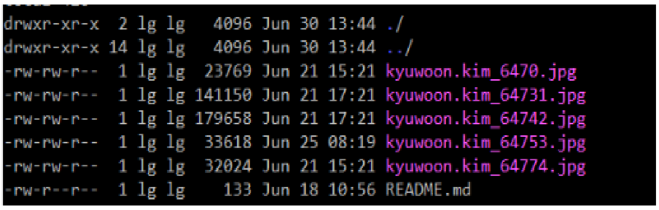
No	Document Name
1	AACS.mov
2	AACS_Architecture_Design.docx
3	AACS_Cryptographic_Algorithms.pptx
4	AACS_Deliverables.xlsx
5	AACS_Opensource_Vulnerability_Report.xlsx
6	AACS_Remaining_Works.pptx
7	AACS_Requirements.docx
8	AACS_Setup_Manual.pptx
9	AACS_Static_Analysis_Report.xlsx
10	AACS_Testcase.xlsx
11	AACS_ThreatAnalysis_RiskAssessment_Result.xlsx
12	AACS_UX.pptx
13	AACS_Work_Schedule.xlsx
14	Team3_Phase1_Presentation.pptx

a) Review Artifacts

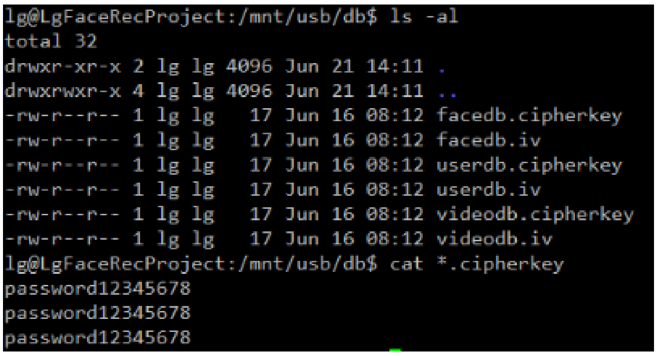
We reviewed the architecture design document, configuration, and source code. Based on the provided artifacts, the expected vulnerabilities were derived as follows.

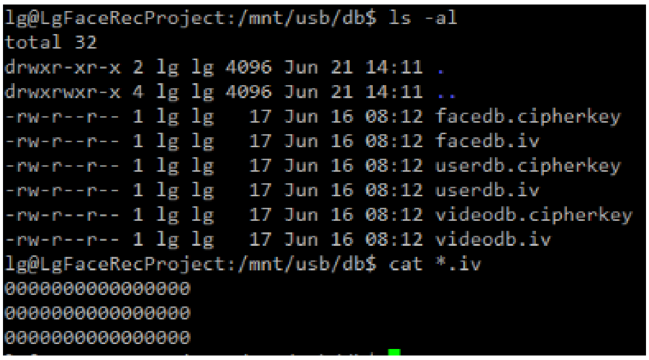
Module	Category	Description	
Client/ Server	Repudiation	Finding	There is no way to track the activities of the systems when applications are terminated.
		How	No logging files in client/server local storage
Server	Insecure Configuration	Finding	Since the key is stored in a USB, that may lead to insecure default behavior if malwares are in the USB

		How	
Client	Information Disclosure	Finding	Server IP/port information is disclosed in clientconf.bin and that may be a start of being the attacker's target.
		How	 clientconf.bin - 메모장 파일(F) 편집(E) 서식(O) 보기(V) 도움말(H) 192.168.0.106 5000 5010
Image Storage	Information Disclosure/ Tampering	Finding	No encryption on image files on the local storage. Attackers can get the student information from the image file name or add/modify/delete the image files if the attacker has access to the system.

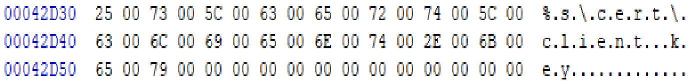
		How	
			`drwxr-xr-x  2 lg lg   4096 Jun 30 13:44 ./`
			`drwxr-xr-x 14 lg lg   4096 Jun 30 13:44 ../`
			`-rw-rw-r--  1 lg lg  23769 Jun 21 15:21 kyuwoon.kim_6470.jpg`
			`-rw-rw-r--  1 lg lg 141150 Jun 21 17:21 kyuwoon.kim_64731.jpg`
			`-rw-rw-r--  1 lg lg 179658 Jun 21 17:21 kyuwoon.kim_64742.jpg`
			`-rw-rw-r--  1 lg lg  33618 Jun 25 08:19 kyuwoon.kim_64753.jpg`
			`-rw-rw-r--  1 lg lg  32024 Jun 21 15:21 kyuwoon.kim_64774.jpg`
			`-rw-r--r--  1 lg lg    133 Jun 18 10:56 README.md`

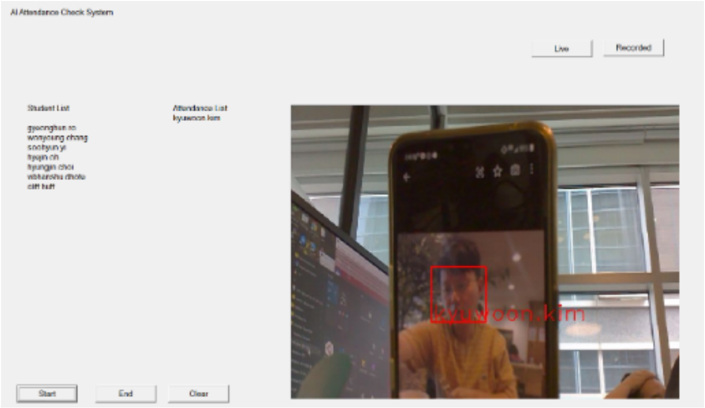
Cryptography	Cryptographic Vulnerability	Finding	The key files for authentication have the same encryption key / IV.

		How	
			`lg@LgFaceRecProject:/mnt/usb/db$ ls -al`
			`total 32`
			`drwxr-xr-x 2 lg lg 4096 Jun 21 14:11 .`
			`drwxrwxr-x 4 lg lg 4096 Jun 21 14:11 ..`
			`-rw-r--r-- 1 lg lg   17 Jun 16 08:12 facedb.cipherkey`
			`-rw-r--r-- 1 lg lg   17 Jun 16 08:12 facedb.iv`
			`-rw-r--r-- 1 lg lg   17 Jun 16 08:12 userdb.cipherkey`
			`-rw-r--r-- 1 lg lg   17 Jun 16 08:12 userdb.iv`
			`-rw-r--r-- 1 lg lg   17 Jun 16 08:12 videodb.cipherkey`
			`-rw-r--r-- 1 lg lg   17 Jun 16 08:12 videodb.iv`
			`lg@LgFaceRecProject:/mnt/usb/db$ cat *.cipherkey`
			`password12345678`
			`password12345678`
			`password12345678`

			
			`lg@LgFaceRecProject:/mnt/usb/db$ ls -al`
			`total 32`
			`drwxr-xr-x 2 lg lg 4096 Jun 21 14:11 .`
			`drwxrwxr-x 4 lg lg 4096 Jun 21 14:11 ..`
			`-rw-r--r-- 1 lg lg   17 Jun 16 08:12 facedb.cipherkey`
			`-rw-r--r-- 1 lg lg   17 Jun 16 08:12 facedb.iv`
			`-rw-r--r-- 1 lg lg   17 Jun 16 08:12 userdb.cipherkey`
			`-rw-r--r-- 1 lg lg   17 Jun 16 08:12 userdb.iv`
			`-rw-r--r-- 1 lg lg   17 Jun 16 08:12 videodb.cipherkey`
			`-rw-r--r-- 1 lg lg   17 Jun 16 08:12 videodb.iv`
			`lg@LgFaceRecProject:/mnt/usb/db$ cat *.iv`
			`0000000000000000`
			`0000000000000000`
			`0000000000000000`

	Information Disclosure	Finding	we can check the path of the private key and the certificate partially/fully by Hex Editor

		How	
			`00042D30  25 00 73 00 5C 00 63 00 65 00 72 00 74 00 5C 00  %.s.\.c.e.r.t.\.`
			`00042D40  63 00 6C 00 69 00 65 00 6E 00 74 00 2E 00 6B 00  c.l.i.e.n.t...k.`
			`00042D50  65 00 79 00 00 00 00 00 00 00 00 00 00 00 00 00  e.y.............`

Face Recognition	Logic Errors	Finding	The system cannot distinguish between the picture and the real person.
		How	 AI Attendance Check System Student List gyeonghun ro wonyoung chang soohyun yi hyojin oh hyungjin choi vibharshu dhote cliff huff Attendance List kyuwoon kim

b) Static Analysis

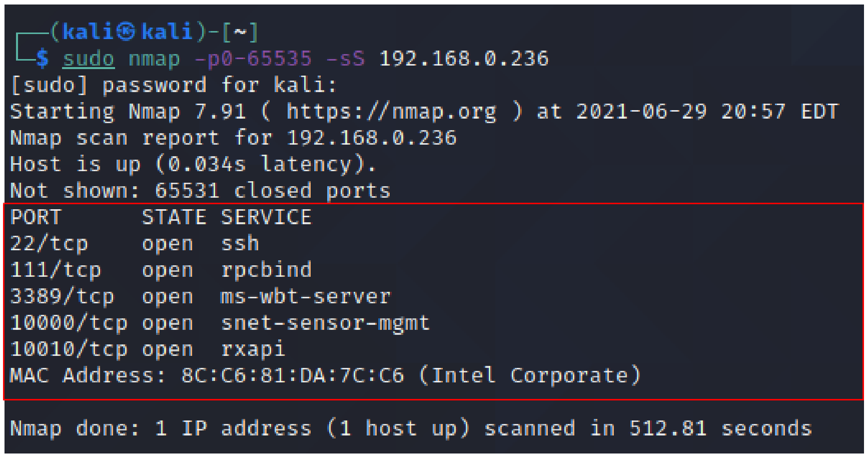
We found some issues that need to be addressed in the source code. And it should also contain evidence of addressing the risks identified. However, there is no evidence of a risk register and addressing of the issues identified.

Tool	Target	Found	Result
Flawfinder (https://dwheeler.com/flawfinder/)	Client	13	Using safe string API/handling buffer API is required to prevent buffer overflow
	Server	3	Checking buffer boundaries are required in face recognition module (Check buffer boundaries if used in a loop including recursive loops (CWE-120, CWE-20))
Trommel	Client	39	Server IP/Port information is disclosed in clientconf.bin and gives a

(https://github.com/CERTCC/trommel)			hint to DoS attacks.
	Server	25	keywords such as password/username/ssl/admin are detected and gives hints when reviewing source codes

c) Network Scanning

Using NMAP we could check the open ports and services versions running on a server that helps us to understand about the services running on the server so that later it helps you while pentesting.

Scanning Result (nmap)


```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -p0-65535 -sS 192.168.0.236
[sudo] password for kali:
Starting Nmap 7.91 ( https://nmap.org ) at 2021-06-29 20:57 EDT
Nmap scan report for 192.168.0.236
Host is up (0.034s latency).
Not shown: 65531 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
111/tcp   open  rpcbind
3389/tcp  open  ms-wbt-server
10000/tcp open  snet-sensor-mgmt
10010/tcp open  rxapi
MAC Address: 8C:C6:81:DA:7C:C6 (Intel Corporate)

Nmap done: 1 IP address (1 host up) scanned in 512.81 seconds
```

d) Review 3rd party library/source codes

This project used open source openSSL and openCV. These open sources may contain vulnerabilities that an attacker could exploit, so we should review them to make sure they do not contain well-known vulnerabilities.

Module	Used Version	CVE Link	Review Result
openSSL	1.1.1k (3/25/2021)	https://www.cvedetails.com/vulnerability-list/vendor_id-217/Openssl.html	Recent CVEs of openSSL can't be affected by the latest version(1.1.1k).
openCV	4.5.1 (latest 4.5.2)	https://www.cvedetails.com/vulnerability-list/vendor_id-16327/Opencv.html	There are no known vulnerabilities at 4.5.1. There are no security patches between 4.5.1 and the latest version(4.5.2).

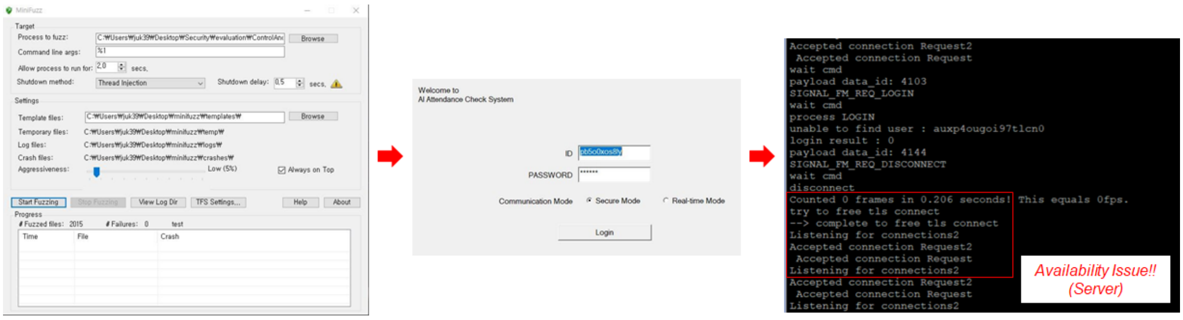
2) Exploitation

a) Fuzzing

i) Client

We performed fuzzing on the client using the MiniFuzz tool. In the client, the vulnerability could not be found through fuzzing, and it was found that the server connected to the client showed an availability problem.

Tool	Target	Environment	Attack Scenario & Result
MiniFuzz	[Client] [Server] ID & PW	Windows 10 Visual Studio 2019	[Client] We had run 2,000 times as below but segmentation fault wasn't found. 1. Rebuild program after adding source code to use the program arguments 2. Generate randomly manipulated user id and password 3. Run the program in order to login using random id and password 4. Check the result of program execution 5. Repeat step 2-4 [Server] No crash but after repeating 500 times of connection,the server cannot initialize SSL. (availability issue)



MiniFuzz

Target

Process to fuzz: C:₩Users₩juk39₩Desktop₩Security₩evaluation₩ControlAn

Command line args: %1

Allow process to run for: 2.0 secs.

Shutdown method: Thread Injection

Shutdown delay: 0.5 secs.

Settings

Template files: C:₩Users₩juk39₩Desktop₩minifuzz₩templates₩

Temporary files: C:₩Users₩juk39₩Desktop₩minifuzz₩temp₩

Log files: C:₩Users₩juk39₩Desktop₩minifuzz₩logs₩

Crash files: C:₩Users₩juk39₩Desktop₩minifuzz₩crashes₩

Aggressiveness: Low (5%)

☑ Always on Top

Progress

Fuzzed files: 2015 # Failures: 0 test

Time	File	Crash

Welcome to
AI Attendance Check System

ID pb5o0xos8y

PASSWORD ******

Communication Mode ● Secure Mode ○ Real-time Mode

```
Accepted connection Request2
 Accepted connection Request
wait cmd
payload data_id: 4103
SIGNAL_FM_REQ_LOGIN
wait cmd
process LOGIN
unable to find user : auxp4ougoi97tlcn0
login result : 0
payload data_id: 4144
SIGNAL_FM_REQ_DISCONNECT
wait cmd
disconnect
Counted 0 frames in 0.206 seconds! This equals 0fps.
try to free tls connect
--> complete to free tls connect
Listening for connections2
Accepted connection Request2
 Accepted connection Request
Listening for connections2
Accepted connection Request2
 Accepted connection Request
Listening for connections2
```

ii) Server

We performed fuzzing on the server using AFL and zuff. We could not find any vulnerabilities in the server through fuzzing.

Tool	Target	Environment	Attack Scenario & Result
AFL	[Server] User DB file	Jetson Nano	AFL doesn't support coverage based fuzzing on the ARM environment.
zzuf	[Server] User DB file	VM Kali Linux	We had run over 30,000 times as below but segmentation fault wasn't found. 1. Rebuild program after removing source code related to face recognition

			2. Generate randomly manipulated user db file using zzuf 3. Run the program in order to read abnormal user db file 4. Check the result of program execution 5. Repeat step 2-4
	[Server] Registered Image file	Jetson Nano	We had run over 10,000 times as below but segmentation fault wasn't found. 1. Rebuild program after removing source code related to socket 2. Generate randomly manipulated jpg file using zzuf 3. Run the program in order to read abnormal jpg file 4. Check the result of program execution 5. Repeat step 2-4

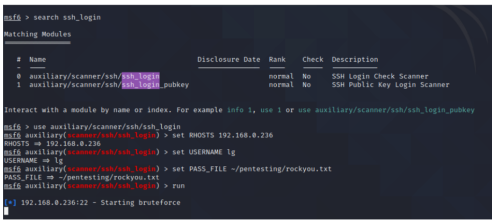
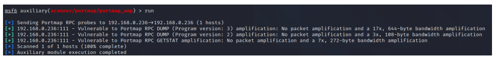
b) Penetration Test

i) General Tools

The SUT(System Under Test) has been scanned for known vulnerabilities that can be exploited using Burp Suite, SSLScan. Other sources of known vulnerabilities that can be exploited were from the document and design review, source code review, firmware review, information gathering and fuzzing. The Burp Suite tool testing for attacking SSL connection was not able to be extensively tested because the system was not a web application.

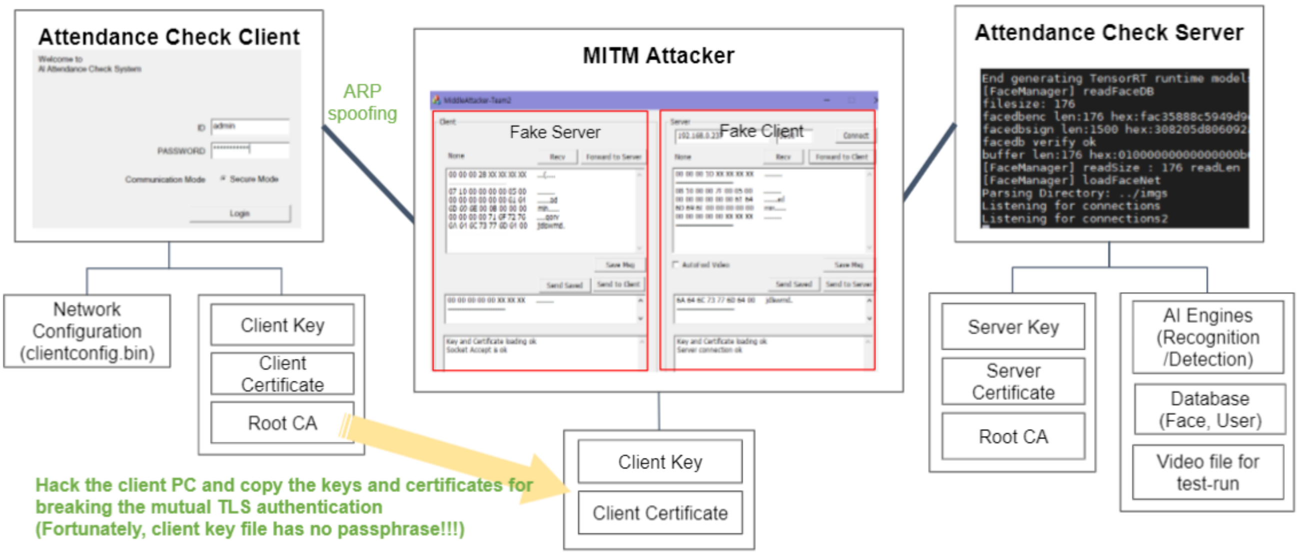
Exploitable vulnerabilities discovered were:

Tool	Environment	Target	Description	Result
Burp Suite	VM Kali Linux	Credentials of server system	Conduct BruteForce attack to get the id/password of the server system.	We failed to attack with Burp intruder to the target port and anything to analyze. It doesn't look like it applies to attack against a kind of this project.
SSL Scan	Windows Subsystem for Linux	Scanning TLS/SSL configuration to find out weakness of openSSL	Perform a wide variety of tests over the specified target. Analysis a comprehensive list of the protocols and ciphers accepted by an SSL/TLS server along with some other information useful in a security test.	Vulnerability found for TLS Compression: ● OpenSSL version does not support compression ● Rebuild with zlib1g-dev package for zlib support
Metasploit	VM Kali Linux	SSH login	Steps to exploit: a) Download hacked password list from internet b) Execute msfconsole	We couldn't get the successful result for 3 days.

			c) Set metasploit module 'auxiliary/scanner/ssh/ssh_login' d) Set RHOSTS, USERNAME, PASS_FILE e) Run	 msf6 > search ssh_login Matching Modules ================ # Name Disclosure Date Rank Check Description - ---- --------------- ---- ----- ----------- 0 auxiliary/scanner/ssh/ssh_login normal No SSH Login Check Scanner 1 auxiliary/scanner/ssh/ssh_login_pubkey normal No SSH Public Key Login Scanner Interact with a module by name or index. For example info 1, use 1 or use auxiliary/scanner/ssh/ssh_login_pubkey msf6 > use auxiliary/scanner/ssh/ssh_login msf6 auxiliary(scanner/ssh/ssh_login) > set RHOSTS 192.168.0.236 RHOSTS ⇒ 192.168.0.236 msf6 auxiliary(scanner/ssh/ssh_login) > set USERNAME lg USERNAME ⇒ lg msf6 auxiliary(scanner/ssh/ssh_login) > set PASS_FILE ~/pentesting/rockyou.txt PASS_FILE ⇒ ~/pentesting/rockyou.txt msf6 auxiliary(scanner/ssh/ssh_login) > run [*] 192.168.0.236:22 - Starting bruteforce
		rpcbind	Steps to exploit: a) Scan 111 port more detailed. b) Execute rpcinfo command to get more information c) Execute 'auxiliary/scanner/portmap/portmap_amp' module	I got some info, but couldn't get deeper in time.  msf6 auxiliary(scanner/portmap/portmap_amp) > run [*] Sending Portmap RPC probes to 192.168.0.236→192.168.0.236 (1 hosts) [+] 192.168.0.236:111 - Vulnerable to Portmap RPC DUMP (Program version: 3) amplification: No packet amplification and a 17x, 644-byte bandwidth amplification [+] 192.168.0.236:111 - Vulnerable to Portmap RPC DUMP (Program version: 2) amplification: No packet amplification and a 3x, 108-byte bandwidth amplification [+] 192.168.0.236:111 - Vulnerable to Portmap RPC GETSTAT amplification: No packet amplification and a 7x, 272-byte bandwidth amplification [*] Scanned 1 of 1 hosts (100% complete) [*] Auxiliary module execution completed

ii) Customized program (for MITM)

Attempts were made to identify system, application, and service information via scanning of the ports and doing the reverse engineering of the client with a hex viewer. And then we create the MITM program customized for pen-testing.



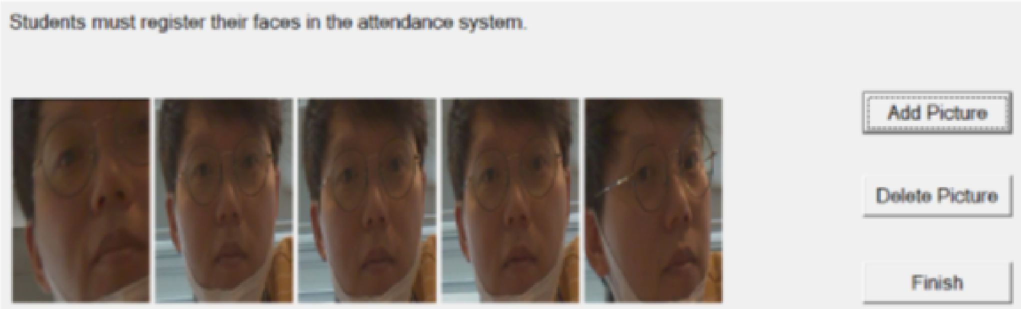
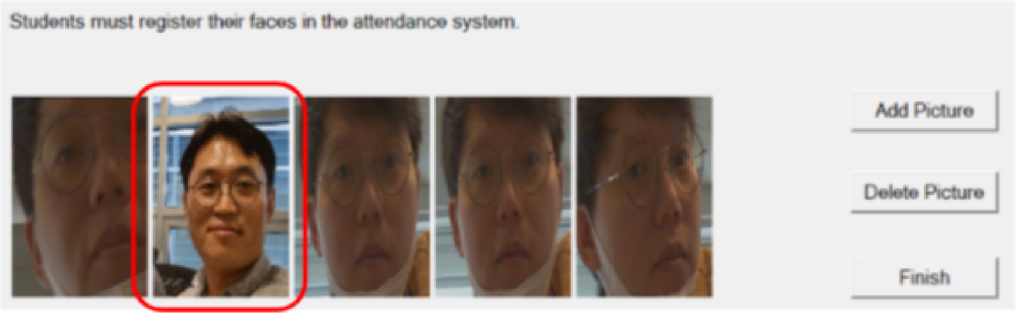
```
End generating TensorRT runtime model
[FaceManager] readFaceDB
filesize: 176
facedbenc len:176 hex:fac35888c5949d9
facedbsign len:1500 hex:308205d806092
facedb verify ok
buffer len:176 hex:01000000000000000b
[FaceManager] readSize : 176 readLen
[FaceManager] loadFaceNet
Parsing Directory: ../imgs
Listening for connections
Listening for connections2
```

Exploitable vulnerabilities discovered were:

- Configuration data on the client was not hidden from being viewed.
- Private keys can be stolen because no encryption applied on the private keys
- Authentication status is not managed on the server side for the requests.
- No privilege checking for user accounts on the server side.

iii) Without pen test tool (for Tampering)

We imagine the attack scenario based on the gathered information. Because this program doesn’t protect image storage, it must have vulnerabilities at that point.

Attack Scenario	
Pre-condition	Attacker gained the access to the server system
Detail steps	< Normal Situation at client side> 1) A normal user login to the client. 2) Add pictures 3) Log out < Attack Point at server side> 4) The attacker replaces one of the images of the user with another one in the server storage
Expected Result	Unauthorized user can pass the attendance system.
Actual Result	The attack was successful and had the expected result. < Before >  Students must register their faces in the attendance system. < After >  Students must register their faces in the attendance system.

4. Risk Assessment

Of the systems scanned, a total of 33 vulnerabilities were found.
These vulnerabilities were categorized and evaluated according to their likelihood of occurrence and the potential damage if they occur.

1) Critical Severity Vulnerability

5 were critical severity vulnerabilities. Critical vulnerabilities require immediate attention. They are relatively easy for attackers to exploit or may provide them with full control of the affected systems.
A list of the critical severity vulnerabilities is provided below:

No	Module	Type	Description
1	Authentication (Server)	Spoofing	An attacker builds a fake client with the stolen key pair of the client and the fake client successfully communicates with the server without log-in procedure. Steps to reproduce: 1) An attacker steals the client certificate and key. 2) Create a fake client program with stolen key pair. 3) Run a fake client and connect to normal server. 4) Send several request messages to the server like add face, get face, get students list, etc.
2	Authentication (Server)	Spoofing	The private key is not encrypted and doesn't have a passphrase, so anybody can use it freely if he steals the key file.
3	Authorization (Server)	Spoofing	An attacker can use a privileged message after login as a normal user. Steps to reproduce: 1) An attacker steals the client certificate and key. 2) Create a fake client program with stolen key pair. 3) Run a fake client and connect to normal server. 4) Log in as a normal user. 5) Send a message which can be used by the administrator only. For example, get students list.
4	Authentication (Client/Server)	Spoofing	An attacker builds a fake server with the stolen key pair of the client and it works with the valid client Steps to reproduce: 1) An attacker steals the client certificate and key. 2) Create a fake server program with stolen key pair. 3) Run a fake server. 4) Lead clients connect to fake servers.

5	Asset (Manipulate the registered student image)	Tampering	If an attacker has access to the system, an attacker can replace the jpg image file of a registered student's face with another one's face image. 1. Get system access 2. Get registered face image names 3. Replace existed image file with new file

2) High Severity Vulnerability

9 were high severity vulnerabilities. High severity vulnerabilities may not provide the security policy of sensitive data to affected systems.

A list of the most frequent high severity vulnerabilities is provided below:

No	Module	Type	Description
6	Face Recognition	Spoofing	In this project, the system can't distinguish between the picture and the real person. So my attempt to recognize me by my photo was successful as below. Steps to reproduce: 1) Execute server/client application 2) Select my selfie on my smartphone and locate the smartphone screen in front of the camera 3) Observe the recognition result
7	Protocol Logic	Tampering	When the live mode is requested and the pre-loaded video file stream is transferred, there is no way to identify in view of protocol.
8	Logging	Repudiation	There is no way to track the activities of the systems when applications are terminated.
9	Authentication (Key)	Information Disclosure	Three databases are encrypted using separated symmetric key file but all key file has the same encryption key.
10	Authentication (IV)	Information Disclosure	According to "6.2 The Cipher Block Chaining Mode" of NIST SP 800-38A which is "Recommendation for Block Cipher Modes of Operation", the IV need not be secret, but it must be unpredictable. However three databases use same IV value and the value of IV can be predictable value(0000000000000000).
11	Asset (Registered student name)	Information Disclosure	The registered student name is stored in Face DB which is encrypted with AES128 CBC, but the student name is also saved as a jpg file name. So, if an attacker has access to the system, they can easily get the registered student name without breaking the Face DB. It can lead to another vulnerability.
12	Asset (Registered student face	Information Disclosure	The registered student face images are stored in the local storage in jpeg format. So, if an attacker has

	image)		access to the system, they can get the registered student face. It can lead to another vulnerability.
13	Authentication (Private key file and certificate file path of server)	Information Disclosure	Using hex edit with the server application, we can know the full path of the private key and certificate of the server .
14	Data encryption	Information Disclosure	When the client requests the student name list to the server, the server transfers all students' name in the raw format.

3) Medium Severity Vulnerability

16 were medium severity vulnerabilities. These vulnerabilities often provide information to attackers that may assist them in mounting subsequent attacks on your network.

A list of the medium severity vulnerabilities is provided below:

No	Module	Type	Description
15	Configuration	Tampering	Since the server key is stored in a USB, that may lead to insecure default behavior if malwares are in the USB
16	Configuration	Tampering	Whenever the socket connection is made with the client newly, the server tried to load the key, certificate and rootca from the USB memory. At that time, if only one of them doesn't exist, the server program goes to crash. So, the USB key must be always attached to Jetson Nano whenever the server runs, and there should not be detached physically and logically. Steps to reproduce: 1) Execute the server program 2) Server is waiting new connection 3) Delete TLS server keypair from USB 4) New TLS connection is arrived from client
17	Protocol Logic	Tampering	Since the first octet of each request message is changed as "00 FF FF FF", the server doesn't respond from any normal client requests. Server codes wait until the remainder part of the request message is arrived. After that, if the client disconnects the network channel, the server will show the error message infinitely and doesn't allow the new network connection.
18	Authentication (TLS cipher suite of client / Private key file name / Certificate file name)	Information Disclosure	Using hex edit with the client application, we can know which cipher suite is used and the key/certificate names are.

19	Authentication (Path of private key and certificate of client)	Information Disclosure	Using hex edit with the client application, we can know the path name of the private key and certificate partially (e.g. "%s\cert\client,key"). But, we know that they are located in the USB memory, so we can guess "%s" is the USB drive name.
20	Authentication (Estimation of administrator's ID)	Information Disclosure	Using hex edit with the server application, we can guess that the administrator ID may be admin or administrator.
21	Configuration	Information disclosure	Server IP/port information is disclosed in clientconf.bin and that may be a start of the attacker's target.
22	Data Encryption	Information Disclosure	Video streaming data is based on JPEG format, but JPEG format is well-known. So, the attacker can easily detect the streaming format and view the sniffed stream data
23	Data Encryption	Information Disclosure	Client transfers the ID & password as a plain text format. They can be revealed easily through MIMT.
24	Face Recognition	DoS	When the camera is out of control, the server is stuck in an infinite loop.
25	Network Connection	DoS	If the client is terminated or the network is disconnected, after the client completes the login procedure with the server, the server displays the error message infinitely and it is impossible to make a new connection.. Following messages are iterated infinitely.
26	Face Recognition	DoS	If a non-image file exists in the image directory, the server aborted. Steps to reproduce: 1) Create .txt file in the image directory. : $ echo 'abcd' > test.txt $ cat test.txt abcd 2) Run the server program
27	Face Recognition	DoS	After the server shows the video file open error, if the client sends the VIDEO start request, the server shows the error message infinitely, and is impossible to serve correctly.
28	Face Recognition	DoS	If the video file play is ended in the test run mode, the server is stuck in an infinite loop.
29	Network Connection & Authentication	DoS	After logging in about 500 times, it is in a state where it can no longer receive messages.
30	Face Recognition	etc	After the camera image is captured to the variable "frame", no code to release it. The memory release code is blocked as the comment.

			178: // frame.release()

4) Low Severity Vulnerability

3 were low severity vulnerabilities. Low severity vulnerabilities do not need to be patch immediately and can be resolved during the next updates maintenance window.

A list of the low severity vulnerabilities is provided below:

No	Module	Type	Description
31	Face Recognition	Tampering	If the file name from the client does not match to the video file in the storage for test-run mode, the server shows the open error message. However, this message is driven from the opencv, not from the server application. It means that the server application doesn't handle this situation.
32	Authentication (TLS cipher suite of server)	Information Disclosure	Using hex edit with the server application, we can know which cipher suite is used.
33	Protocol Logic	etc	The integer data of the protocol doesn't apply the host-to-network conversion and uses the struct type instead of the fixed size. So, 32 bit machines and Mac machines can't communicate with the server. If those machines send the protocol data to the server, the server may be corrupted because of the misunderstood data length.

5. Recommendations

No	Module	Type	Mitigation	Severity
1	Authentication (Server)	Spoofing	The server must manage the login state and credentials of the logged in user.	Critical
2	Authentication (Server)	Spoofing	Key file should be encrypted or have a passphrase. Passphrase should be strong enough against the brute-force attack.	Critical
3	Authorization (Server)	Spoofing	The server must handle the requested message based on the authorization of the logged-in user.	Critical
4	Authentication (Client/Server)	Spoofing	The client must verify the CN field of the delivered server certificate.	Critical
5	Asset (Manipulate the registered student image)	Tampering	The filename and jpg file must be encrypted.	Critical
6	Face Recognition	Spoofing	The face recognition AI model should be enhanced to identify picture and real person.	High
7	Protocol Logic	Tampering	Protocol for the video streaming should have the identifier between video streaming and video file streaming.	High
8	Logging	Repudiation	Must write log in files.	High
9	Authentication (Key)	Information Disclosure		High
10	Authentication (IV)	Information Disclosure	Must use randomly generated value for IV. Must use a different IV value for each db.	High
11	Asset (Registered student name)	Information Disclosure	The filename must be encrypted.	High
12	Asset (Registered student face image)	Information Disclosure	The jpg file must be encrypted.	High
13	Authentication (Private key file and certificate file path of server)	Information Disclosure	Strings related with the security information should be encoded in the data hiding method	High
14	Data Encryption	Information Disclosure	About the user credential information, the data should be transferred after encoded through the data hiding method, not a plain text format.	High

15	Configuration	Tampering	Must check the contents of USB before reading them	Medium
16	Configuration	Tampering	Server must load and check the key, certificate and rootCA at the beginning of the run. No need to load and check them every time of the new channel connection.	Medium
17	Protocol Logic	Tampering	If the length of each request protocol data is too big, server codes should disconnect the channel because it is an abnormal protocol. If the network channel is closed, the server codes should detect it and initialize the flow status.	Medium
18	Authentication (TLS cipher suite of client / Private key file name / Certificate file name)	Information Disclosure	Strings related with the security information should be encoded in the data hiding method	Medium
19	Authentication (Path of private key and certificate of client)	Information Disclosure	Strings related with the security information should be encoded in the data hiding method	Medium
20	Authentication (Estimation of administrator's ID)	Information Disclosure	Strings related with the security information should be encoded in the data hiding method	Medium
21	Configuration	Information disclosure	Should hide server information.	Medium
22	Data Encryption	Information Disclosure	If the JPEG header is common for all video stream data, remove the header and leave the body alone. When the image data is received, merge it to the JPEG header and call the OpenCV drawing API.	Medium
23	Data Encryption	Information Disclosure	About the user credential information, the data must be transferred after encoded through the data hiding method, not a plain text format.	Medium
24	Face Recognition	DoS	Should handle the unexpected case properly.	Medium
25	Network Connection	DoS	If the network channel is closed, the server codes should detect it and initialize the flow status.	Medium
26	Face Recognition	DoS	The program should filter the unsupported file type.	Medium

27	Face Recognition	DoS	The new assignment of the VideoStreamer handler should be checked if it is ok or not. If the assignment is failed, file play should be stopped, not try to read the video stream.	Medium
28	Face Recognition	DoS	When the video file playing ends, the video streamer object should be initialized and the video file should be re-loaded to play again.	Medium
29	Network Connection & Authentication	DoS	When the connection is closed, the ssl context must be initialized to null.	Medium
30	Face Recognition	etc	The code should be uncommented.	Medium
31	Face Recognition	Tampering	When the video file name is given to VideoStreamer class, the file existence should be checked first. For this service, the client doesn't need to give the file name for the remote file in the server.	Low
32	Authentication (TLS cipher suite of server)	Information Disclosure	Strings related with the security information should be encoded in the data hiding method	Low
33	Protocol Logic	etc	Instead of the structure type for the field data, use the fixed size to the field data and host-to-network conversion should be applied.	Low