Final Security Project Report

July/2/2021 Team 4

Contents

Introduction to Team 4

Aka Potential


Clifford(mentor)


Daesik


Kyungsik


Hanil


Yujin

Dahee


Jinchul

Phase 1

Lead & schedule management: Daesik Kim
Architect: Kyungsik Lee
Configuration management & build: Hanil Jang, Yujin Lee (use github)
Acceptance Test & QA: Dahee Jung, Jinchul Kim
Risk Assessment & Mitigations: All
Development & Documentation:

- Application(User Interface) side: Hanil Jang, Yujin Lee, Dahee Jung
- Jetson Nano side: Daesik Kim, Kyungsik Lee, Jinchul Kim

Phase 2

- Security Assessment Process Management: Hanil
- Penetration Testing: Daesik
- Dynamic Analysis: Dahee
- Static Analysis: Jinchul
- Input validation/Testing setup: Yujin and Kyungsik

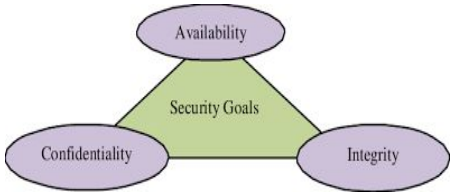
Phase 1

Security Development for CCTV system

Project Goals & Application

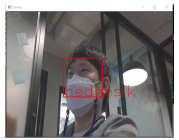
Security Development for Tartan system

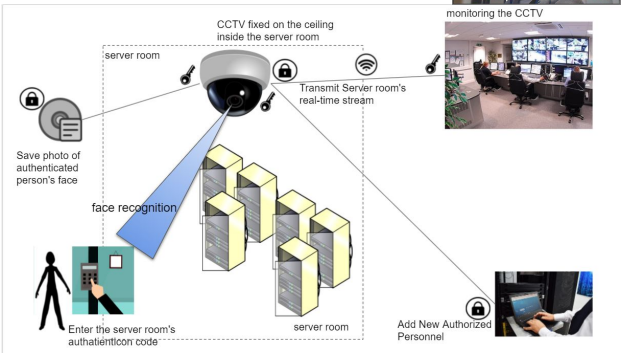
- Business perspective
  - Think of this system as MVP(minimum viable product)
- Security Goals
  - Focused on enhancing security of the product.
  - Should be designed to achieve three principles(CIA).



Application

- Functional requirements
  - Monitoring and tracking people with CCTV in the server room
  - Detecting unauthorized people with facial recognition





- Security requirements
  - Network section between CCTV and client are encrypted.
  - Relevant data to privacy info should be encrypted securely.
  - Keys which are used for encryption should be kept safely.

Design - Threat Modeling & Risk Assessment

Threat Modeling

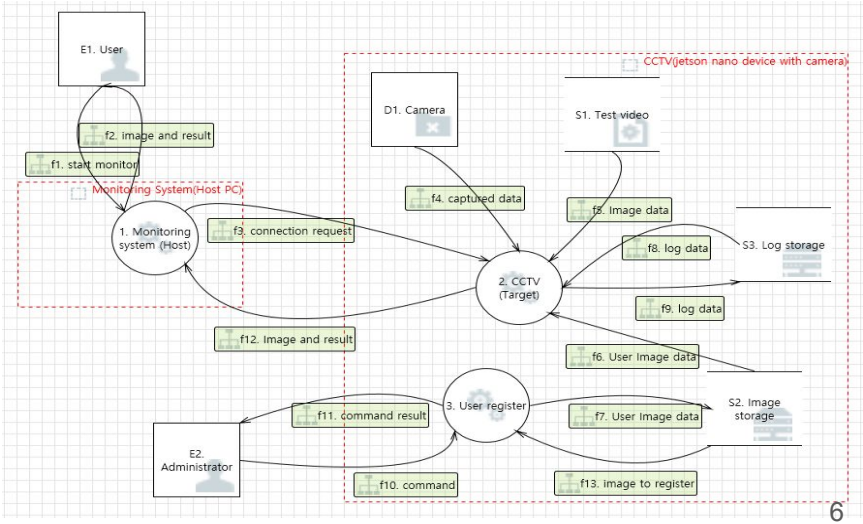
- Data Flow Diagrams
  Decompose the system into parts and show that each part is not susceptible to relevant threats.
- Employ threat modeling using followings
  - STRIDE
  - PnG

Risk Assessment

- Identify threats we must mitigate.
- Derive high risks from overall risk severity.

Threats	Mitigated	High	Medium	Low
90	25	11	45	9

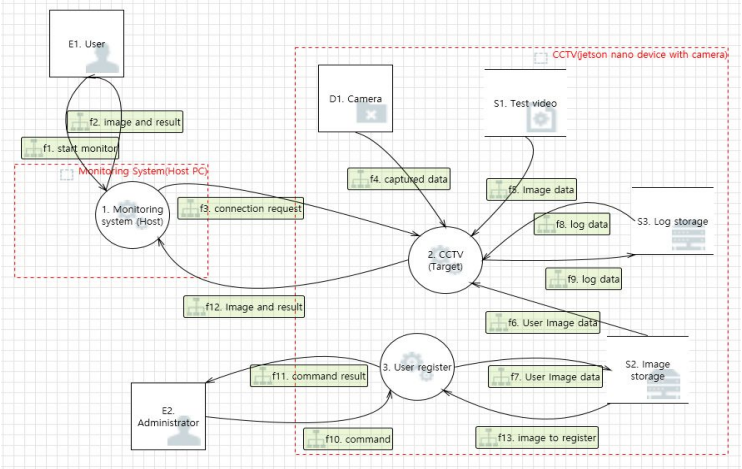
Overall Risk Severity				
Impact	HIGH	Medium	High	Critical
	MEDIUM	Low	Medium	High
	LOW	Note	Low	Medium
		LOW	MEDIUM	HIGH
	Likelihood			

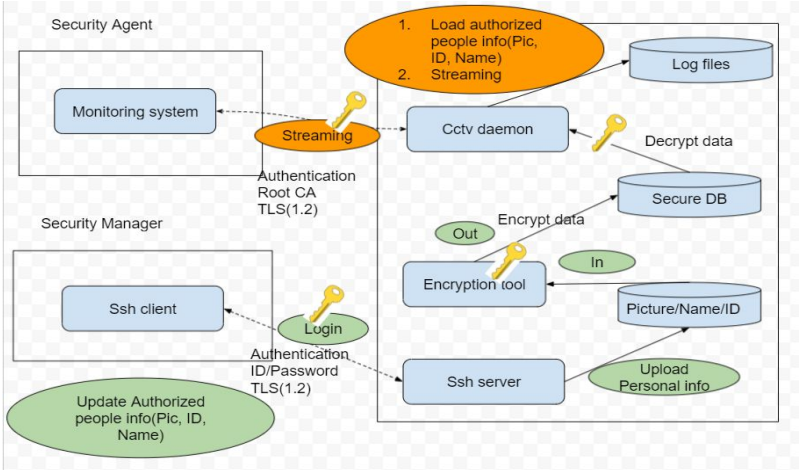


Design - Mitigations

Risk mitigations

- We have identified high risks through risk assessment.
- There are four sections which require security development to mitigate those risks.
- Network/User Information/Key management/Logging





Implementation

Secure Coding

- Static Analysis
  Analyzing the source code prior to the compilation provides a highly scalable method of security code review and helps ensure that secure coding policies are being followed.

Result of static analysis tools

Tools	Target	Total Detected	false positive	To mitigate	Remark
Sonarcloud	Monitoring system components	247	247	0	- 218 issues are detected as a code smell type, which is false positive and the rest are minor issues.
Code x-ray	all components	24	24	0	- The issues detected by Blocker(1 issue) and Major(4 issues) are about the files(out of scope) or have no effect on the code.
Flawfinder	CCTV and user register components	48	43	5	- 5 issues are fixed.(2 issues related to integer overflow, 3 related to statically-sized buffer)

*) We decided to fix the issues found in the static analysis if necessary for items greater than Major (FlawFinder Level 3).

Reflection on Phase 1

- We couldn't make a situation where two factor authentication is required whiling defining the system requirements.
- PnG was helpful in additionally identifying threats from the attacker's point of view.
- It is unfortunate that the threat analysis for the entire certificate management lifecycle was insufficient.
- We did not properly investigate the vulnerabilities of subcomponent and base system (linux) configuration. (efficient task management)

Phase 2

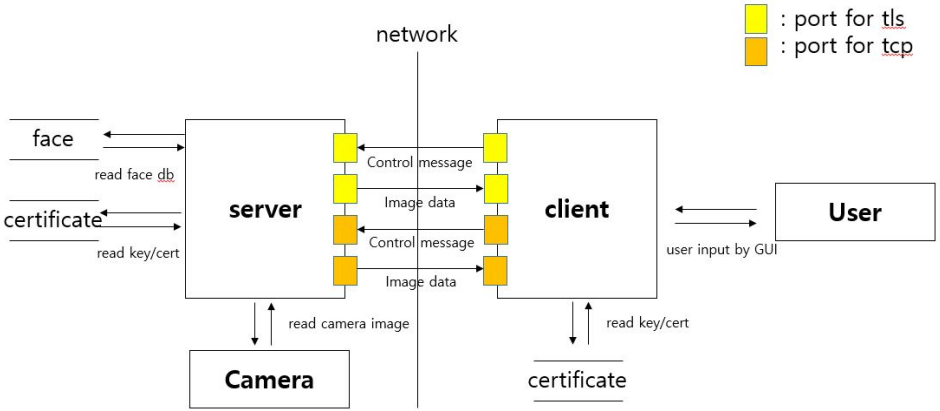
Security Assessment for Target System

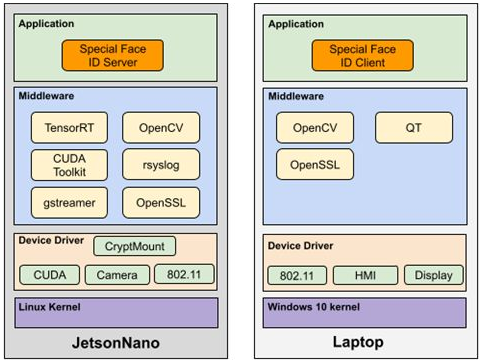
Target System(Team 5) Overview

Functional requirement
- Monitoring video conference attendees with facial recognition
- GUI based user input
  - User registration

Security requirement
- Encrypt user information/certification
- Mutual authentication

Known Issues
- The system is designed not to be disconnected.
  (It it required to reboot whenever server and client are disconnected.)
- No automatic recovery. The system is required to reboot manually.





Security Assessment Approach

Planning

- Assign tasks from backlog and do daily meeting
- Check work items

Security Assessment Process

- Target system analysis -> Strategy -> Assessment -> Report

Strategy

- Find common vulnerabilities based on OWASP report.
- Identify the vulnerability to the subsystem used by the target system.
- Investigate mitigation for security assets and find weaknesses of mitigations.
- Find mistakes or flaws in Design or Implementation.
- Leverage vulnerabilities we encountered for the security development in phase 1 to assess the target system.

Overall Security Assessment Report

Summary of Vulnerabilities

	Vulnerabilities	Severity	Explotiable
1	Operation Mode tampering	High	O
2	Buffer overrun due to “`nameToRegister`” global variable(File length)	High	O
3	Dos attack to opened port of rpcbind	High	O
4	Uncontrolled resource consumption(no socket close)	High	O
5	Client certificate is exposed	High	
6	Openssl vulnerability	Medium	
7	Insufficient Logging & Monitoring	Low	

security assessment report.pdf

Vulnerability - OWASP Top 10 Vulnerabilities Check

Top 10	ID	Relevant	Check Mitigation	Notes
1	A1:2017-Injection			
2	A2:2017-Broken Authentication	Related	**Mitigated**	
3	A3:2017-Sensitive Data Exposure			
4	A4:2017-XML External Entities (XXE)			
5	A5:2017-Broken Access Control	Related	**Mitigated**	
6	A6:2017-Security Misconfiguration	Related	**Mitigated**	
7	A7:2017-Cross-Site Scripting XSS			
8	A8:2017-Insecure Deserialization			
9	A9:2017-Using Components with Known Vulnerabilities	Related	**Need mitigation**	Openssl library which is required for the server program should be updated to the latest version(1.1.1k)
10	A10:2017-Insufficient Logging & Monitoring	Related	**Need mitigation**	Log information for non-repudiation is not sufficient. Only connection info is available.

Vulnerability - Artifact Review

Client certificate exposed

- Server certificates are secured by cryptmount.
- Client certificate is one of security assets, but it is not sufficiently protected.

Security Requirements.xlsx

SR 4-3	Client certificates must be stored in secure storage	6,7,8,9	Spoofing, DoS, Information Disclosure attacks on DS3. Certificate Data Storage	Assume APIs for secure stroage is used

- Mitigation(recommendation)
  - Windows supports “Certificate Store” and provides Certificate Manager as a management tool.
  - Certificates are protected with user credentials.

Operation mode tampering

- Demo will be provided in the exploit slide.

User requirements.docx

Req39	4. How fault/error detection, recovery, and reporting is designed and implemented.

No Automatic Recovery system(server side)

- Client or Server disconnect or crash -> Login as LG -> Kill server -> Run server -> Enter certificate password, Is it supposed to be functional?

Vulnerability - Input Validation 1/3

Fuzz Testing

- Fuzzer analysis

Fuzzing tools	QT GUI app fuzzing support	Network payload transport support	Fuzzing type	Previous experience
AFL	Not supported	Not supported	mutation based dumb fuzzer	Experienced
Peach Fuzzer	Not supported	Not supported	smart or dumb fuzzer	Not experienced
WinAFL	Not supported	Not supported	mutation based dumb fuzzer	Not experienced
sfuzz	Not supported	Supported	dumb	Not experienced

```
placeholderText: qsTr("Person Name")
selectByMouse: true
enabled: false
validator: RegExpValidator {
    regExp: /(^[a-zA-Z₩d₩.]$)|(^[a-zA-Z₩d₩.][a-zA-Z₩d₩. ]{0,249}[a-zA-Z₩d₩.]$)/
}
```

*) Input validation using regex in the client

- Test Result

Fuzzing target(input)	Number of vulnerabilities found	Types of vulnerabilities	Fuzzing tools used	Test execution time
IP value in client app	0	Not applicable	AFL	4 hours
User name in client app	0*)	Not applicable	AFL	8 hours
Server ports	1**)	Uncontrolled resource consumption (CWE - 400)	sfuzz	8 hours

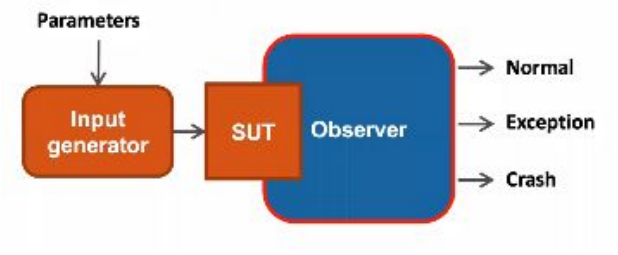
```
Listening for TLS connection:  Control Port
Listening for TLS connection: Image Port
547021601056:error:1408F09C:SSL routines:ssl3_get_re
record.c:322:



547013208352:error:1408F09C:SSL routines:ssl3_get_re
record.c:322:
ERROR on accept: Too many open files
AcceptTcpConnection Failed
```

**) Crash log output from server terminal

Vulnerability - Input Validation 2/3

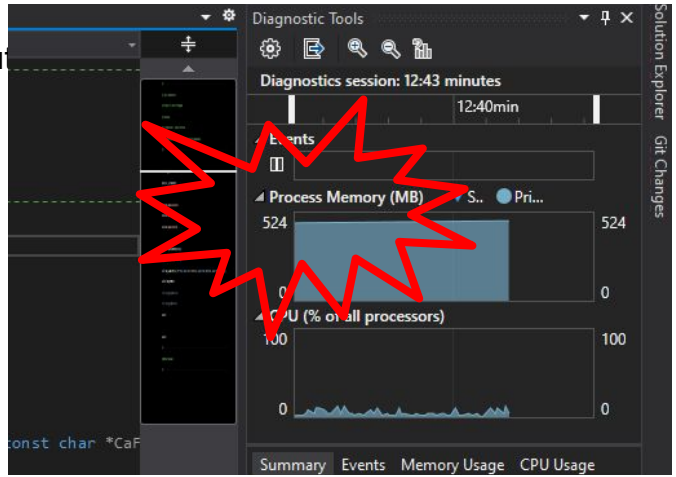
Fuzzing Challenges

- Apply automatic recovery to continue for the next test
- Client memory leak
  - *)Memory usage(after network connection): 1 MiB/sec.
- Server memory leak
  - Objects regarding port are allocated after each connection but not freed.
- **)Too many open files: No close() on sockets
- Server system goes to abnormal state when a exception occurs but not terminated.
  - Oracle Problem: Add exit() in the path to abnormal state to terminate process.




```
open: cnt: 981 port 0x7f3401a420 cfd 3f6 0
open: cnt: 982 port 0x7f3401a440 cfd 3f7 0
open: cnt: 983 port 0x7f3401a460 cfd 3f8 0
open: cnt: 984 port 0x7f3401a480 cfd 3f9 0
open: cnt: 985 port 0x7f3401a4a0 cfd 3fa 0
open: cnt: 986 port 0x7f3401a4c0 cfd 3fb 0
open: cnt: 987 port 0x7f3401a4e0 cfd 3fc 0
open: cnt: 988 port 0x7f3401a500 cfd 3fd 0
open: cnt: 989 port 0x7f3401a520 cfd 3fe 0
open: cnt: 990 port 0x7f3401a540 cfd 3ff 0
ERROR on accept: Too many open files
AcceptTcpConnection Failed
```

**)Error screen from server terminal

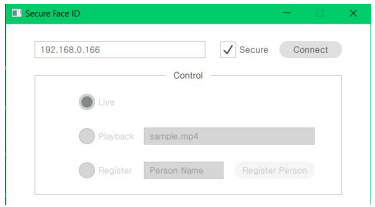


*)Confirm the memory increase in visual studio

Vulnerability - Input Validation 3/3

Input Test Scenario

1. User Registration
2. Try to create a user file with user name
3. Fail to create due to too long file name



main.cpp

```
} else if (recvData.msgType == E_MSG_ADD_USER ) {
    if(recvData.dataLen > 0 || recvData.dataLen < MAX_NAME_LEN) {
        memset(nameToRegister, 0, sizeof(nameToRegister));
```

It should be '&&' not a '||'

	Max Input Length	Test Input Length	Validation
GUI(Client)	256	256	PASS
User Registration	**BUG**	256	PASS
File name	255	262(= 256 + 2 + 4) "Name" + "_?" + ".jpg"	**FAIL** (255 < 262)

Vulnerability - Static Analysis (1/2)

Approach strategy
Review all issues and find vulnerabilities that can be actual targets of attack.

Review issues with compare to base code.
Any issues that are likely to be attacked are checked to be came from the code written by the dev team, not the base code.

		False positives	Ignore	Need Investigation	Total
sfid-server-master	base code	3	22	-	46
	modified	-	9	12	
sfid-client-main	base code	-	-	-	12
	modified	1	3	8	

Issue list - static_analysis-flawfinder-5team

Vulnerability - Static Analysis (2/2)

In-depth code review

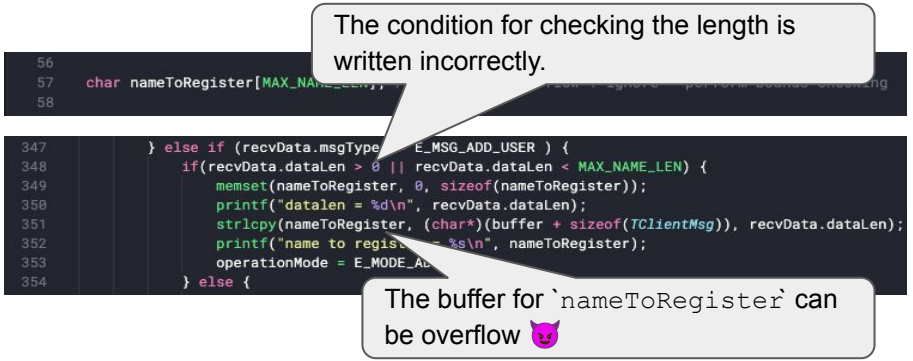
For 12 issues classified as 'needs investigation'

Issue #19

“Statically-sized arrays can be improperly restricted, leading to potential overflows or other issues (CWE-119!/CWE-120) ...”

There is a bug in the condition code for boundary check.

-> likely to attack!



Issue #20

“Statically-sized arrays can be improperly restricted, leading to potential overflows or other issues (CWE-119!/CWE-120) ...”

Using fixed-sized buffer for receiving data from socket, but overflow could not be made due to the code receiving the data by limiting the length. **-> unable to attack.**

For more information - Static analysis and exploit trials

Vulnerability - Test Case Analysis

Verify Test Cases provided

Overview of whether the test case covers the all requirements

- Total number of requirements = 18
- Number of requirements covered = 7.25[*)]

Requirements Coverage = 40.28%

Overview of testing progress by counting the number of passed and failed tests

- Total number of tests to be run = 15
- Number of tests passed = 12
- Number of tests failed = 3

Percentage of tests passed = 80.00%

Review on verification

Test cases do not cover all the requirements

- No TC for Functional Requirements.
- Missing path when considering the path coverage for each requirement.

No test results were found in the provided artifacts and 3 test cases failed.

- TC-02 : No error message pops up
- TC-05 : Image not saved when extremely long file name is entered
- TC-14 : Image file can see without encryption

Vulnerabilities

- Found on TC-05 : Vulnerability - Input Validation 3/3

[*)] Number of requirements covered = Actual number of written test cases / Expected number of test cases calculated by path coverage
https://docs.google.com/spreadsheets/d/1cnqD7IU0gPA3hAFEIK-aBDbe_Lx_kivZEckjGIAuPLY/edit?usp=sharing

Vulnerability - Penetration Testing

Analysis & Test domain and tools

- OS : metasploit
- Network : wireshark, nmap
- Server/Client program : metasploit
- Library used in Server/Client : find used library version and search CVEs not fixed

Vulnerability analysis

- Network analysis : data is encrypted and mutually authenticate each other
- OS analysis : Only 8 port are opened Including four ports used for camera programs
  Dos attack is possible with rpcbind port.
- Library analysis : openssl used in server version is 1.1.1 but latest version is 1.1.1k.
  Some vulnerabilities are exist which are registered to CVE.

Vulnerabilities

- openssl : some vulnerabilities are exist in openssl version between 1.1.1 and 1.1.1k (openssl CVE ex:CVE-2021-23840(overflow))
- rpcbind : dos attack is possible and that vulnerability is not fixed. (CVE-2017-8779)

Recommandation

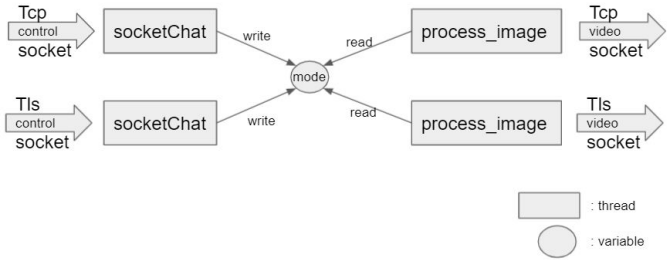
- Use latest library used in program. (openssl 1.1.1 -> 1.1.1k)
- Disable rpcbind service and use network firewall to prevent dos attack.

```
PORT     STATE SERVICE       VERSION
22/tcp   open  ssh           OpenSSH 7.6p1
53/tcp   open  domain        dnsmasq 2.79
111/tcp  open  rpcbind       2-4 (RPC #100
3389/tcp open  ms-wbt-server xrdp
5000/tcp open  upnp?
5001/tcp open  tcpwrapped
6000/tcp open  X11?
6001/tcp open  X11:1?
Service Info: OS: Linux; CPE: cpe:/o:linux:
```

Port scanning result by nmap

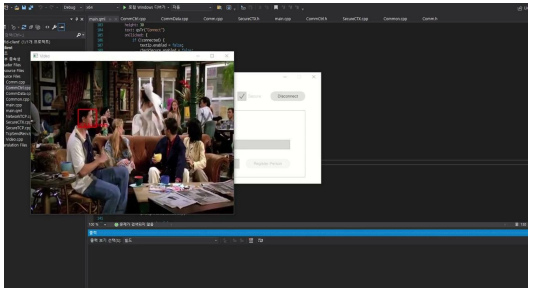
Exploits - Operation Mode Tampering

How to exploit vulnerabilities we found (Demo)



Tampering on camera live stream(Misuse Case)

- Precondition
  User is watching the camera live stream with secure mode.
- Flow
  1. The attacker’s client connects to the server with insecure mode.
  2. The attacker sends a command to switch to playback mode.
- Postcondition
  The user watches video stream, “Friends” instead of camera.



Exploits - Buffer overrun by sending manipulated data

Manipulate data

Generate the msg beginning with '\x00\x03\xff\xff' which contains E_MSG_ADD_USER as msgType and \xffff (which could be 2562-1=65535) as dataLen

Connect and send data

Using python script, connected to TLS control socket (port 6000) with proper private key and certificate.

Data overrun has been check with GDB.

Observe using GDB

`nameToRegister` gets overflowed and the data is written to the following global variables which are `_ConnCli[]` and `_ProcImg[]`.

Lessons Learned

- By comparing and investigating the mitigated contents, it was very easy to find related vulnerabilities. I thought that vulnerabilities could be easily exposed if similar contents were not mitigated to the best practice level.
- We thought that libssl-dev v1.1.1 installed by default in Ubuntu 18.04 would be the latest version, so we did not think to check the vulnerability. I had to use the latest version, v1.1.1k, with all vulnerabilities fixed.
- Do not ignore static analysis issues, it may come with critical vulnerability.
- The same vulnerability also seems to vary in severity depending on the use scenario. While it is important to determine and apply threats and security measures, it is also important to define clear product functional requirements, usage scenarios, and operating environments.
- As it is very difficult to visualize the completeness of the system while analyzing the test cases, I felt that establishing and agreeing on metrics and criteria is very important.

Q & A session

Deliverables

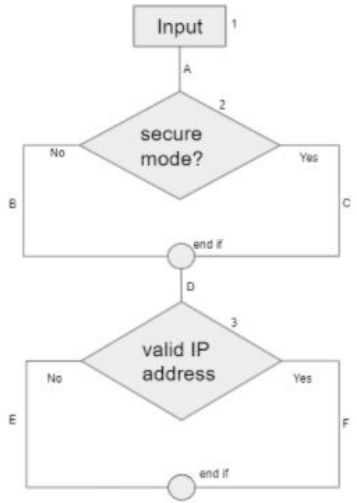
https://github.com/hijang/lsc_cctv

Back Up Slide

https://docs.google.com/spreadsheets/d/1cnqD7IU0gPA3hAFEIK-aBDbe_Lx_kivZEckjGIAuPLY/edit?usp=sharing

Back Up Slide

*)	Test coverage strategy
	- Path Coverage
	Path testing is a structural testing method in order to find every possible executable path.
	- Reason for choosing
	Path testing is a structural testing method in order to find every possible executable path.
	Ex) For SR 1-1, there are 4 possible routes as follows.
	1. 1A-2B-D-3E
	2. 1A-2B-D-3F
	3. 1A-2C-D-3E
	4. 1A-2C-D-3F



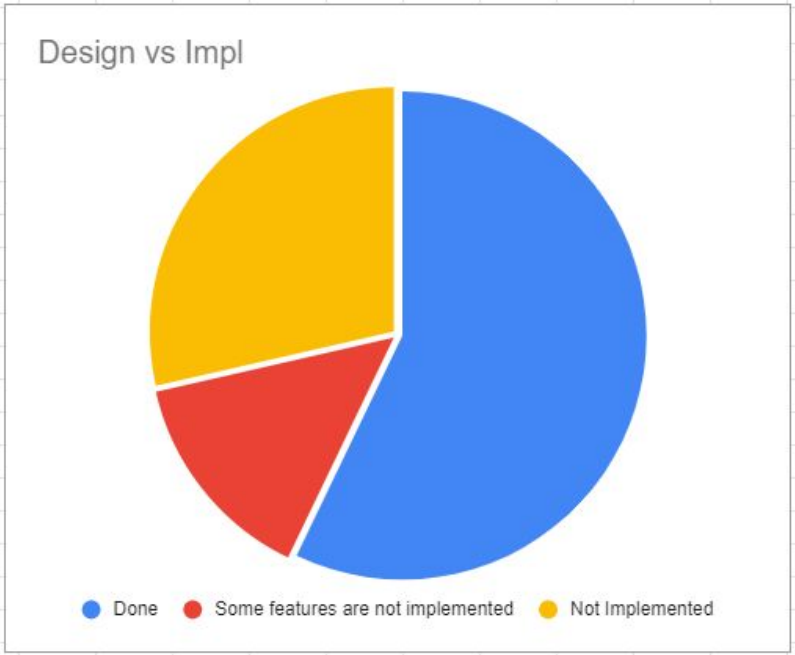
https://docs.google.com/spreadsheets/d/1cnqD7IU0gPA3hAFEIK-aBDbe_Lx_kivZEckjGIAuPLY/edit?usp=sharing

Back Up Slide

Table showing the correlation between requirements(design) and implementation

Implementation status for a total of 21 requirements

Done	12
Some features are not implemented	3
Not Implemented	6



https://docs.google.com/spreadsheets/d/1cnqD7IU0gPA3hAFEIK-aBDbe_Lx_kivZEckjGIAuPLY/edit?usp=sharing