Phase 2 Report

Team 3

About This Document

Document Information

Issuing authority	Team 3
Team 3 Mentor	Cliff Huff
Team 3 Members	Kyuwoon Kim, Gyeonghun Ro, Wonyoung Chang, Soohyun Yi, Hyejin Oh, Hyungjin Choi, Vibhanshu Dhote

Revision History

Verion	Date	Comment	Author
1.0	2021-06-28	Initial Release	Team 3
1.1	2021-07-020	Final Version Release	Team 3

Purpose

This document is the final phase 2 report by Team 3. We evaluated Team 4's CCTV monitoring system during phase 2. We describe all the security activities we have done and all the vulnerabilities we found during phase 2 in this document.

Assessment Scope

- Team4 Source Code : https://github.com/hijang/lsc_cctv/
- Team4 Software Requirements
- Team4 Software Design
- Team4 Risk Assessment
- Team4 Static analysis
- Team4 Test case
- Team4 Security Requirement Mitigation TCs
- Team4 ReadMe
- Team4 CCTV_ThreatList_STRIDE_and_PnG

Related Documents

Documents related to this document include :

- Team4 Vulnerability list by Team3
- Team4 Assessment_CC_Report by Team3
- Team4 Fuzzing Results by Team3
- Team4 Input Output Analysis by Team3
- Team4 Open Source Vulnerabilities by Team3
- Team4 Penetration Test Report by Team3
- Team4 Port scan report by Team3
- Team4 Security Assessment Report by Nessus by Team3
- Team4 Security Assessment Reports by Team3
- Team4 Static Analysis (RATS) by Team3
- Team4 Threat Analysis by PNG by Team3

Table of Contents

1. Executive Summary

1.1. Summary

In phase 2, the monitoring system provided by Team 4 was evaluated. From the point of view of the evaluation process, it was an element of concern about where to start.

At first, information was obtained through the provided deliverables, and the structure of the system was created by using that information. The design data received from Team 4 was not sufficient, so we had to create a new context diagram and architecture design based on the provided output data.

In order to identify vulnerabilities of Team 4's system, several tools were used. During the process of reviewing identified vulnerabilities, we were able to identify several security holes.

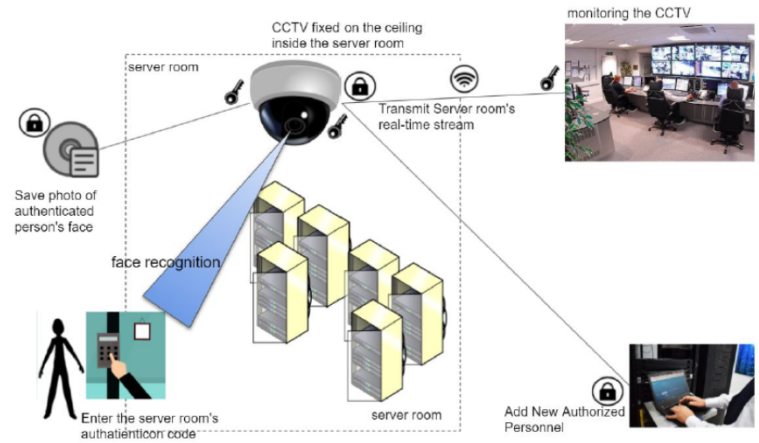
We attempted a system attack, hijacking the system account from a hacker's point of view in various ways.

We also tried to evaluate the system with the requirements of Common Criteria. By reviewing the evaluation requirements of Common Criteria, we can see the assessment target more objectively with the viewpoint of the evaluator. Our target evaluation level was EAL1 of Common Criteria. The goal of EAL 1 is to check if the target system is functional tested.

Finally, some security applications necessary for the Team4 system were summarized and suggested, and the strengths and weaknesses of the Team4 system were organized and provided.

2. Target System Overview

2.1. Main System



Our assessment target is a **CCTV monitoring system** implemented by Team 4. This system is designed for security in the server room.

The following is three major parts in this system :

- **CCTV**: A server application which processes camera image and face recognition.
- **Monitor System**: A client program that receives images from the network and displays them.
- **User Register**: A user registration program for adding and deleting users. It is an auxiliary application of the server side and has no network input/output function.

The basic scenario is that 1) CCTV installed in the server room detects people entering and exiting, and 2) the screen is monitored through the Security Agent and 3) The CCTV administrator performs user registration and deletion for the person entering the server room.

The basic approach by Team 4 was to avoid implementing GUI features and focused on enhancing security of the product instead. Team 4 assumption was that the certifications for secure communication were already installed in both CCTV and monitoring system so that no GUI program was necessary. That affects the attack surface minimum and makes our analysis difficult.

2.2. Users

There are three types of users in this system.

- **Authorized Person**: A person who enters the server room and is subject to CCTV footage. To become an Authorized Person, you must apply for user registration to the Monitoring System Manager in advance.
- **Security Agent**: A person who is looking at the Monitoring System does not do anything other than looking at the screen.
- **Monitoring System Manager**: A person who registers as a user. In the design document, only user registration is specified, but in reality, server application operation, maintenance, management, etc. are all done.

3. Overview of Security Assessment

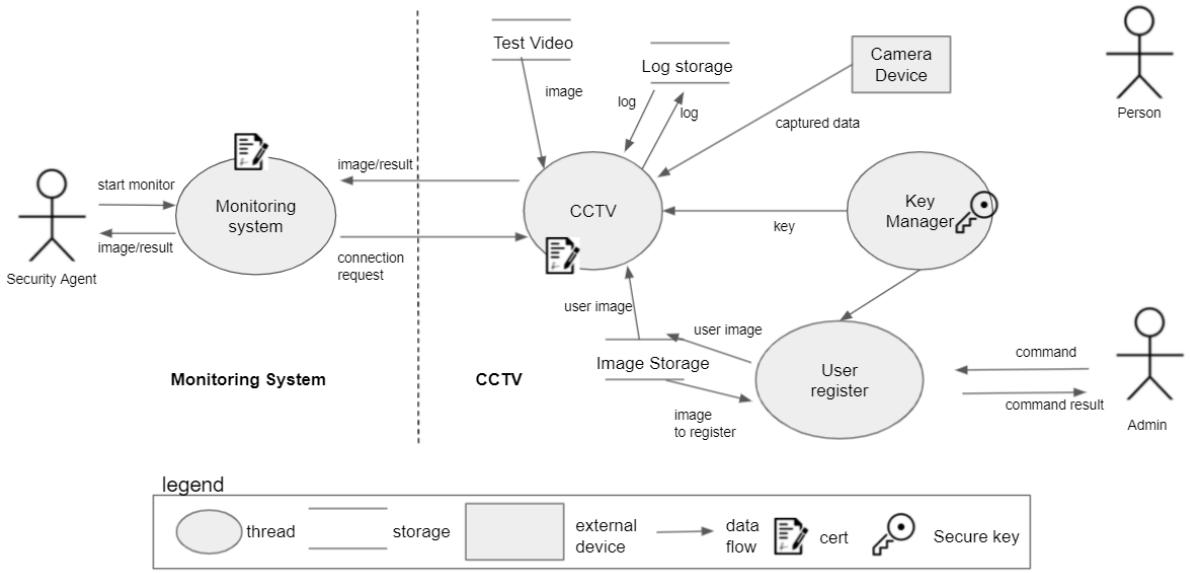
The following figure shows the overall security activities we have done in Phase 2 :

- Code / Design Review
  - Design document/code matching
  - Security code review
  - Checking security reflections in design
  - Threat modeling based on PnG
  - Security policy analysis
  - Program usage analysis
- Static analysis
  - Static analysis by tool
  - 3rd party open source vulnerability check
- Dynamic analysis
  - Fuzz Testing: Zuff, AFL
  - Vulnerability Scanner: nmap, nessus, other tools
  - Data input/output analysis
  - Exploit
  - Test case analysis
- Assessment by Common Criteria
  - Check Requirements for EAL 1

4. Threat Modeling

4.1. Threat Modeling Review of Team 4

Based on the code and design review, we drew data flow diagram to simplify the CCTV monitoring system as follows :



When we compared threat modeling outputs of phase 1 between our team (Team 3) and Team 4, there was not much difference. It is mainly because the basic source code from CMU was the same and both teams derived threats by using the same threat modeling tool, MS Threat modeling tool. So, most of the threats categorized by STRIDE methodology were the same. The difference was that among the derived threats, team 4 focused more on the mitigations for non-repudiation, information disclosure, and escalation of privilege within the limited time, whereas our team (team 3) pays more attention to tampering, information disclosure, and spoofing. The following is the summary of mitigations of CCTV monitoring system by team 4.

What to protect (assets)	Network	User information	Key management	Logging
How to protect (mitigations)	TLS 1.2 based mutual authentication	User image encryption	Key manager	CCTV logs

4.2. Threat Modeling by PnG

Team 4 implemented good security features (mentioned in the previous section) of the CCTV monitoring system in phase 1 and we tried to find security holes. We ran the system and reviewed all the source code of the system and underwent many trials and errors. Based on all the trials, we thought we can show the attack points effectively with PnG methodology. The following personas show the misuse cases which can happen in CCTV monitoring.

	 PnG1 : Mark	 PnG2 :Tom	 PnG3 :John
PnG Description	One of security agents who works in three shifts	Employee who plans to move to rival company	Employee who would like to make his co-worker project go into trouble
Motivation	To gain money for overlooking the unauthorized access	To sell project data	To disrupt IT infrastructure during crucial hours
Skills	Knowledge of "manager" password Knowledge that there is no user authentication mechanism in the system	Knowledge in hacking techniques and tools	Knowledge of image recognition mechanism.
Misuse Cases	1) Login with common "manager" account 2) Add new (unauthorized) users to the system 3) When the new users enter server room, no alarm rings. 4) Delete the users from the system, no one knows he does the mis-behavior.	1) Brute force attack with login cracking tool and obtain "manager" credentials. 2) Login with "manager" account and change his picture with the authorized person's picture 3) When he enters server room, no alarm.	1) Take the picture of authorized person and make mask with it. 2) Put the mask when entering Server room, no Alarm.

5. Open Source Known Vulnerabilities

We noticed that open source known vulnerabilities for the CCTV monitoring system were not surveyed by Team 4. There are lots of open sources used in this project and 34 open source known vulnerabilities identified by our analysis. The remaining known vulnerabilities can be exploited by attackers. To apply the latest security updates is recommended.

This is the summary of known vulnerabilities by each open source.

- Jetson Linux 32.5 : 26 vulnerabilities
- OpenCV 4.1.1 : 6 vulnerabilities
- OpenSSL v1.1.1 : 2 vulnerabilities

The following is the details of open source vulnerabilities and our review comments :

No.	OSS Name / Version	CVE ID	Severity	NVD Score	Description	Comments
1	OpenSSL v1.1.1	CVE-2021-23840	High	7.5	Calls to EVP_CipherUpdate, EVP_EncryptUpdate and EVP_DecryptUpdate may overflow the output length argument in some cases where the input length is close to the maximum permissible length for an integer on the platform.	EVP_CipherUpdate used. but the input length is not close to the maximum length. It is limited to 1024.
2	OpenSSL v1.1.1	CVE-2019-1543	High	7.5	ChaCha20-Poly1305 is an AEAD cipher, and requires a unique nonce input for every encryption operation. RFC 7539 specifies that the nonce value (IV) should be 96 bits (12 bytes).	CCTV does not use ChaCha20-Poly1305 cipher.
3	OpenCV 4.1.1	CVE-2019-5064	HIGH	8.8	An exploitable heap buffer overflow vulnerability exists in the data structure persistence functionality of OpenCV, before version 4.2.0.	cv::FileStorage (cv::JSONParser) did not used.
4	OpenCV 4.1.1	CVE-2019-5063	HIGH	8.8	An exploitable heap buffer overflow vulnerability exists in the data structure persistence functionality of OpenCV 4.1.0.	cv::FileStorage (cv::XMLParser) did not used.
5	OpenCV 4.1.1	CVE-2019-19624	MEDIUM	6.5	An out-of-bounds read was discovered in OpenCV before 4.1.1. Specifically, variable coarsest_scale is assumed to be greater than or equal to finest_scale within the calc()/ocl_calc() functions in dis_flow.cpp.	This flaw can be exploited when a small, carefully crafted image is loaded by an application linked to OpenCV.
6	OpenCV 4.1.1	CVE-2019-16249	MEDIUM	5.3	OpenCV 4.1.1 has an out-of-bounds read in hal_baseline::v_load in core/hal/intrin_sse.hpp when called from computeSSDMeanNorm in modules/video/src/dis_flow.cpp.	This can happen when replaying a malformed video.
7	OpenCV 4.1.1	CVE-2019-15939	MEDIUM	5.9	There is a divide-by-zero error in cv::HOGDescriptor::getDescriptorSize in modules/objdetect/src/hog.cpp.	This can happen when calling cv::HOGDescriptor::getDescriptorSize with cellsize 0.
8	OpenCV 4.1.1	CVE-2019-14493	HIGH	7.5	There is a NULL pointer dereference in the function cv::XMLParser::parse at modules/core/src/persistence.cpp.	cv::XMLParser was not used.
9	Jetson	CVE-20	HIGH	8.2	Trusty (the trusted OS produced by NVIDIA for	Fixed in Jetson Linux

	Linux 32.5	21-3437 2			Jetson devices) driver contains a vulnerability in the NVIDIA OTE protocol message parsing code where an integer overflow in a malloc() size calculation leads to a buffer overflow on the heap, which might result in information disclosure, escalation of privileges, and denial of service.	version 32.5.1. To apply a security patch is needed.
10	Jetson Linux 32.5	CVE-20 21-3437 3	HIGH	7.9	Trusty trusted Linux kernel (TLK) contains a vulnerability in the NVIDIA TLK kernel where a lack of heap hardening could cause heap overflows, which might lead to information disclosure and denial of service.	Fixed in Jetson Linux version 32.5.1. To apply a security patch is needed.
11	Jetson Linux 32.5	CVE-20 21-3437 4	HIGH	7.7	Trusty contains a vulnerability in command handlers where the length of input buffers is not verified. This vulnerability can cause memory corruption, which may lead to information disclosure, escalation of privileges, and denial of service.	Fixed in Jetson Linux version 32.5.1. To apply a security patch is needed.
12	Jetson Linux 32.5	CVE-20 21-3437 5	HIGH	7.7	Trusty contains a vulnerability in all trusted applications (TAs) where the stack cookie was not randomized, which might result in stack-based buffer overflow, leading to denial of service, escalation of privileges, and information disclosure.	Fixed in Jetson Linux version 32.5.1.To apply a security patch is needed. .
13	Jetson Linux 32.5	CVE-20 21-3437 6	HIGH	7.7	Trusty contains a vulnerability in the HDCP service TA where bounds checking in command 5 is missing. Improper restriction of operations within the bounds of a memory buffer might lead to denial of service, escalation of privileges, and information disclosure.	Fixed in Jetson Linux version 32.5.1.To apply a security patch is needed.
14	Jetson Linux 32.5	CVE-20 21-3437 7	HIGH	7.7	Trusty contains a vulnerability in the HDCP service TA where bounds checking in command 9 is missing. Improper restriction of operations within the bounds of a memory buffer might lead to escalation of privileges, information disclosure, and denial of service.	Fixed in Jetson Linux version 32.5.1. To apply a security patch is needed.
15	Jetson Linux 32.5	CVE-20 21-3437 8	HIGH	7.7	Trusty contains a vulnerability in the HDCP service TA where bounds checking in command 11 is missing. Improper restriction of operations within the bounds of a memory buffer might lead to information disclosure, denial of service, or escalation of privileges.	Fixed in Jetson Linux version 32.5.1. To apply a security patch is needed.
16	Jetson Linux 32.5	CVE-20 21-3437 9	HIGH	7.7	Trusty contains a vulnerability in the HDCP service TA where bounds checking in command 10 is missing. The length of an I/O buffer parameter is not checked, which might lead to memory corruption.	Fixed in Jetson Linux version 32.5.1. To apply a security patch is needed.
17	Jetson Linux 32.5	CVE-20 21-3438 0	HIGH	7	Bootloader contains a vulnerability in NVIDIA MB2 where potential heap overflow might cause corruption of the heap metadata, which might lead to arbitrary code execution, denial of service, and information disclosure during secure boot.	Fixed in Jetson Linux version 32.5.1. To apply a security patch is needed.
18	Jetson Linux 32.5	CVE-20 21-3438 1	MEDI UM	6.7	Trusty TLK contains a vulnerability in the NVIDIA TLK kernel function where a lack of checks allows the exploitation of an integer overflow on the size parameter of the tz_map_shared_mem function, which might lead to denial of service, information disclosure, or data tampering.	Fixed in Jetson Linux version 32.5.1.To apply a security patch is needed.
19	Jetson Linux 32.5	CVE-20 21-3438 2	MEDI UM	6.7	Trusty TLK contains a vulnerability in the NVIDIA TLK kernel's tz_map_shared_mem function where an integer overflow on the size parameter causes the request buffer and the logging buffer to overflow, allowing writes to arbitrary addresses within the kernel.	Fixed in Jetson Linux version 32.5.1. To apply a security patch is needed.
20	Jetson Linux	CVE-20 21-3438	MEDI UM	6.4	Bootloader contains a vulnerability in NVIDIA MB2 where a potential heap overflow might lead to denial	Fixed in Jetson Linux version 32.5.1. To apply a

	32.5	3			of service or escalation of privileges.	security patch is needed.
21	Jetson Linux 32.5	CVE-2021-34384	MEDIUM	6.3	Bootloader contains a vulnerability in NVIDIA MB2 where a potential heap overflow could cause memory corruption, which might lead to denial of service or code execution.	Fixed in Jetson Linux version 32.5.1. To apply a security patch is needed.
22	Jetson Linux 32.5	CVE-2021-34385	MEDIUM	6.3	Trusty TLK contains a vulnerability in the NVIDIA TLK kernel where an integer overflow in the calculation of a length could lead to a heap overflow.	Fixed in Jetson Linux version 32.5.1. To apply a security patch is needed.
23	Jetson Linux 32.5	CVE-2021-34386	MEDIUM	6.3	Trusty TLK contains a vulnerability in the NVIDIA TLK kernel where an integer overflow in the calloc size calculation can cause the multiplication of count and size can overflow, which might lead to heap overflows.	Fixed in Jetson Linux version 32.5.1. To apply a security patch is needed.
24	Jetson Linux 32.5	CVE-2021-34387	MEDIUM	6.3	The ARM® TrustZone Technology on which Trusty is based on contains a vulnerability in access permission settings where the portion of the DRAM reserved for TrustZone is identity-mapped by TLK with read, write, and execute permissions, which gives write access to kernel code and data that is otherwise mapped read only.	Fixed in Jetson Linux version 32.5.1. To apply a security patch is needed.
25	Jetson Linux 32.5	CVE-2021-34388	MEDIUM	6.3	Bootloader contains a vulnerability in NVIDIA MB2 where a potential heap overflow might allow an attacker to control all the RAM after the heap block, leading to denial of service or code execution.	Fixed in Jetson Linux version 32.5.1. To apply a security patch is needed.
26	Jetson Linux 32.5	CVE-2021-34389	MEDIUM	5.9	Trusty contains a vulnerability in NVIDIA OTE protocol message parsing code, which is present in all the TAs. An incorrect bounds check leads to a memory leak of a portion of the heap situated after a stream buffer.	Fixed in Jetson Linux version 32.5.1. To apply a security patch is needed.
27	Jetson Linux 32.5	CVE-2021-34390	MEDIUM	5.3	Trusty TLK contains a vulnerability in the NVIDIA TLK kernel function where a lack of checks allows the exploitation of an integer overflow on the size parameter of the tz_map_shared_mem function.	Fixed in Jetson Linux version 32.5.1. To apply a security patch is needed.
28	Jetson Linux 32.5	CVE-2021-34391	MEDIUM	5.3	Trusty TLK contains a vulnerability in the NVIDIA TLK kernel's tz_handle_trusted_app_smc function where a lack of integer overflow checks on the req_off and param_ofs variables leads to memory corruption of critical kernel structures.	Fixed in Jetson Linux version 32.5.1. To apply a security patch is needed.
29	Jetson Linux 32.5	CVE-2021-34392	MEDIUM	4.4	Trusty TLK contains a vulnerability in the NVIDIA TLK kernel where an integer overflow in the tz_map_shared_mem function can bypass boundary checks, which might lead to denial of service.	Fixed in Jetson Linux version 32.5.1.To apply a security patch is needed.
30	Jetson Linux 32.5	CVE-2021-34393	MEDIUM	4.2	Trusty contains a vulnerability in TSEC TA which deserializes the incoming messages even though the TSEC TA does not expose any command. This vulnerability might allow an attacker to exploit the deserializer to impact code execution, causing information disclosure.	Fixed in Jetson Linux version 32.5.1. To apply a security patch is needed.
31	Jetson Linux 32.5	CVE-2021-34394	MEDIUM	4.2	Trusty contains a vulnerability in all TAs whose deserializer does not reject messages with multiple occurrences of the same parameter. The deserialization of untrusted data might allow an attacker to exploit the deserializer to impact code execution.	Fixed in Jetson Linux version 32.5.1. To apply a security patch is needed.
32	Jetson Linux 32.5	CVE-2021-34395	LOW	3.9	Trusty TLK contains a vulnerability in its access permission settings where it does not properly restrict access to a resource from a user with local privileges, which might lead to limited information disclosure and limited denial of service.	Fixed in Jetson Linux version 32.5.1. To apply a security patch is needed.

33	Jetson Linux 32.5	CVE-2021-34396	LOW	3	Bootloader contains a vulnerability in access permission settings where unauthorized software may be able to overwrite NVIDIA MB2 code, which would result in limited denial of service.	Fixed in Jetson Linux version 32.5.1.To apply a security patch is needed.
34	Jetson Linux 32.5	CVE-2021-34397	LOW	1.9	Bootloader contains a vulnerability in NVIDIA MB2, which may cause free-the-wrong-heap, which may lead to limited denial of service.	Fixed in Jetson Linux version 32.5.1. To apply a security patch is needed.

6. Vulnerability Scanner

6.1. NMAP

We used nmap for open port scanning. Three open ports were available in the target device. We used this information to attack the target.

```
┌──(kali㉿kali)-[~]
└─$ nmap -sV 192.168.0.106
Starting Nmap 7.91 ( https://nmap.org ) at 2021-06-24 23:06 EDT
Nmap scan report for 192.168.0.106
Host is up (0.0084s latency).
Not shown: 997 closed ports

PORT     STATE SERVICE       VERSION
22/tcp   open  ssh           OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux;
protocol 2.0)
111/tcp  open  rpcbind       2-4 (RPC #100000)
3389/tcp open  ms-wbt-server xrdp

Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
Nmap done: 1 IP address (1 host up) scanned in 17.24 seconds
```

This is the nmap script we run against port 3389. We couldn't find any issue from port 3389. The other investigation and findings about port 22 and 111 are mentioned in Chapter 11 and 12 respectively.

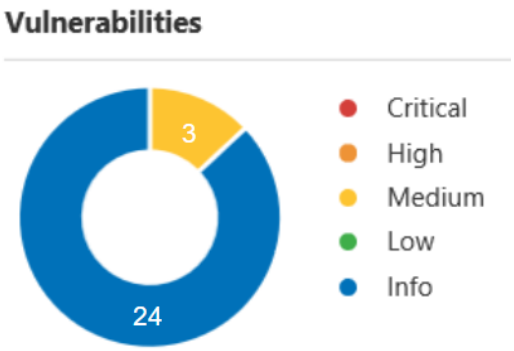
```
┌──(kali㉿kali)-[/usr/share/nmap/scripts]
└─$ nmap -p 3389 --script rdp-enum-encryption 192.168.0.106
Starting Nmap 7.91 ( https://nmap.org ) at 2021-06-25 03:18 EDT
Nmap scan report for 192.168.0.106
Host is up (0.013s latency).

PORT     STATE SERVICE
3389/tcp open  ms-wbt-server
| rdp-enum-encryption:
|   Security layer
|     CredSSP (NLA): SUCCESS
|     CredSSP with Early User Auth: SUCCESS
|     Native RDP: SUCCESS
|     RDSTLS: SUCCESS
|     SSL: SUCCESS
|   RDP Encryption level: High
|     128-bit RC4: SUCCESS
|_  RDP Protocol Version:  RDP 5.x, 6.x, 7.x, or 8.x server

Nmap done: 1 IP address (1 host up) scanned in 2.90 seconds
```

6.2. Nessus

We ran Nessus network vulnerability scanner against CCTV monitoring system. The total 27 vulnerabilities detected in total, 24 vulnerabilities are just information, not the vulnerability.



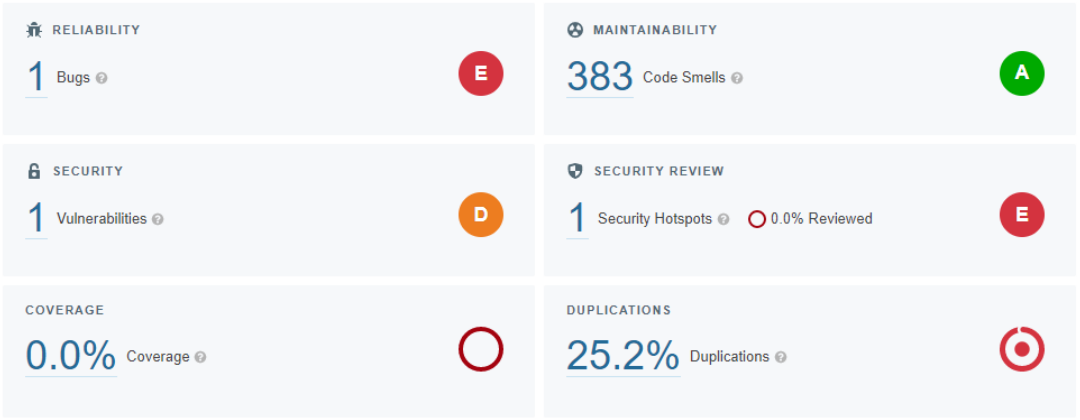
Here are the details of three medium vulnerabilities. Man in the middle weakness (1st item) was further investigated and mentioned in attack point analysis. SSL related vulnerabilities (2nd and 3rd item) were not meaningful because they were related to project constraints.

No.	Name	Severity	Description	Solution
1	Microsoft Windows Remote Desktop Protocol Server Man-in-the-Middle Weakness	Medium	This flaw exists because the RDP server stores a hard-coded RSA private key in the mstlsapi.dll library. Any local user with access to this file (on any Windows system) can retrieve the key and use it for this attack.	- Force the use of SSL as a transport layer for this service if supported, or/and - Select the 'Allow connections only from computers running Remote Desktop with Network Level Authentication' setting if it is available.
2	SSL Self-Signed Certificate	Medium	The X.509 certificate chain for this service is not signed by a recognized certificate authority.	Purchase or generate a proper SSL certificate for this service
3	SSL Certificate Cannot Be Trusted	Medium	The top of the certificate chain sent by the server might not be descended from a known public certificate authority.	Purchase or generate a proper SSL certificate for this service.

7. Static Analysis

7.1. SonarCloud

For static analysis, a sonarcloud was used and 1 bug detected and 1 vulnerability detected by the security perspective. Also, 383 code smell type items were meaningless.



Here are the details of found items by sonarcloud. SSL connection vulnerability was further investigated and mentioned in attack point analysis. Printf-style format string was not meaningful because typecasting was not a critical bug. The rest of 383 items were code smell type, not the vulnerability.

Type	File Path	Description	Line	Link
Vulnerability	MonitoringSystem/SslConnect.cpp	Enable server hostname verification on this SSL/TLS connection	52	https://sonarcloud.io/project/issues?id=chj1986_lsc_cctv&open=AXo7pUX0xwP7BFHx5ihC&resolved=false&types=VULNERABILITY
Bug	CCTV/FakeCCTV.cpp	Printf-style format strings should not lead to unexpected behavior at runtime	112	https://sonarcloud.io/code?id=chj1986_lsc_cctv&selected=chj1986_lsc_cctv%3ACCTV%2FFakeCCTV.cpp&line=112

7.2. Rats

Also, RATs was used for static analysis, the total 41 items detected in total, 11 defects are high, 12 defects are medium and 18 defects are low. However, no remarkable defect was found.

Tools	Total Detected	High	Medium	Low	Remark
RATs	41	11	12	18	- No remarkable defect found.

No.	Severity	Issue	Description	File	Lines
1	High	lstrcpy	Check to be sure that argument 2 passed to this function call will not copy more data than can be handled, resulting in a buffer overflow.	/lsc_cctv//Common/SG_InputBox.cpp	51
2	High	wprintf	Check to be sure that the non-constant format string passed as argument 1to this function call does not come from an untrusted source that could have added formatting characters that the code is not prepared to handle.	/lsc_cctv//Common/utils.cpp	79
3	High	strncat	Consider using strlcat() instead.	/lsc_cctv//LgFaceRecDemoTCP_Jetson_NanoV2/src/logger.cpp	89, 90, 91, 92
4	Medium	getchar	Check buffer boundaries if calling this function in a loop and make sure you are not in danger of writing past the allocated space.	/lsc_cctv//LgFaceRecDemoTCP_Jetson_NanoV2/src/faceNet.cpp	9

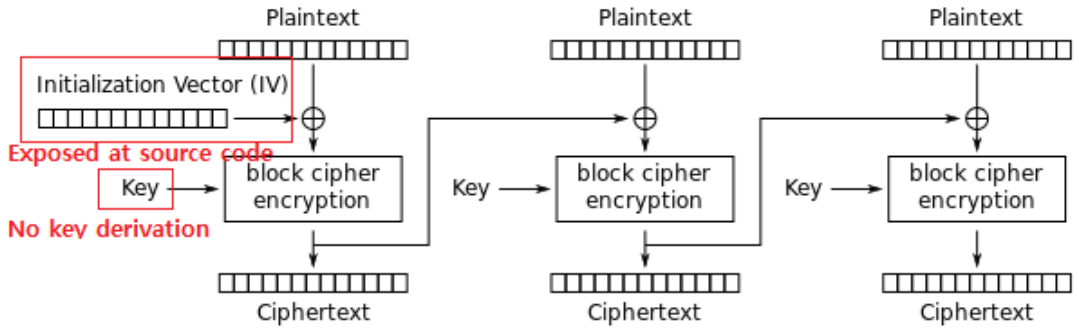
8. Hard-Coded Credentials

8.1. Hard-coded Initialization Vector

This is the code in lsc_cctv/LgFaceRecDemoTCP_Jetson_NanoV2/src/cctvCrypto.cpp. In line 13, The value of Initialization Vector is hard-coded.

```
1   #include "cctvCrypto.h"
2   #include "key_manager.h"
3
4   int do_crypt_file(const char *src, const char *dest, int mode) {
5     FILE *in, *out;
6     int res = 1;
7     unsigned char inbuf[1024], outbuf[1024 + EVP_MAX_BLOCK_LENGTH];
8     int in_len, out_len;
9     EVP_CIPHER_CTX *ctx;
10    const char desc[] = "fk";
11
12    unsigned char key[256] = { 0, };
13    unsigned char iv[] = "1234567887654321";
14    int klen = 0;
```

In CCTV, AES-CBC-128 algorithm is used for block encryption. It is practically difficult to hack the system with this algorithm. However, the Initialization Vector used for CBC is exposed in the source code, and PBKDF (password-based key derivation functions, NIST SP 800-132) suggested by NIST is not applied. These don't directly lead to data leak, but hackers can reduce the Iteration effect of AES so the attack can be made faster.



Cipher Block Chaining (CBC) mode encryption

The Initialization Vector is also exposed by the strings command at linux shell as follows :

```
$ strings LgFaceRecDemoTCP_Jetson_NanoV2  | grep 1234
1234567887654321
```

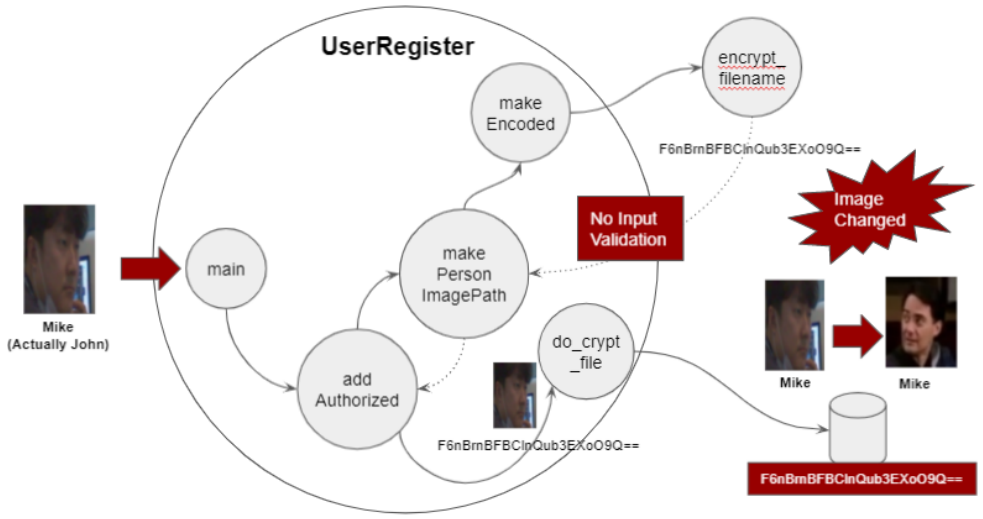
8.2. Recommendation

The initialisation vector (IV) during the encryption should be generated by a pseudorandom number generator instead of hard-coded. The purpose of IV is to decrease the possibility of the same plaintext/ciphertext pairs, which makes it more difficult to decrypt the ciphertext without the key.

9. Spoofing Attacks

9.1. Spoofing against User Register Program

If the server access is possible, pay attention to the file created after user registration on the server. The file name itself is encrypted, so it cannot be used arbitrarily. However, the user name is used for creating the encrypted file name in the user registration process, and if another picture is registered with the same name, the file can be replaced with the original file name. Therefore, if one user is normally registered and a second user is registered with the same name, when the second user enters, the system recognizes as if the first user entered.



1) Register the first user (Mike) as Mike

```
./UserRegister add Mike.jpg Mike
```

2) Mike.jpg→ “F6nBrnBFBC1n!ub3EXoO9Q==”

```
drwxrwxr-x 13 cctv cctv   4096 Jun 22 00:38  ..
-rw-rw-r--  1 cctv cctv 103504 Jun 22 02:51 'F6nBrnBFBClnQub3EXoO9Q=='
```

3) Register the second user(John) as mike

```
./UserRegister add John.jpg Mike
```

4) Actual File : John.jpg, User/File Name : Mike → “F6nBrnBFBC1n!ub3EXoO9Q==”

```
drwxrwxr-x 13 cctv cctv  4096 Jun 22 00:38  ..
-rw-rw-r--  1 cctv cctv 27280 Jun 22 02:53 'F6nBrnBFBClnQub3EXoO9Q=='
```

5) Since there is no duplicate check for names already registered in the system, the registered images are replaced. As a result, John can enter the server room pretending to be Mike.

9.2. Face Recognition Errors

We also found an error in the face recognition of the camera. We assume that various types of fake faces can be exposed to the camera, such as the picture shown on the screen. If the camera recognizes it as a real person in the absence of the Security Agent, it can be a big problem. Attacker puts the fake face with the authorized person's image and can try to enter the server room.

This is the scenarios we consider :

1. No face (Make face invisible)
2. Printed image
3. Displayed image (Replay image with pad or smartphone)
4. Glasses (Wearing glasses to look like a different person)
5. Dust mask
6. Rigid mask (mask made of hard material)
7. Flexible paper mask (like rubber face mask)
8. Fake head

In our experiment, the possible attacks against the CCTV monitoring system were case 1, 2, 3, 4, 5. In case 1, 4, and 5, the modified face could not be recognized as a face. These attacks make no log (like, in/stay/out) on the system. In case 2 and 3, the displayed picture was recognized as another person's face.

9.3. Recommendation

To apply two factor authentication will reinforce the security of the system in depth.

10. Certificate Attacks

10.1. Certificate Purpose Issue

In the code block which handles the certificate, there is developer code remaining, which does not check certificate name. And the purpose of the certificate is so wide. We tried to test to change the client certificate to a server certificate.

server.crt is the server's certificate, and it can be used on the client because both SSL client/server are set to Yes for purpose.

```
$ openssl x509 -in server.crt -purpose -noout
Certificate purposes:
SSL client : Yes
SSL client CA : No
SSL server : Yes
SSL server CA : No
```

We changed the name of server.crt into client.crt and we tried to use it in the host pc and it was successful. When you see the following log, you can see client.crt is actually server.crt (See OU parameter)

```
Client's crt subject : /C=KR/ST=Seoul/L=Gangnam/O=LGE/OU=4tential server unit/CN=*.4tentia

Client's crt issuer : /C=KR/ST=Seoul/L=Gangnam/O=LGE/OU=SecSpecialist/CN=4tential CA Root

Verifying client crt is success.
```

10.2. RootCA Verification Issue

We noticed that there is no part to check the rootca certificate in the application. So we try to change the RootCA certificate of the CCTV monitoring system with the RootCA certificate of our team and test it. We assumed that the CCTV keyring could be replaced by a hacker. We changed all the certificates for both server and client. Even though both certificates were replaced, the program did not notice the problem at all.

Installed a malicious certificate on the server and a malicious rootca certificate on the client. communication is normal as follows:

```
Tring to connect(1)...
Server certificate:
        subject: /C=KR/ST=Seoul/L=Seoul/O=LG/OU=team3/CN=team3 server/emailAddress=cmu-team3@lge.com
        issuer: /C=KR/ST=Seoul/L=Seoul/O=LG/OU=team3/CN=team3 rootca/emailAddress=cmu-team3@lge.com
```

Installed a malicious certificate on the client and a malicious rootca certificate on the server. communication is normal.

```
[acceptConnection:179] : Client's crt subject : /C=KR/ST=Seoul/L=Seoul/O=LG/OU=team3/CN=team3 client/emailAddress=cmu-team3@lge.com
[acceptConnection:182] : Client's crt issuer : /C=KR/ST=Seoul/L=Seoul/O=LG/OU=team3/CN=team3 rootca/emailAddress=cmu-team3@lge.com
```

10.3. MITM attack

A monitoring system (client) certificate was exported from the PC and a man-in-the-middle attack can be attempted using this. The screen sent by the server can be intercepted by a man in the middle, and the man in the middle can send another screen to the monitoring system. It is possible for an intermediary to use the client certificate without permission and to connect with the client by using it as if it were a server certificate. In addition, this is related to TLS version control, and it leads to a TLS 1.0 connection from the middleman to both Server/Client, exposing the risk on the network.

This is the network packet capture for default connections.

No.	Time	Source	Destination	DPort	Protocol	Length	Info
32	1.831721	192.168.0.243	192.168.0.106	5000	TLSv1.3	347	Client Hello
34	1.846724	192.168.0.106	192.168.0.243	55473	TLSv1.3	1514	Server Hello, Change Cipher Spec, Application Data, Application Data
35	1.846724	192.168.0.106	192.168.0.243	55473	TLSv1.3	568	Application Data, Application Data, Application Data
37	1.849316	192.168.0.243	192.168.0.106	5000	TLSv1.3	1216	Change Cipher Spec, Application Data, Application Data, Application Data
38	1.865309	192.168.0.106	192.168.0.243	55473	TLSv1.3	1061	Application Data
40	1.934438	192.168.0.106	192.168.0.243	55473	TLSv1.3	1061	Application Data
43	2.017376	192.168.0.106	192.168.0.243	55473	TLSv1.3	80	Application Data

When you click the packet, you can see that this application is designed to support various TLS versions, but among them, it is connected to TLS1.3.

```
▾ Extension: supported_versions (len=9)
    Type: supported_versions (43)
    Length: 9
    Supported Versions length: 8
    Supported Version: TLS 1.3 (0x0304)
    Supported Version: TLS 1.2 (0x0303)
    Supported Version: TLS 1.1 (0x0302)
    Supported Version: TLS 1.0 (0x0301)
```

Create a man-in-the-middle and establish a connection to the server and client. And force both connections to TLS1.0 with this command in the host PC.

```
$ openssl s_client -connect 192.168.0.106:5000 -cert client.crt -key client.key -CAfile
rootca.crt -tls1
```

Data is being exposed from intermediaries and network security is not working as expected, not TLS 1.3 but TLS 1.0.

No.	Time	Source	Destination	DPort	Protocol	Length	Info
4	2.905786	192.168.0.154	192.168.0.106	5000	TLSv1	158	Client Hello
5	2.923928	192.168.0.106	192.168.0.154	62084	TLSv1	1514	Server Hello, Certificate
6	2.923928	192.168.0.106	192.168.0.154	62084	TLSv1	355	Server Key Exchange, Certificate Request, Server Hello Done
7	2.928480	192.168.0.154	192.168.0.106	5000	TLSv1	1202	Certificate, Client Key Exchange, Certificate Verify, Change Cipher Spec, Encrypted Handsh
8	2.950565	192.168.0.106	192.168.0.154	62084	TLSv1	1060	New Session Ticket, Change Cipher Spec, Encrypted Handshake Message
10	3.048468	192.168.0.106	192.168.0.154	62084	TLSv1	136	Application Data, Application Data
11	3.048468	192.168.0.106	192.168.0.154	62084	TLSv1	1514	Application Data

10.4. Recommendation

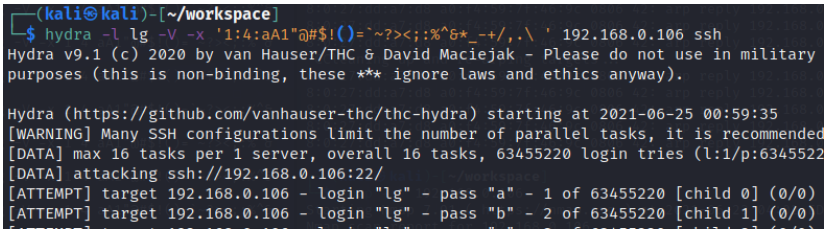
The use of certificates is limited to the purpose of the certificate and the verification of the certificate is required.

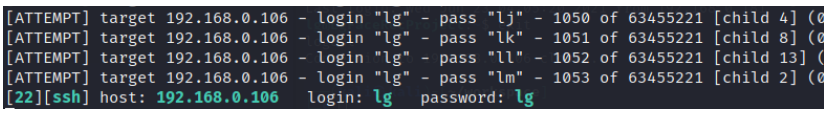
11. Brute Force Attack

11.1. Login cracking with Hydra

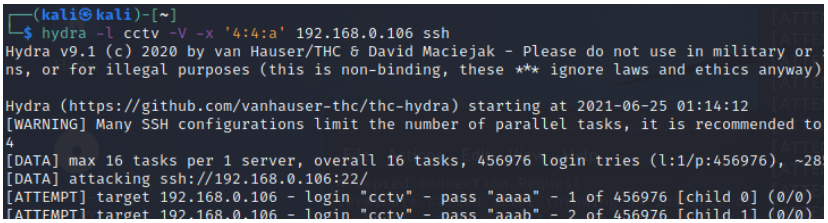
Hydra is a parallelized login cracker which supports numerous protocols to attack. It is very fast and flexible, and new modules are easy to add. It supports various protocols like HTTPS, LDAP, SSH, MySQL etc. We used Hydra for brute-force attack against ssh.

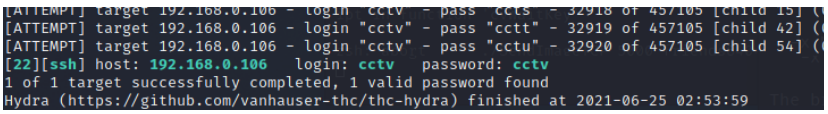
- We try to attack the target with the username lg : Got the password in 1053 attempts.





- We try to attack the target with the username cctv : Got the password in 32920 attempts.





11.2. Recommendation

Here is the security password policy to consider :

1. Enforce Password History policy : The Enforce Password History policy will set how often an old password can be reused.

2. Minimum Password Age policy : This policy determines how long users must keep a password before they can change it.

3. Maximum Password Age policy : The Maximum Password Age policy determines how long users can keep a password before they are required to change it.

4. Minimum Password Length policy : This policy determines the minimum number of characters needed to create a password

5. Passwords Must Meet Complexity Requirements policy

- Passwords can't contain the user name or parts of the user's full name, such as their first name.
- Passwords must use at least three of the four available character types: lowercase letters, uppercase letters, numbers, and symbols.

6. Reset Password : The local administrator password should be reset every 180 days for greater security and the service account password should be reset at least once a year during maintenance time.

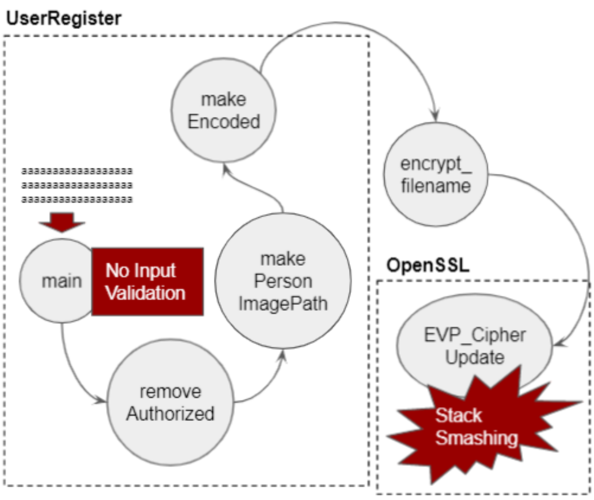
7. Use Strong Passphrases : Strong passphrases with a minimum of 15 characters should always be used to protect domain administrator accounts.

12. Stack Buffer Overflow

12.1. Buffer Overflow

When registering a user, the name is received as a command argument. Although a 1024-byte buffer is used, a stack smash occurs when a long name is entered.

< Stack Smashing (User Add/Remove) >



1) User Add : We try to register a user name with a very long string.

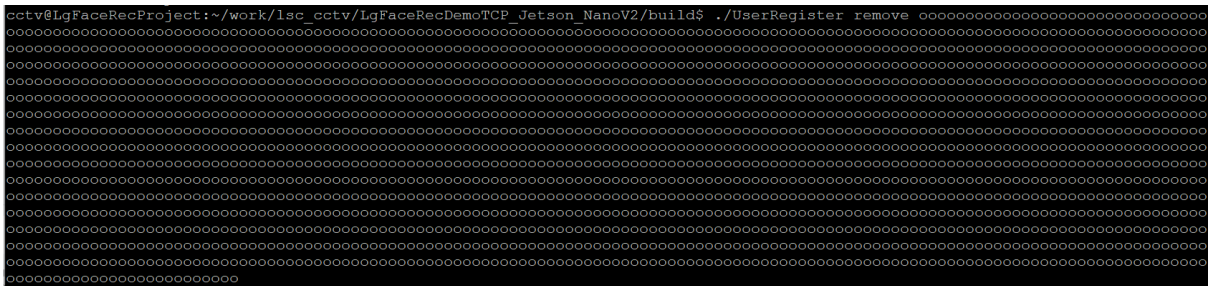
```
cctv@LgFaceRecProject:~/work/lsc_cctv/LgFaceRecDemoTCP_Jetson_NanoV2/build$ ./UserRegister add Mike.jpg aaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaa
```

Stack smashing occurs as follows in the process of encrypting the input name :

```
End generate rnet runtime models
rawName = ../mtCNNModels/det3_relu.engine
size1925584
size1925584
UNKNOWN: Deserialize required 45888 microseconds.

End generating TensorRT runtime models
imageStorePath: /home/cctv/work/lsc_cctv/LgFaceRecDemoTCP_Jetson_NanoV2/imgs
link user to session keyring
*** stack smashing detected ***: <unknown> terminated
Aborted (core dumped)
```

2) User Remove : We try to remove a user name with a very long string.



Stack smashing occurs as follows in the process of encrypting the input name :

```
imageStorePath: /home/cctv/work/lsc_cctv/LgFaceRecDemoTCP_Jetson_NanoV2/imgs
link user to session keyring
*** stack smashing detected ***: <unknown> terminated
Aborted (core dumped)
```

The symptom above happens because of the following cctv source code. This code makes the buffer overflow. Variable outbuf has limit size in stack, buf encrypted data can be written over the limit.

```
char* encrypt_filename(const char *filename) {

  unsigned char outbuf[1024 + EVP_MAX_BLOCK_LENGTH] = {0,};

  ...

  if (!EVP_CipherUpdate(ctx, outbuf, &out_len, (const unsigned char*)filename,
strlen(filename))) {

    printf("EVP_CipherUpdate error.\n");

    goto exit;

  }
```

12.2. Recommendation

The logic to validate the length of username should be implemented in User Register Application.

13. Denial of Service Attack

13.1. RPC port attack with Metasploit

The Metasploit framework is a very powerful tool which can be used by hackers to probe systematic vulnerabilities on networks and servers. Because it's an open-source framework, it can be easily customized and used with most operating systems. With Metasploit, we can use ready-made or custom code and introduce it into a network to probe for weak spots.

In this project, Jetson Linux 32.5 is used and libtirpc is one of the modules inside the kernel. There was a Denial Of Service vulnerability (CVE-2017-8779) in libtirpc 1.0.1, 1.0.2-rc and 1.0.2-rc3. When we checked, Jetson Linux 32.5 included 1.0.10 version of libtirpc which is already patched.

But, when we tried with rpcbomb exploit in mfsconsole, the video streaming of CCTV monitoring System stopped while running the exploit which means our DOS attack was successful. But we observe no crashes during execution. When we stop running the exploit, the video streaming plays back to normal again.

```
Metasploit tip: Use help <command> to learn more
about any command

msf6 > use auxiliary/dos/rpc/rpcbomb
msf6 auxiliary(dos/rpc/rpcbomb) > set rhosts 192.168.0.106
rhosts ⇒ 192.168.0.106
msf6 auxiliary(dos/rpc/rpcbomb) > exploit
```

13.2. Recommendation

The unused port should be closed after release.

14. Protocol robustness testing

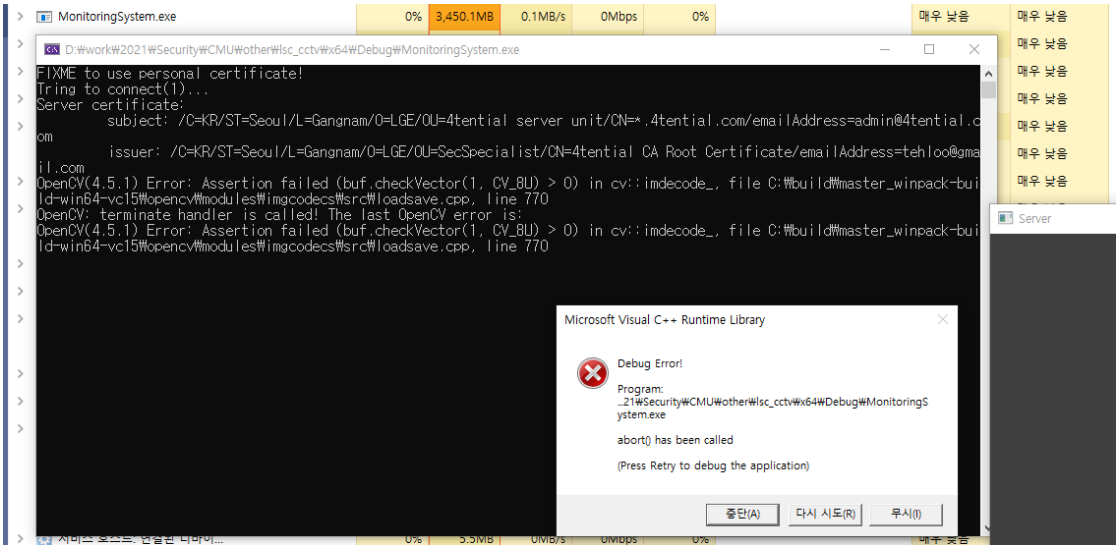
14.1. Protocol robustness testing

We tested the robustness of the network. In the CCTV monitoring system, all the data only flows from the server's module, and the network uses TLS, so there is no way to verify the input data in the client.

In our testing, we created 4G size data with python script on the client side and tried to pass it to the CCTV monitoring system as follows.

```
> type x2.py
import sys
sys.stdout.buffer.write(b'\xFF'* 4 + b'\xFF'* 4294967295)
#Actually, Python does not recognize 4294967295, so It need to write it as multiple
parts.
#The image size is FFFFFFFF bytes, and the dummy data is prepared
#as much as the corresponding size.
> py x2.py > input2
> type input2 | openssl s_server -cert client.crt -key client.key -CAfile rootca.crt
-port 5000
Using default temp DH parameters
ACCEPT    ←  At this time, run client application
ERROR
shutting down SSL
CONNECTION CLOSED
```

Program is stopped for a long time and assertfail occurs in opencv.



14.2. Simple Client Program

This part is not exactly a vulnerability but the comments about implementation. Since this system sends a JPEG image as soon as it is connected to the server, we can easily obtain a jpeg image without a monitoring system as long as you have a certificate.

```
openssl s_client -connect 192.168.0.106:5000 -cert client.crt -key client.key -CAfile
rootca.crt -noservername -quiet -verify_quiet > output.bin
```

Program is stopped for a long. output.bin has the following structure.

[SIZE1=4 BYTE][JPEG SIZE1][SIZE2=4 BYTE][JPEG SIZE2]....

Therefore, if you add a simple parser that can extract size and jpeg, you can use almost all the functions of the monitoring system without the monitoring system.

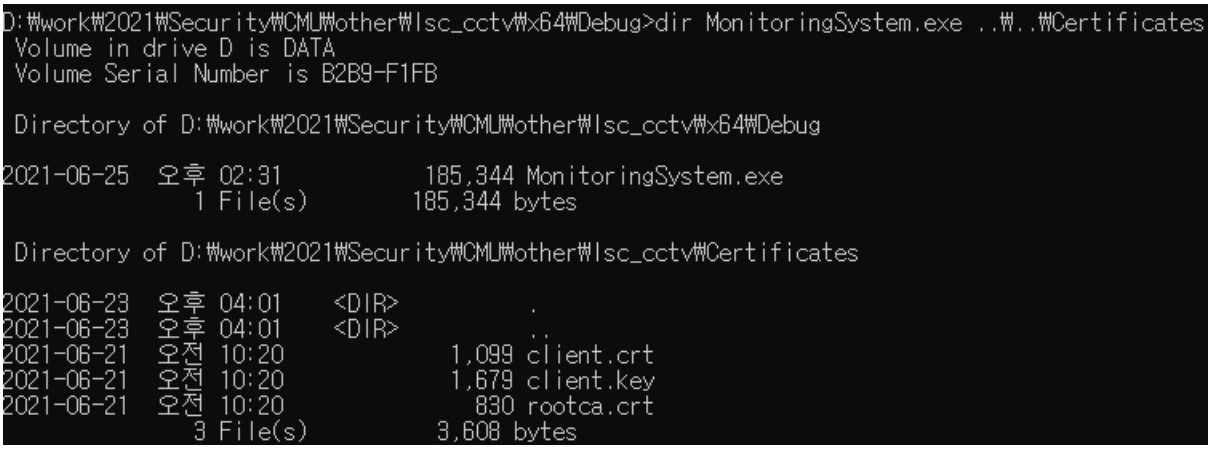
14.3. Recommendation

Input validation against file size is required.

15. Deployment Issues

15.1. Deployment Issues

When distributing the program, Tean 4 distributed source code rather than as an executable file. A user should have to compile it and can create an executable file arbitrarily. This means that when a certain user uses an executable file, the user cannot know whether it is a legitimate file or not. Therefore, when distributing the program, the signed executable file or executable file and the hash value to check it must be provided together. In addition, since Release mode is not defined during binary build, it is compiled in Debug mode. In this case, it is not optimized and all debug symbols are exposed, increasing the possibility of disassembler analysis.



15.2. Recommendation

The distribution file should be released in Release mode.

16. Fuzz Testing

16.1. zzuf

We used zzuf to perform fuzz testing against the CCTV monitoring system.

16.1.1. Video Fuzzing with zzuf

1) Test Data Generation & Execution : The following command was used to generate 100 mutated video files.

```
$ zzuf -s 0:100 -q -C 0 ./LgFaceRecDemoTCP_Jetson_NanoV2 ./friends640x480.mp4
```

2) Test Result : The following was the abnormal termination observed after execution.

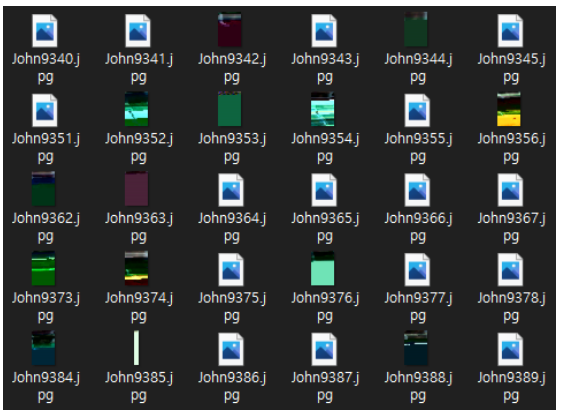
```
$ zzuf[s=17,r=0.004]: signal 11 (SIGSEGV)
$ zzuf[s=56,r=0.004]: signal 11 (SIGSEGV)
$ zzuf[s=75,r=0.004]: signal 11 (SIGSEGV)
$ zzuf[s=77,r=0.004]: signal 11 (SIGSEGV)
$ zzuf[s=91,r=0.004]: signal 11 (SIGSEGV)
```

16.1.2. Image Fuzzing with zzuf

1) Test Data Generation : The following shell script was used to generate 10000 manipulated jpeg image files using the zuff tool.

```
for i in {0..10000}
do
 zzuf -s$i -r.001 < ../John.jpg > John$i.jpg
done
```

This is a snapshot of the result after test data generation was completed.



2) Test Execution : The following shell script was used to register users 10000 times with 10000 malformed jpg images.

```
for f in *.jpg
do
  echo -n "$f start" >> john.log
  ../UserRegister add $f john
  echo " result : $?" >> john.log
done
```

3) Test Result : There were two types of termination after running the test script.

① **Return 0 (Normal Termination)**

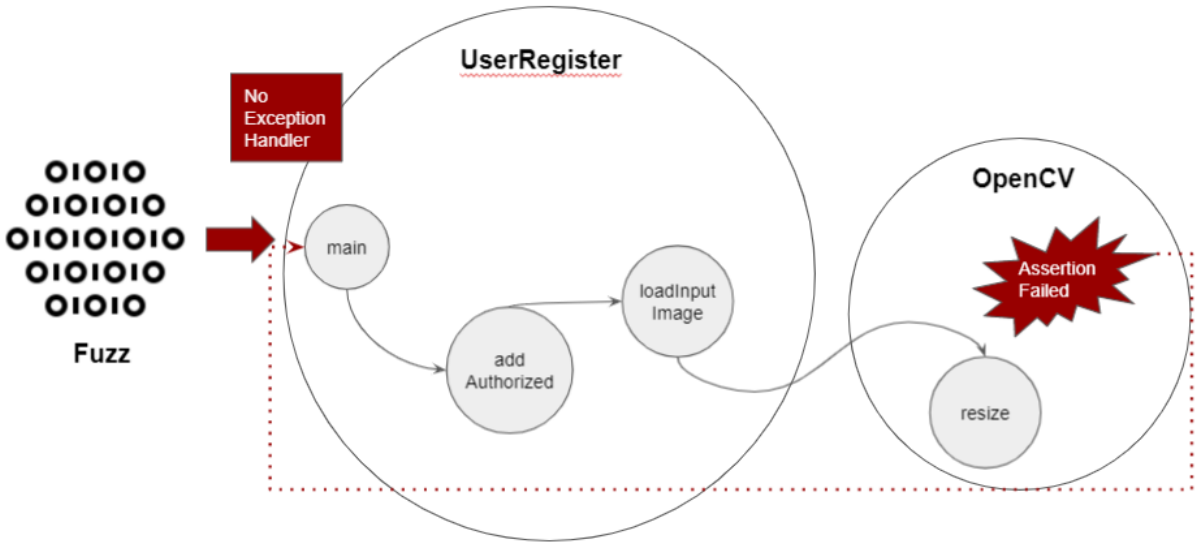
When the header is recognizable, it is recognized as an image file, even though the JPEG image is damaged. The program ended normally through error handling in CCTV.

② **Return 134 (Abnormal Termination)**

When the header of the JPEG image is damaged, an exception occurs because it is not recognized as an image in openCV. The program was terminated due to an exception in OpenCV. Because CCTV did not handle the exception separately, the program ended with an abort like the screen capture below. For handling this error properly, the exception handling logic to the User Register program must be considered and implemented.

```
End generating TensorRT runtime models
terminate called after throwing an instance of 'cv::Exception'
  what():  OpenCV(4.5.1) /home/lg/opencv/modules/imgproc/src/resize.cpp:4051: error: (-215:Assertion failed) !ssize.empty()
 function 'resize'

Aborted (core dumped)
```



16.1.3. The Modification of AI Engine Data with zzuf

We checked whether the CCTV monitoring system detects the trials against the modification of facenet and mtcnn engine files Since it is very difficult to manipulate the engine file directly, we made the mutated engine data by using zzuf and started the program. The program ended with a segmentation fault.

```
$ mv facenet.engine facenet.engine.org
$ ./zzuf -s0 -r0.004 < facenet.engine.org > facenet.engine
$ cd ../build
$ ./LgFaceRecDemoTCP_Jetson_NanoV2 ~/friends640x480_Trim.mp4
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 - ::InstanceNormalization_TRT version 1
link user to session keyring
link user to session keyring
size48844547
ERROR: coreReadArchive.cpp (31) - Serialization Error in verifyHeader: 0 (Magic tag does not match)
ERROR: INVALID_STATE: std::exception
ERROR: INVALID_CONFIG: Deserialize the cuda engine failed.

Segmentation fault (core dumped)
$
```

16.2. AFL

We chose AFL based on its higher code coverage capability, widely used and free availability. We used it to execute it in both instrumented mode and non-instrumented mode. AFL environment setup with the Jetson nano took around 1 day.

16.2.1. Instrumented mode

1) Test Execution :
① Recompiled the source code with AFL generated CC and CXX compilers afl-clang-fast and afl-clang-fast++, respectively.
② Solved compile time errors and set the environment variables to start the execution.

```
/home/test/tools/fuzz/AFL/afl-fuzz -i testcase_dir -o findings_dir
./LgFaceRecDemoTCP_Jetson_NanoV2 @@
```

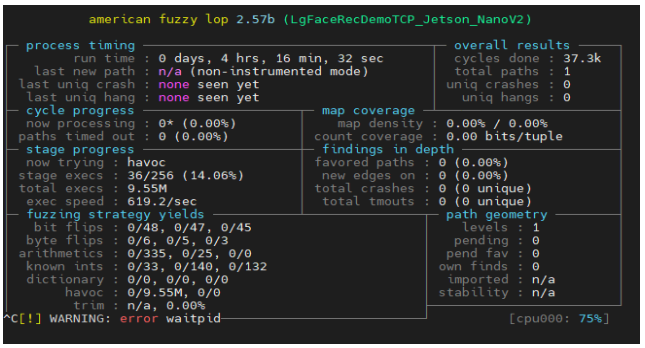
2) Test Result : Faced OOM fault errors, tried passing maximum available memory but not able to proceed further.

16.2.2. Non-instrumented mode

1) Test Execution :
① Used existing binaries, no need to recompile, use '-n' command line option with afl-fuzz.
② Able to start the execution and does not face OOM issues.

```
/home/test/tools/fuzz/AFL/afl-fuzz -n -i testcase_dir -o findings_dir
./LgFaceRecDemoTCP_Jetson_NanoV2 @@
```

2) Test Result : We kept it running for 5-6 hours, but complete execution needs much more time. It was difficult to keep it running for a long time on a Jetson nano device considering Jetson nano device safety due to overheating.



16.3. Recommendation

Error handling logic should be implemented in the CCTV monitoring system.

17. Common Criteria (CC)

17.1. Common Criteria Overview

The Common Criteria for Information Technology Security Evaluation (referred to as Common Criteria or CC) is an international standard (ISO/IEC 15408) for computer security certification. Common Criteria provides assurance that the process of specification, implementation and evaluation of a computer security product has been conducted in a rigorous and standard and repeatable manner at a level that is commensurate with the target environment for use.

The evaluation process of Common Criteria also tries to establish the level of confidence that may be placed in the product's security features through quality assurance processes. The following table shows the summary of assurance class and levels in Common Criteria.

Assurance class	Assurance Family	Assurance Components by Evaluation Assurance Level						
		EAL1	EAL2	EAL3	EAL4	EAL5	EAL6	EAL7
Development	ADV_ARC		1	1	1	1	1	1
	ADV_FSP	1	2	3	4	5	5	6
	ADV_IMP				1	1	2	2
	ADV_INT					2	3	3
	ADV_SPM						1	1
	ADV_TDS		1	2	3	4	5	6
Guidance documents	AGD_OPE	1	1	1	1	1	1	1
	AGD_PRE	1	1	1	1	1	1	1
Life-cycle support	ALC_CMC	1	2	3	4	4	5	5
	ALC_CMS	1	2	3	4	5	5	5
	ALC_DEL		1	1	1	1	1	1
	ALC_DVS			1	1	1	2	2
	ALC_FLR							
	ALC_LCD			1	1	1	1	2
	ALC_TAT				1	2	3	3
Security Target evaluation	ASE_CCL	1	1	1	1	1	1	1
	ASE_ECD	1	1	1	1	1	1	1
	ASE_INT	1	1	1	1	1	1	1
	ASE_OBJ	1	2	2	2	2	2	2
	ASE_REQ	1	2	2	2	2	2	2
	ASE_SPD		1	1	1	1	1	1
	ASE_TSS	1	1	1	1	1	1	1
Tests	ATE_COV		1	2	2	2	3	3
	ATE_DPT			1	2	3	3	4
	ATE_FUN		1	1	1	1	2	2
	ATE_IND	1	2	2	2	2	2	3
Vulnerability assessment	AVA_VAN	1	2	2	3	4	5	5

Common Criteria is the certification which requires a high level of assurance and long evaluation time compared to other certifications. To get Common Criteria certification, security assessment by the authorized lab must be required. Common criteria provides 7 assurance levels. Actually, in the security field, to meet EAL2 is not easy at all. To get CC certification, manufacturers should invest much time and effort to prepare all the documents and develop security features. Thus, we aim to review CCTV monitoring system with EAL 1 requirements. The following table shows the assurance class of EAL 1.

Assurance Class	Assurance components
ADV: Development	ADV_FSP.1 Basic functional specification
AGD: Guidance documents	AGD_OPE.1 Operational user guidance
	AGD_PRE.1 Preparative procedures
ALC: Life-cycle support	ALC_CMC.1 Labelling of the TOE
	ALC_CMS.1 TOE CM coverage
ASE: Security Target evaluation	ASE_CCL.1 Conformance claims
	ASE_ECD.1 Extended components definition
	ASE_INT.1 ST introduction
	ASE_OBJ.1 Security objectives for the operational environment
	ASE_REQ.1 Stated security requirements
	ASE_TSS.1 TOE summary specification
ATE: Tests	ATE_IND.1 Independent testing - conformance
AVA: Vulnerability assessment	AVA_VAN.1 Vulnerability survey

17.2. Summary of Evaluation Result

Assurance Class	Assurance Component	Evaluator Action Elements	Verdict		
			Evaluator Action Elements	Assurance Component	Assurance Class
ADV : Development	ADV_FSP.1 : Basic functional specification	ADV_FSP.1.1E	Pass	Pass	Pass
		ADV_FSP.1.2E	Pass		
AGD : Guidance documents	AGD_OPE.1 : Operation user guidance	AGD_OPE.1.1E	Pass	Pass	Pass
	AGD_PRE.1 : Preparative procedures	AGD_PRE.1.1E	Pass	Pass	
		AGD_PRE.1.2E	Pass		
ALC : Life-cycle support	ALC_CMC.1 : Labelling of the TOE	ALC_CMC.1.1E	Pass	Pass	Pass
	ALC_CMS.1 : TOE CM coverage	ALC_CMS.1.1E	Pass	Pass	
ASE: Security Target evaluation	ASE_CCL.1 : Conformance claims	ASE_CCL.1.1E	NA	NA	Pass
	ASE_ECD.1 : Extended components definition	ASE_ECD.1.1E	NA	NA	
		ASE_ECD.1.2E	NA		
	ASE_INT.1 : ST introduction	ASE_INT.1.1E	Pass	Pass	
		ASE_INT.1.2E	Pass		
	ASE_OBJ.1 : Security objectives for the operational environment	ASE_OBJ.1.1E	Pass	Pass	
	ASE_REQ.1 : Stated security requirements	ASE_REQ.1.1E	Pass	Pass	
	ASE_TSS.1 : TOE summary specification	ASE_TSS.1.1E	Pass	Pass	
		ASE_TSS.1.2E	Pass		

ATE : Tests	ATE_IND.1 : Independent testing - conformance	ATE_IND.1.1E	Pass	Pass	Pass
		ATE_IND.1.2E	Pass		
AVA : Vulnerability assessment	AVA_VAN.1 : Vulnerability survey	AVA_VAN.1.1E	Pass	Pass	Pass
		AVA_VAN.1.2E	Pass		
		AVA_VAN.1.3E	Pass		

17.3. ADV : Development

* ADV_FSP.1 : Basic functional specification

No	Item	Contents	Assessment Result	Justification
1	ADV_FSP.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.	Pass	1. The software requirement.pdf document page 3 explains the Security Goals for Confidentality, Integrity, and Availability. 2. It excludes the part about traceability.
2	ADV_FSP.1.2E	The evaluator shall determine that the functional specification is an accurate and complete instantiation of the SFRs.	Pass	The software requirement.pdf document page 8 explains the SFR.

17.4. AGD : Guidance Documents

* AGD_OPE.1 Operation user guidance
* AGD_PRE.1 Preparative procedures

No	Item	Contents	Assessment Result	Justification
1	AGD_OPE.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.	Pass	The Software Requirement.pdf file explains the operation part.
2	AGD_PRE.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.	Pass	It was provided as a README.md file, but there are deficiencies. I hope to write more in detail in the future.
3	AGD_PRE.1.2E	The evaluator shall apply the preparative procedures to confirm that the TOE can be prepared securely for operation.	Pass	It was provided as a README.md file, but there are deficiencies. I hope to write more in detail in the future.

17.5. ALC : Life-cycle Support

* ALC_CMC.1 : Labelling of the TOE
* ALC_CMS.1 : TOE CM coverage

No	Item	Contents	Assessment Result	Justification
1	ALC_CMC.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.	Pass	Identification of configuration items for the TOE is done through git.
2	ALC_CMS.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.	Pass	The TOE related data is managed by git.

17.6. ASE : Security Target Evaluation

* ASE_CCL.1 : Conformance claims
* ASE_ECD.1 : Extended components definition
* ASE_INT.1 : ST introduction
* ASE_OBJ.1 : Security objectives for the operational environment
* ASE_REQ.1 : Stated security requirements
* ASE_TSS.1 : TOE summary specification

No	Item	Contents	Assessment Result	Justification
1	ASE_CCL.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.	NA	This part is outside the scope of the project and is not evaluated.
2	ASE_ECD.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.	NA	This part is outside the scope of the project and is not evaluated.
3	ASE_ECD.1.2E	The evaluator shall confirm that no extended component can be clearly expressed using existing components.	NA	This part is outside the scope of the project and is not evaluated.
4	ASE_INT.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.	Pass	It is evaluated by replacing ST with Software Design.pdf and Software Requirements.pdf files.
5	ASE_INT.1.2E	The evaluator shall confirm that the TOE reference, the TOE overview, and the TOE description are consistent with each other.	Pass	It is evaluated by replacing ST with Software Design.pdf and Software Requirements.pdf files.
6	ASE_OBJ.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.	Pass	The security objectives for the operating environment are described in the Software Requirement.pdf file.

7	ASE_REQ.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.	Pass	The security requirements are described in the Software Requirement.pdf file.
8	ASE_TSS.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.	Pass	The TOE description is described in the Software Design.pdf file.
9	ASE_TSS.1.2E	The evaluator shall confirm that the TOE summary specification is consistent with the TOE overview and the TOE description.	Pass	The TOE description is described in the Software Design.pdf file.

17.7. ATE : Tests

* ATE_IND.1 Independent testing - conformance

No	Item	Contents	Assessment Result	Justification
1	ATE_IND.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.	Pass	The system provides a test case.pdf file for testing.
2	ATE_IND.1.2E	The evaluator shall test a subset of the TSF to confirm that the TSF operates as specified.	Pass	The system provides a test case.pdf file for testing.

17.8. AVA : Tests Vulnerability Assessment

* AVA_VAN.1 : Vulnerability survey

No	Item	Contents	Assessment Result	Justification
1	AVA_VAN.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.	Pass	Vulnerability was reviewed, but there were no critical items.
2	AVA_VAN.1.2E	The evaluator shall perform a search of public domain sources to identify potential vulnerabilities in the TOE.	Pass	Vulnerability was reviewed, but there were no critical items.
3	AVA_VAN.1.3E	The evaluator shall conduct penetration testing, based on the identified potential vulnerabilities, to determine that the TOE is resistant to attacks performed by an attacker possessing Basic attack potential.	Pass	Vulnerability was reviewed, but there were no critical items.

18. Summary of Assessment Result

We tried to penetrate the system with various attacks. The following is the summary of the vulnerabilities we found during our investigation.

	Severity = High	Severity = Medium	Severit y= Low
Common Attack	TLS version control Deployment process attack	Encryption Key	
Input Data Attack	Engine data file modification		Weak input validation Fuzz image data
Outer Attack	Ssh account login	Face recognition error	Rpcbind dos attack
Inner Attack	Mutual authentication attack RootCA certificate change attack MITM attack	Same person name attack	Protocol robustness testing

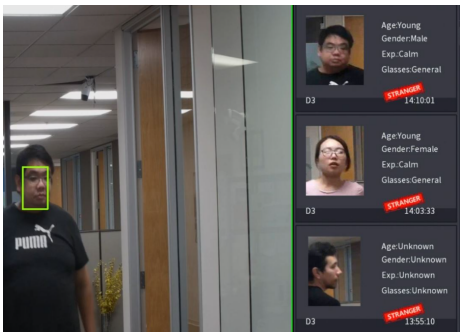
19. Strength / Weakness

19.1. Strength

- Minimal interface design for security
- Simplified use through mutual authentication
- Provides user register function for minimal privileges
- Encryption applied to user image files
- Apply OS key control (key ring)
- Apply log for non-repudiation
- Unidirectional network data flow

19.2. Weakness

- Since some of the basic requirements are not implemented, the areas that can be evaluated are limited.
- **Remove developer code** : Because the fixme code is not deleted, certificates for other systems can be used in the system.
- **Remove C coding style** : There are parts that adhere to the C coding style, and the compiler will handle them well though. (true/1, nullptr/NULL)
- **Missing certificate control** : When using a certificate, the process of revoke and redistribution is required, but it is missing. Also, the process of authenticating the rootca certificate is missing. In addition, it is necessary to strengthen the part that authenticates the counterpart's certificate.
- **TLS version control error** : The design document says that 1.2 is supported. But the actual operation is set to 1.3, and TLS1.0, TLS1.1, TLS1.2, and TLS1.3 are all supported by program settings.
- Hard-coded IV of AES-CBC, and secure key expansion missing
- The distribution file should be released with the release mode. It is required to implement a verification function to check if the executable file is the legitimate file.
- The Monitoring system has no UI, so the security agent must check the strangers only in real time. It would be convenient if the GUI for stranger history management is provided as captured below.



20. Lesson & Learned

1. As we evaluate the work of other teams, it seems to be an opportunity to reconsider what we did.

2. In order to review security vulnerabilities, I had the experience of analyzing the project from a new perspective.

3. Another team's vulnerability is our team's vulnerability. We've found that it takes a lot of effort to create a secure program.

4. I'm impressed that the sudo command has had the vulnerability for about 10 years. I thought the other team's work was well written and perfect, but as we continued to review it with patience, we saw something to improve. I have come to realize that security assessments need to be constantly reviewed from all sides.

5. Analyzing vulnerabilities from an attacker’s point of view was a new experience, and I have found the importance of analyzing and removing threats from multiple perspectives.

6. It’s interesting to review other team’s security features. CMU/LGE githubs can be the security coding reference book for me! Thank you!

7. We must have in depth knowledge of at least one tool in each phase of evaluation, including its uses, internals, pros and cons.