black hat EUROPE 2019

DECEMBER 2-5, 2019

EXCEL LONDON, UK

Fuzzing and Exploiting Virtual Channels in Microsoft Remote Desktop Protocol for Fun and Profit

Chun Sung Park · Yeongjin Jang · Seungjoo Kim* · Ki Taek Lee

KOREA UNIVERSITY · Oregon State University · KOREA UNIVERSITY · KOREA UNIVERSITY · SAMSUNG

* Corresponding Author

ABOUT US

- Chun Sung Park
  - Graduate student at SANE LAB, Korea University
  - Also a CTO of Diffense, Inc.

- Yeongjin Jang
  - Assistant Professor of Computer Science, Oregon State University

※ This work is partially supported by Institute for Information & communications Technology Promotion(IITP) grant funded by the Korea government(MSIT) (No.2018-0-00532,Development of High-Assurance(≥EAL6) Secure Microkernel)

ABOUT US

- Seungjoo (Gabriel) Kim **(Corresponding Author, skim71@korea.ac.kr)**
  - Professor of Graduate School of Information Security, Korea University

- Ki Taek Lee
  - Ph.D. candidate at SANE Lab, Korea University
  - Offensive security researcher and penetration tester at Samsung Research

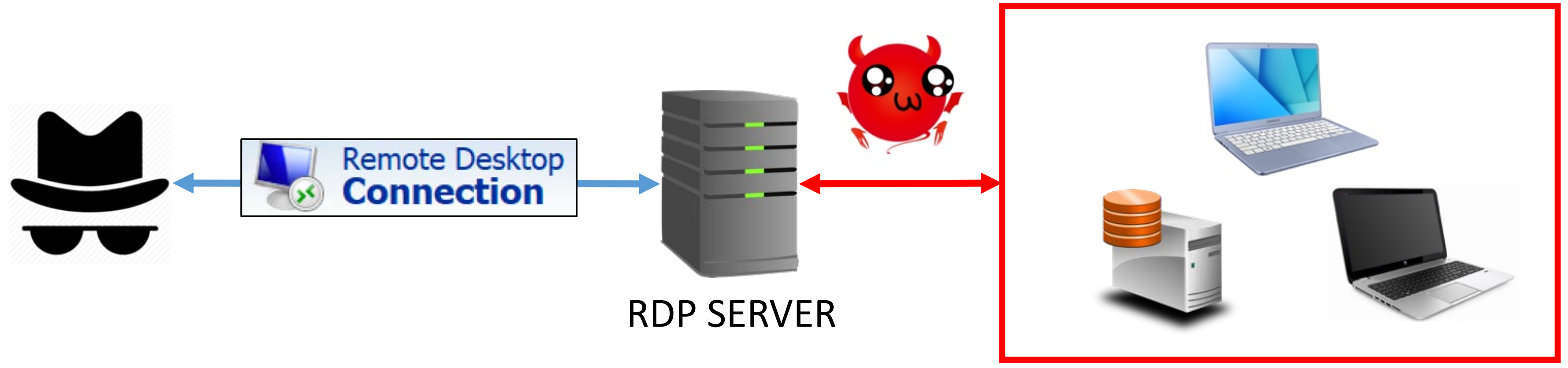
※ This work is partially supported by Institute for Information & communications Technology Promotion(IITP) grant funded by the Korea government(MSIT) (No.2018-0-00532,Development of High-Assurance(≥EAL6) Secure Microkernel)

AGENDA

- Motivation
- Background: Recon on the official MS RDP client and its protocol
- Finding Vulnerability Automatically: Build an RDP client fuzzer
- Applying the RDP Client Fuzzer
- Vulnerabilities
- RDP Heap Feng Shui
- Demo
- Future work & Conclusion

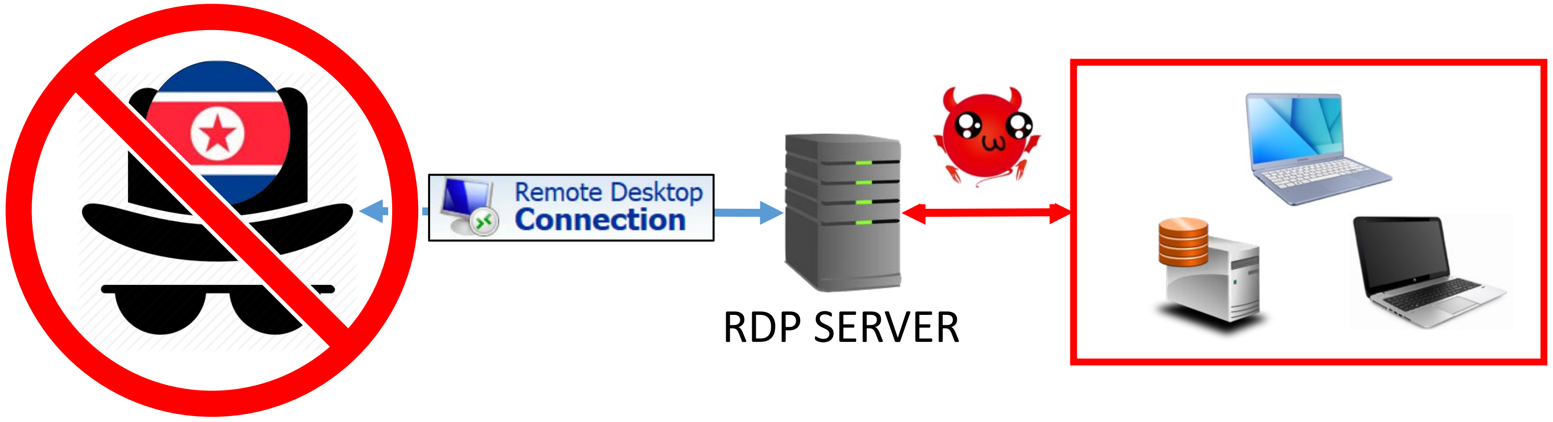
MOTIVATION

- Hackers are using some RDP servers to shadow their IP address for attacks
- Hacked windows servers in public are configured as an RDP server
  - Attackers launch attacks from this computer by logging into the RDP server
  - Victims can only see the IP address of the RDP server



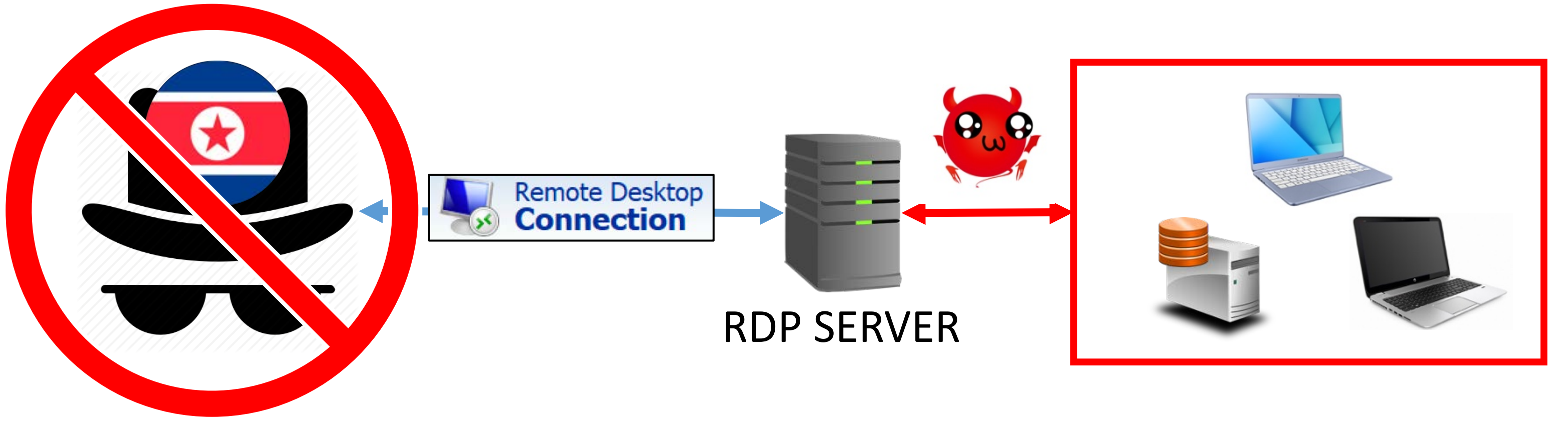
MOTIVATION

- Allegedly being used by many North Korean Hackers
  - No proof, but from off-the-record discussions with security analysts
- Came up with a fun motivation:
  - Can we pwn RDP clients to launch attacks back to attackers?



MOTIVATION

- Disclaimer
  - We did not use this techniques in public (yes, it's illegal)
  - We would like to share our journey for such a cool motivation in this presentation

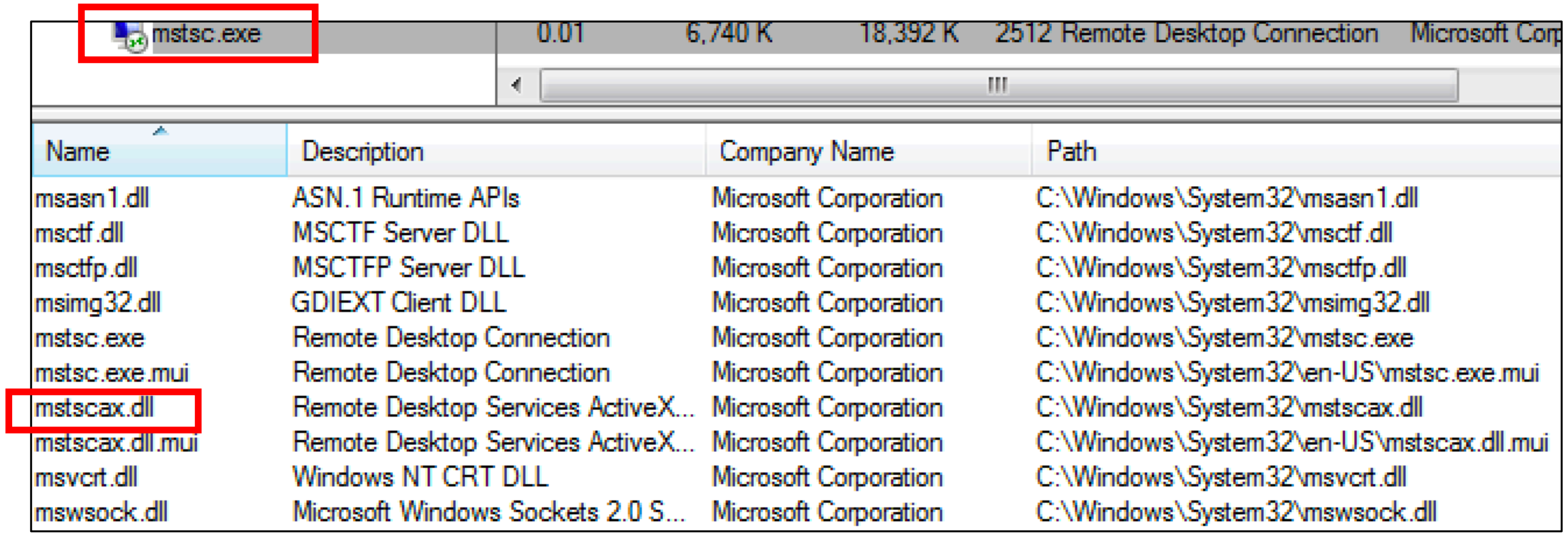


Another Note

- Disclaimer
  - **Some part of this research project cannot be disclosed by the restrictions applied to one of the speakers (NDA-related)**
  - **But, we will disclose as much as we can to entertain the audiences!**

Recon on MS RDP client

- Need an understand the RDP protocol
- Requires two files to take a look at (via IDA Pro)
  - Mstsc.exe
  - Mstscax.dll

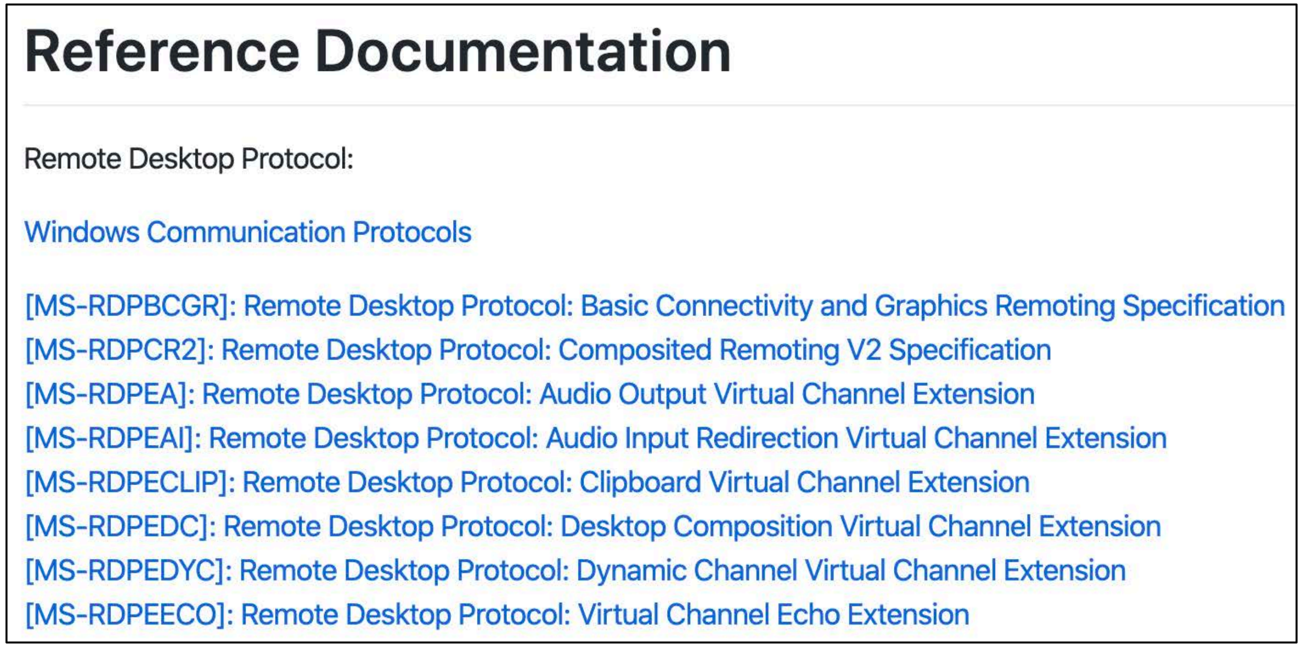


```
mstsc.exe    0.01    6,740 K    18,392 K    2512 Remote Desktop Connection    Microsoft Corp
```

Name	Description	Company Name	Path
msasn1.dll	ASN.1 Runtime APIs	Microsoft Corporation	C:\Windows\System32\msasn1.dll
msctf.dll	MSCTF Server DLL	Microsoft Corporation	C:\Windows\System32\msctf.dll
msctfp.dll	MSCTFP Server DLL	Microsoft Corporation	C:\Windows\System32\msctfp.dll
msimg32.dll	GDIEXT Client DLL	Microsoft Corporation	C:\Windows\System32\msimg32.dll
mstsc.exe	Remote Desktop Connection	Microsoft Corporation	C:\Windows\System32\mstsc.exe
mstsc.exe.mui	Remote Desktop Connection	Microsoft Corporation	C:\Windows\System32\en-US\mstsc.exe.mui
mstscax.dll	Remote Desktop Services ActiveX...	Microsoft Corporation	C:\Windows\System32\mstscax.dll
mstscax.dll.mui	Remote Desktop Services ActiveX...	Microsoft Corporation	C:\Windows\System32\en-US\mstscax.dll.mui
msvcrt.dll	Windows NT CRT DLL	Microsoft Corporation	C:\Windows\System32\msvcrt.dll
mswsock.dll	Microsoft Windows Sockets 2.0 S...	Microsoft Corporation	C:\Windows\System32\mswsock.dll

Recon on MS RDP client

- Additional (human-readable) resources: publicly available documentations
  - https://github.com/FreeRDP/FreeRDP/wiki/Reference-Documentation
  - 56 documentations about RDP specification is out there



Reference Documentation

Remote Desktop Protocol:

Windows Communication Protocols

[MS-RDPBCGR]: Remote Desktop Protocol: Basic Connectivity and Graphics Remoting Specification
[MS-RDPCR2]: Remote Desktop Protocol: Composited Remoting V2 Specification
[MS-RDPEA]: Remote Desktop Protocol: Audio Output Virtual Channel Extension
[MS-RDPEAI]: Remote Desktop Protocol: Audio Input Redirection Virtual Channel Extension
[MS-RDPECLIP]: Remote Desktop Protocol: Clipboard Virtual Channel Extension
[MS-RDPEDC]: Remote Desktop Protocol: Desktop Composition Virtual Channel Extension
[MS-RDPEDYC]: Remote Desktop Protocol: Dynamic Channel Virtual Channel Extension
[MS-RDPEECO]: Remote Desktop Protocol: Virtual Channel Echo Extension

Recon on MS RDP client

- We may also get hints for the RDP protocol from one of the open-source implementation: FreeRDP
  - https://github.com/FreeRDP/FreeRDP
  - Read the code, headers, data types, enums, etc. -> enables more efficient reversing

FreeRDP: A Remote Desktop Protocol Implementation

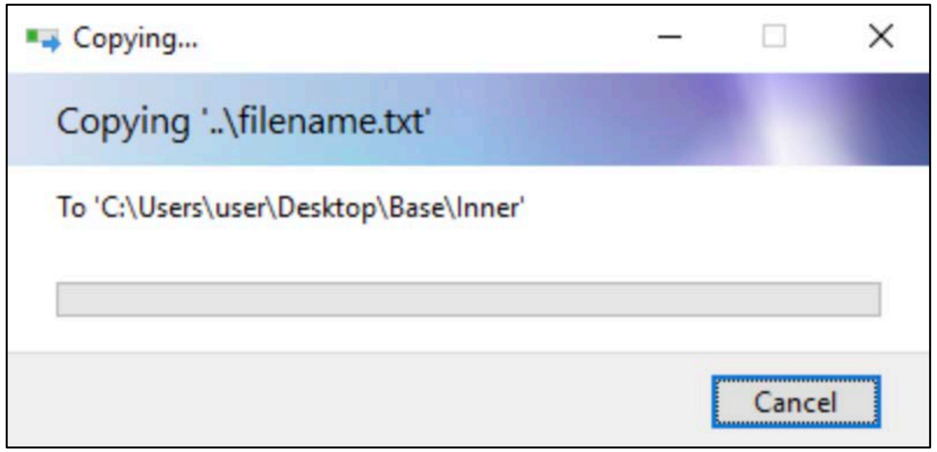
FreeRDP is a free implementation of the Remote Desktop Protocol (RDP), released under the Apache license. Enjoy the freedom of using your software wherever you want, the way you want it, in a world where interoperability can finally liberate your computing experience.

IRC: #freerdp @ irc.freenode.net

Recon on MS RDP client

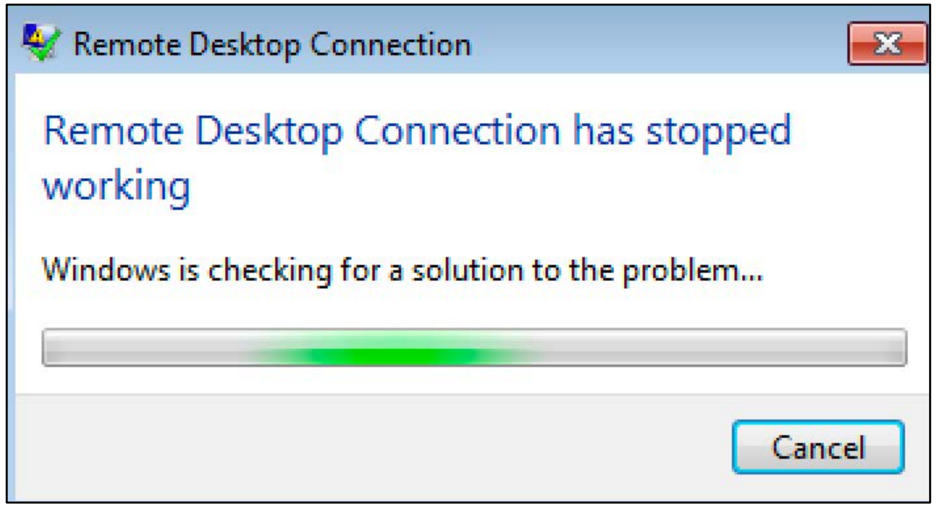
- Latest Attack Case by CHECKPOINT
- Attacked the clipboard channel : Ctrl C + Ctrl V
  - A Path-traversal attack!
  - Poisoned RDP at BH USA 2019

- In contrast, our focus is at:
  - **Memory Corruption vulns in the MS RDP Client that allows Remote Code Execution**


Copying...

Copying '..\filename.txt'

To 'C:\Users\user\Desktop\Base\Inner'

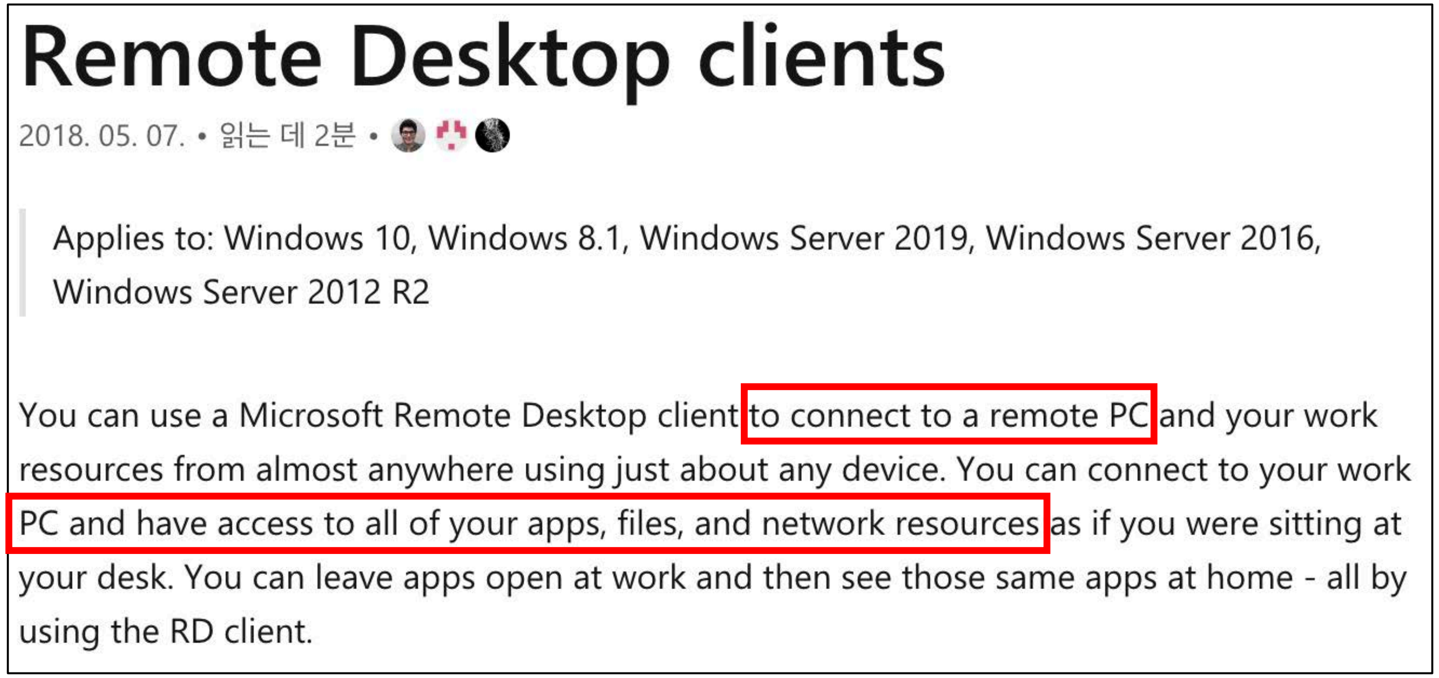

Remote Desktop Connection

Remote Desktop Connection has stopped working

Windows is checking for a solution to the problem...

Recon on MS RDP client

- Available resources to the RDP client



Remote Desktop clients

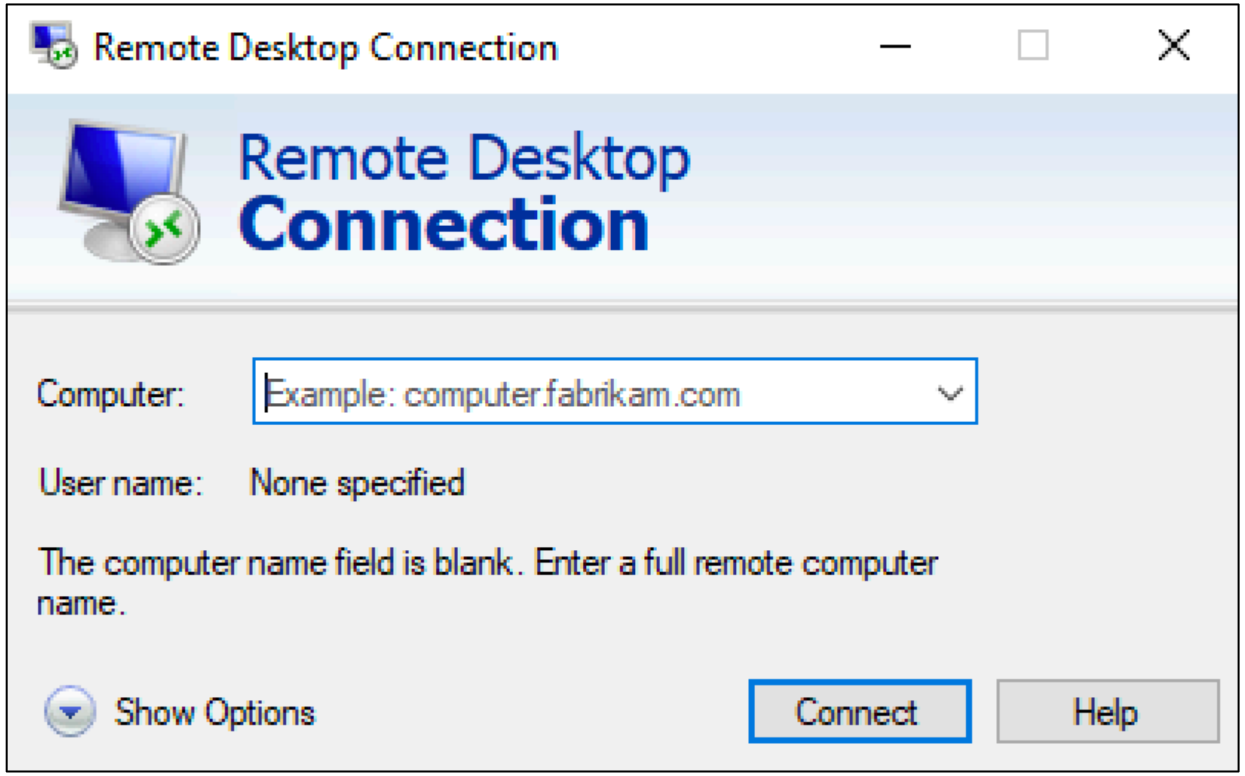
2018. 05. 07. • 읽는 데 2분 •

> Applies to: Windows 10, Windows 8.1, Windows Server 2019, Windows Server 2016, Windows Server 2012 R2

You can use a Microsoft Remote Desktop client to connect to a remote PC and your work resources from almost anywhere using just about any device. You can connect to your work PC and have access to all of your apps, files, and network resources as if you were sitting at your desk. You can leave apps open at work and then see those same apps at home - all by using the RD client.

Recon on MS RDP client

- How many (official) RDP Clients are there?
- 5 types by OS/App
  - **mstsc.exe (native app)**
  - Windows 10 app (AppContainer)
  - Android App
  - iOS App
  - macOS App



Remote Desktop Connection

Remote Desktop **Connection**

Computer: Example: computer.fabrikam.com

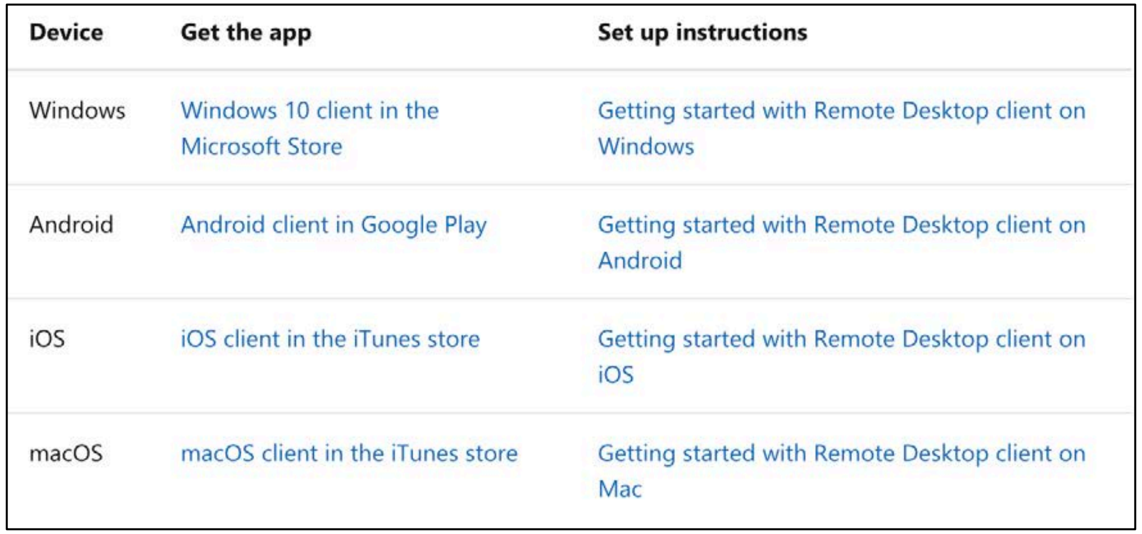
User name: None specified

The computer name field is blank. Enter a full remote computer name.

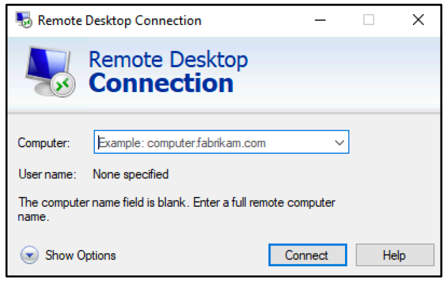
Recon on MS RDP client

- How many (official) RDP Clients are there?
- 5 types by OS/App
  - **mstsc.exe (native app)**
  - Windows 10 app (AppContainer)
  - Android App
  - iOS App
  - macOS App

We assume all these implementation will share the same code base for the protocol



Device	Get the app	Set up instructions
Windows	Windows 10 client in the Microsoft Store	Getting started with Remote Desktop client on Windows
Android	Android client in Google Play	Getting started with Remote Desktop client on Android
iOS	iOS client in the iTunes store	Getting started with Remote Desktop client on iOS
macOS	macOS client in the iTunes store	Getting started with Remote Desktop client on Mac



Remote Desktop Connection

Remote Desktop **Connection**

Computer: Example: computer.fabrikam.com

User name: None specified

The computer name field is blank. Enter a full remote computer name.

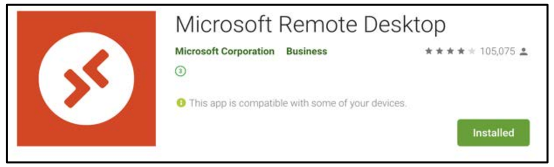
Microsoft Remote Desktop

Microsoft Corporation • **Productivity**

♡ Wish list

★★★★☆ 811

Use the Microsoft Remote Desktop app to connect to a remote PC or virtual apps and desktops made available by your admin. The app helps you be productive no matter where you are.



Microsoft Remote Desktop

Microsoft Corporation Business ★★★★☆ 105,075

This app is compatible with some of your devices.

Mac App Store Preview

Open the Mac App Store to buy and download apps.

Microsoft Remote Desktop 10 4+

Get work done from anywhere

Microsoft Corporation

★★★☆☆ 2.9, 724 Ratings

Free

This app is only available on the App Store for iOS devices.

Microsoft Remote Desktop 4+

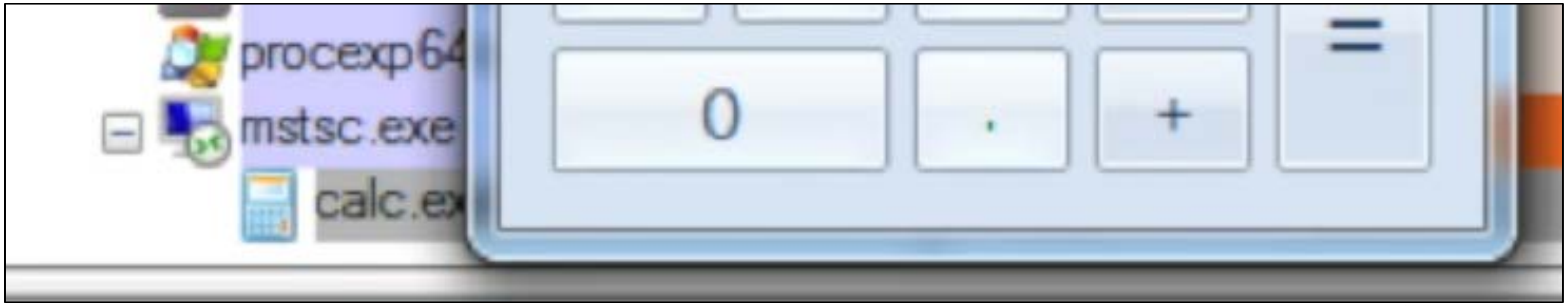
Microsoft Corporation

★★★☆☆ 2.9, 708 Ratings

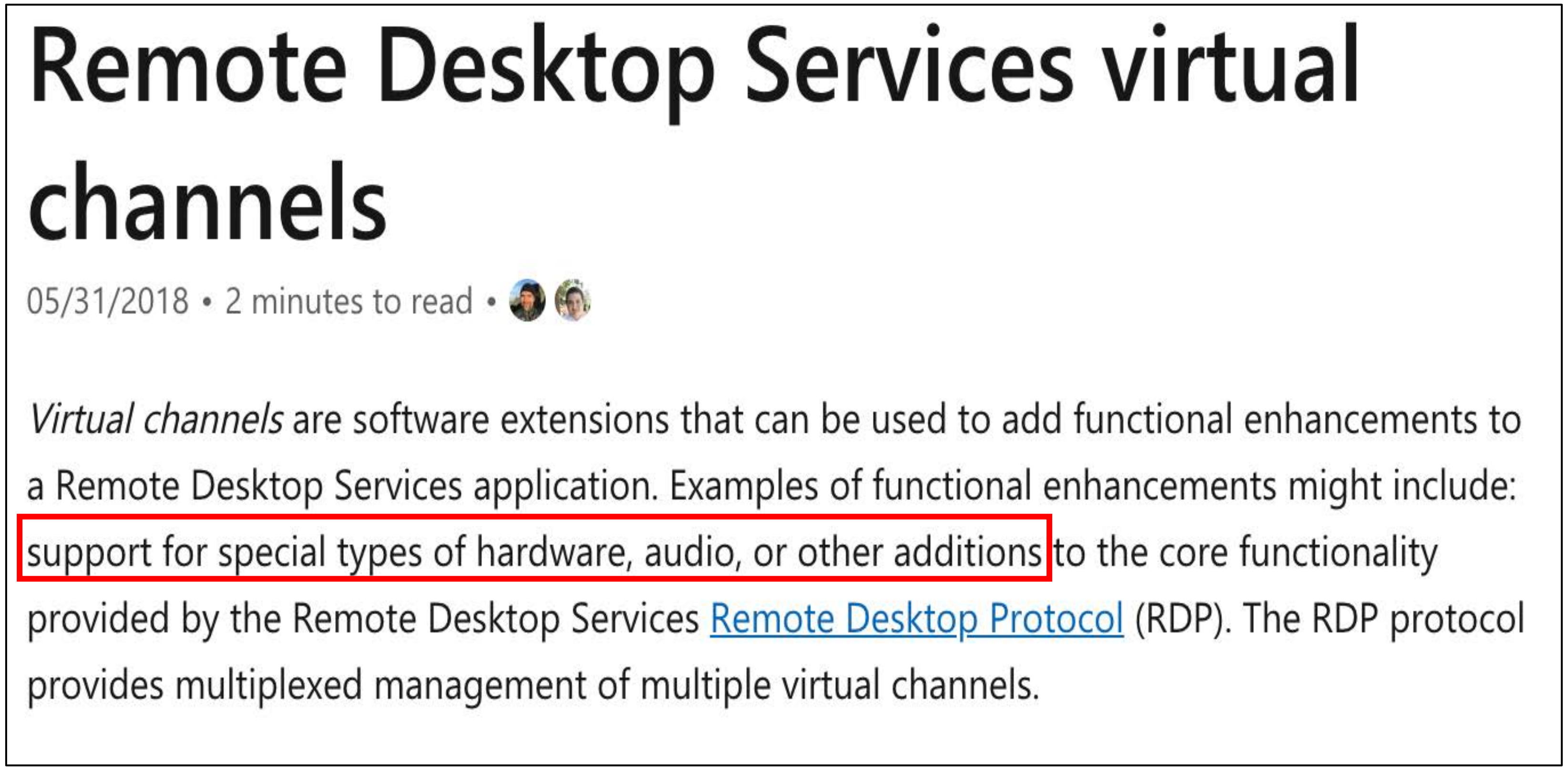
Free

Our GOAL

- Find an exploitable memory corruption vulnerability in official RDP clients
- Popping shell from the RDP Client
  - Need to find the part of the program that handles 'memory' a lot
    - Intuition: more code for handling memory, more memory corruption vulnerabilities!
  - We purposely send malicious data from a compromised RDP Server to a client using virtual channels



What is Virtual Channel?



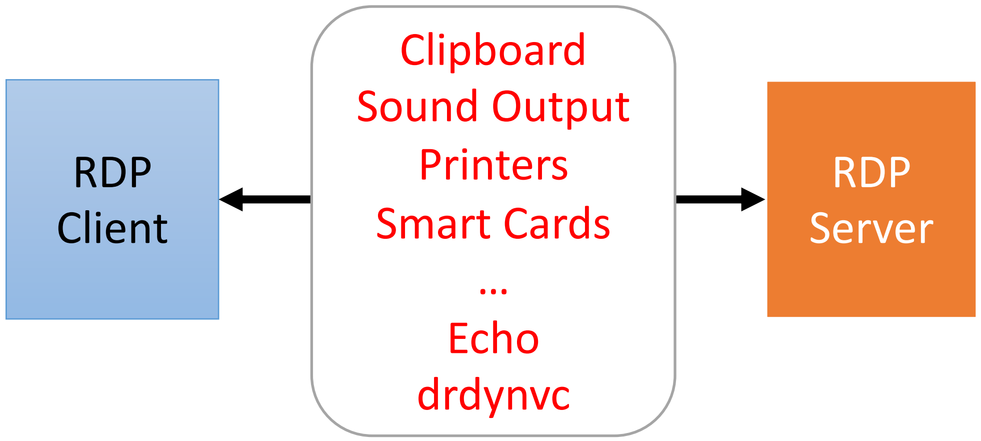
Remote Desktop Services virtual channels

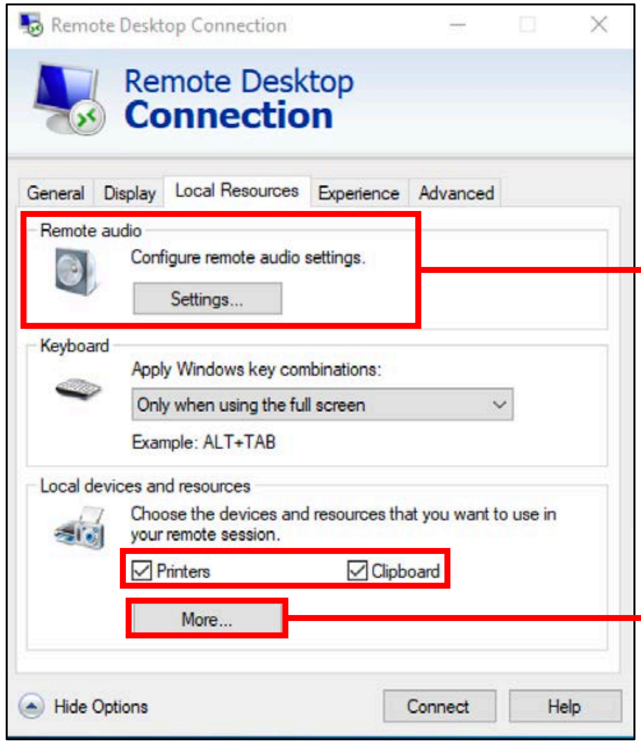
05/31/2018 • 2 minutes to read •

Virtual channels are software extensions that can be used to add functional enhancements to a Remote Desktop Services application. Examples of functional enhancements might include: support for special types of hardware, audio, or other additions to the core functionality provided by the Remote Desktop Services Remote Desktop Protocol (RDP). The RDP protocol provides multiplexed management of multiple virtual channels.

Virtual Channel Examples

- Default Channels

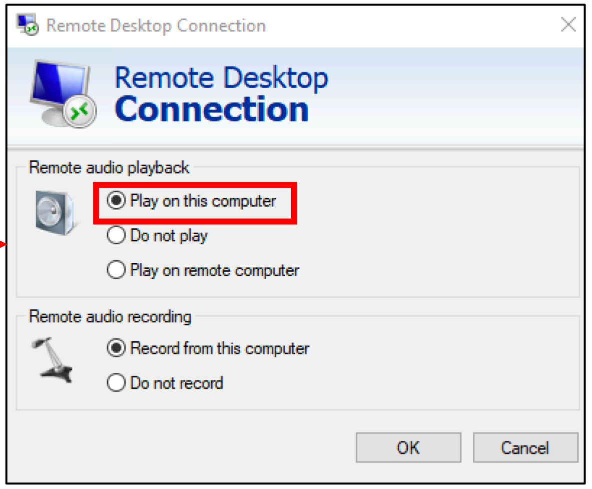




Remote audio
Configure remote audio settings.

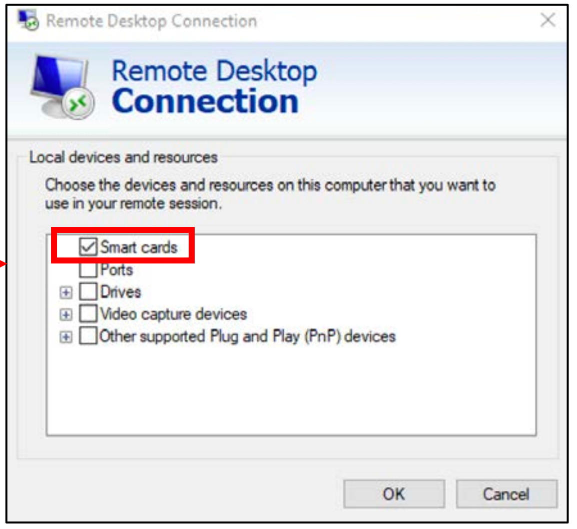
Keyboard
Apply Windows key combinations:
Only when using the full screen
Example: ALT+TAB

Local devices and resources
Choose the devices and resources that you want to use in your remote session.
[✓ Printers] [✓ Clipboard]



Remote audio playback
[✓ Play on this computer]
Do not play
Play on remote computer

Remote audio recording
[✓ Record from this computer]
Do not record

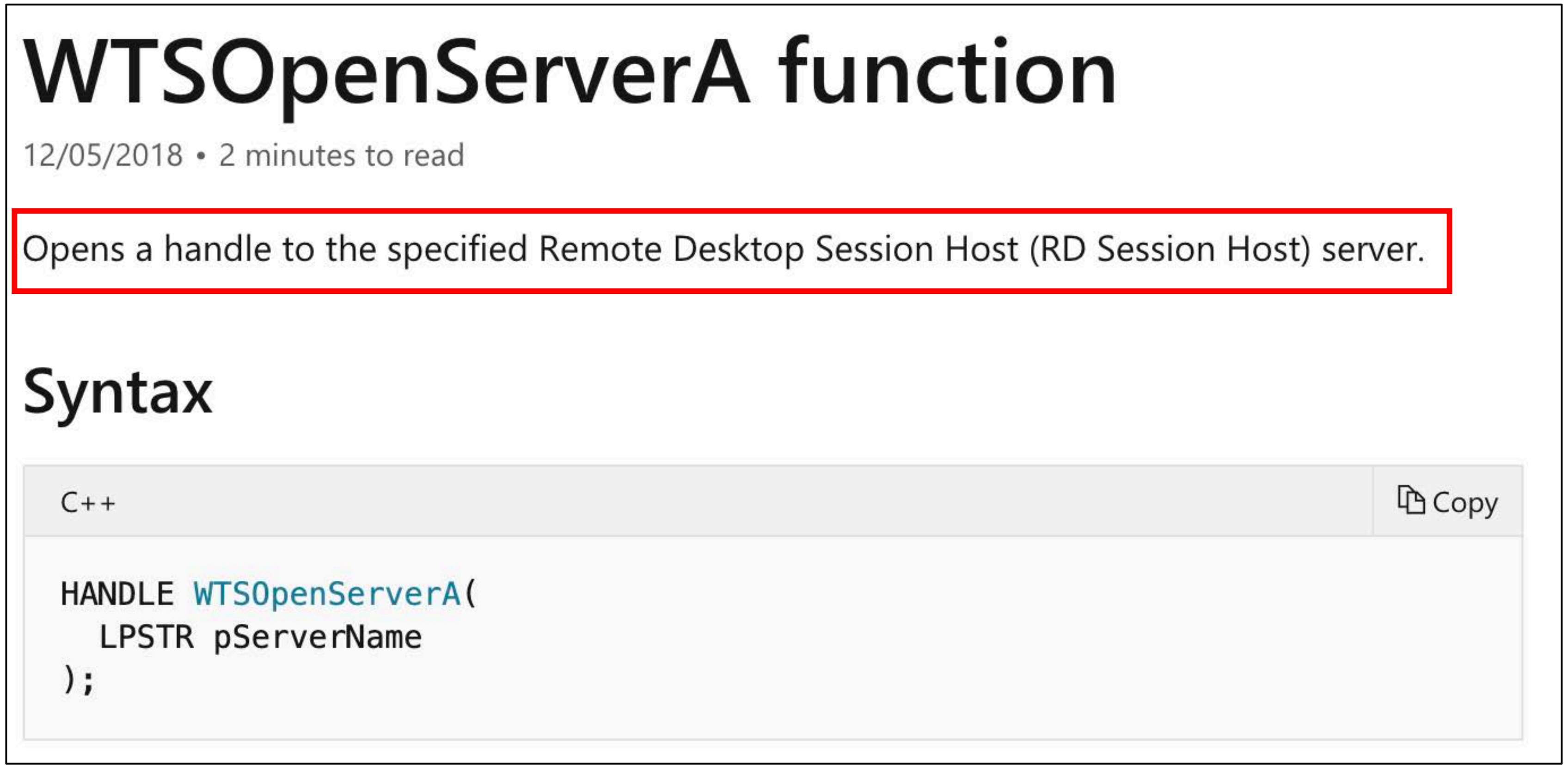


Local devices and resources
Choose the devices and resources on this computer that you want to use in your remote session.
[✓ Smart cards]
Ports
Drives
Video capture devices
Other supported Plug and Play (PnP) devices

Internally, handing of such features includes protocol parsing logic, lots of heap memory allocation/free, etc., so it's a great target for fuzzing!

Virtual Channel Protocol

- **Open Server**
- Open Channel
- Read Channel
- Write Channel
- Close Channel
- Close Server



WTSOpenServerA function

12/05/2018 • 2 minutes to read

Opens a handle to the specified Remote Desktop Session Host (RD Session Host) server.

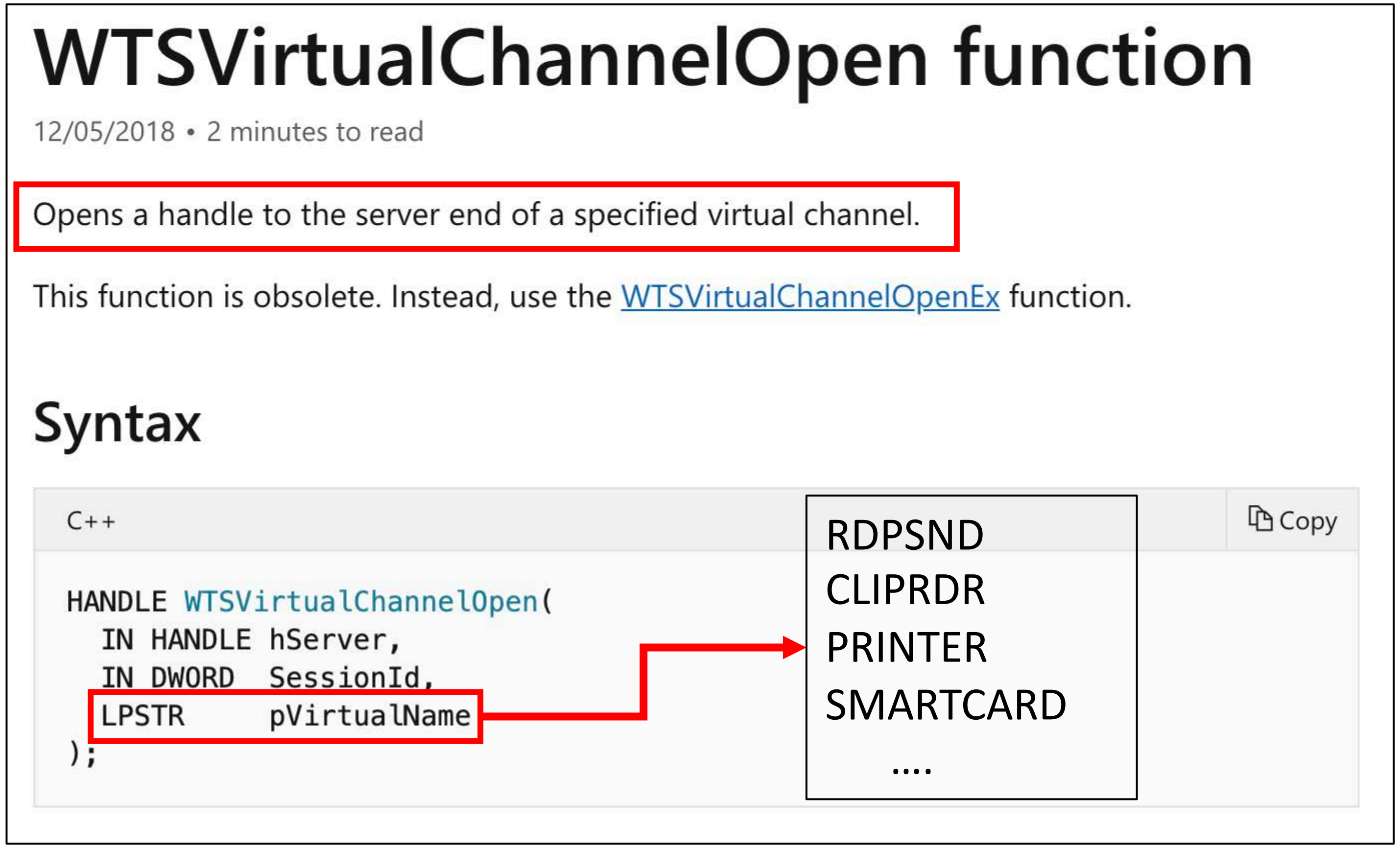
Syntax

C++

```cpp
HANDLE WTSOpenServerA(
  LPSTR pServerName
);
```

Virtual Channel Protocol

- Open Server
- **Open Channel**
- Read Channel
- Write Channel
- Close Channel
- Close Server



WTSVirtualChannelOpen function

12/05/2018 • 2 minutes to read

Opens a handle to the server end of a specified virtual channel.

This function is obsolete. Instead, use the WTSVirtualChannelOpenEx function.

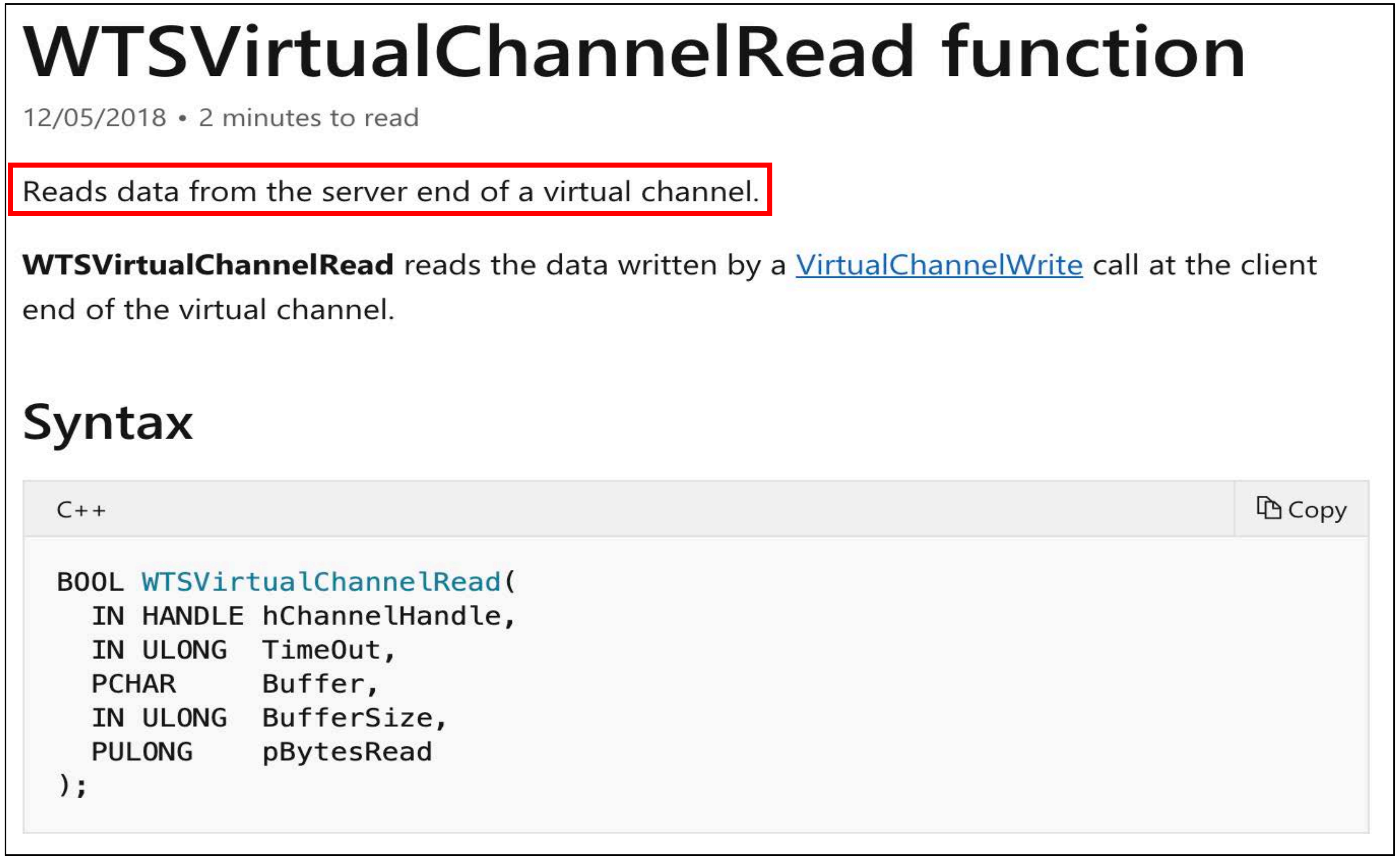
Syntax

C++ Copy

```
HANDLE WTSVirtualChannelOpen(
  IN HANDLE hServer,
  IN DWORD  SessionId,
  LPSTR     pVirtualName
);
```

Virtual Channel Protocol

- Open Server
- Open Channel
- **Read Channel**
- Write Channel
- Close Channel
- Close Server



WTSVirtualChannelRead function

12/05/2018 • 2 minutes to read

Reads data from the server end of a virtual channel.

WTSVirtualChannelRead reads the data written by a VirtualChannelWrite call at the client end of the virtual channel.

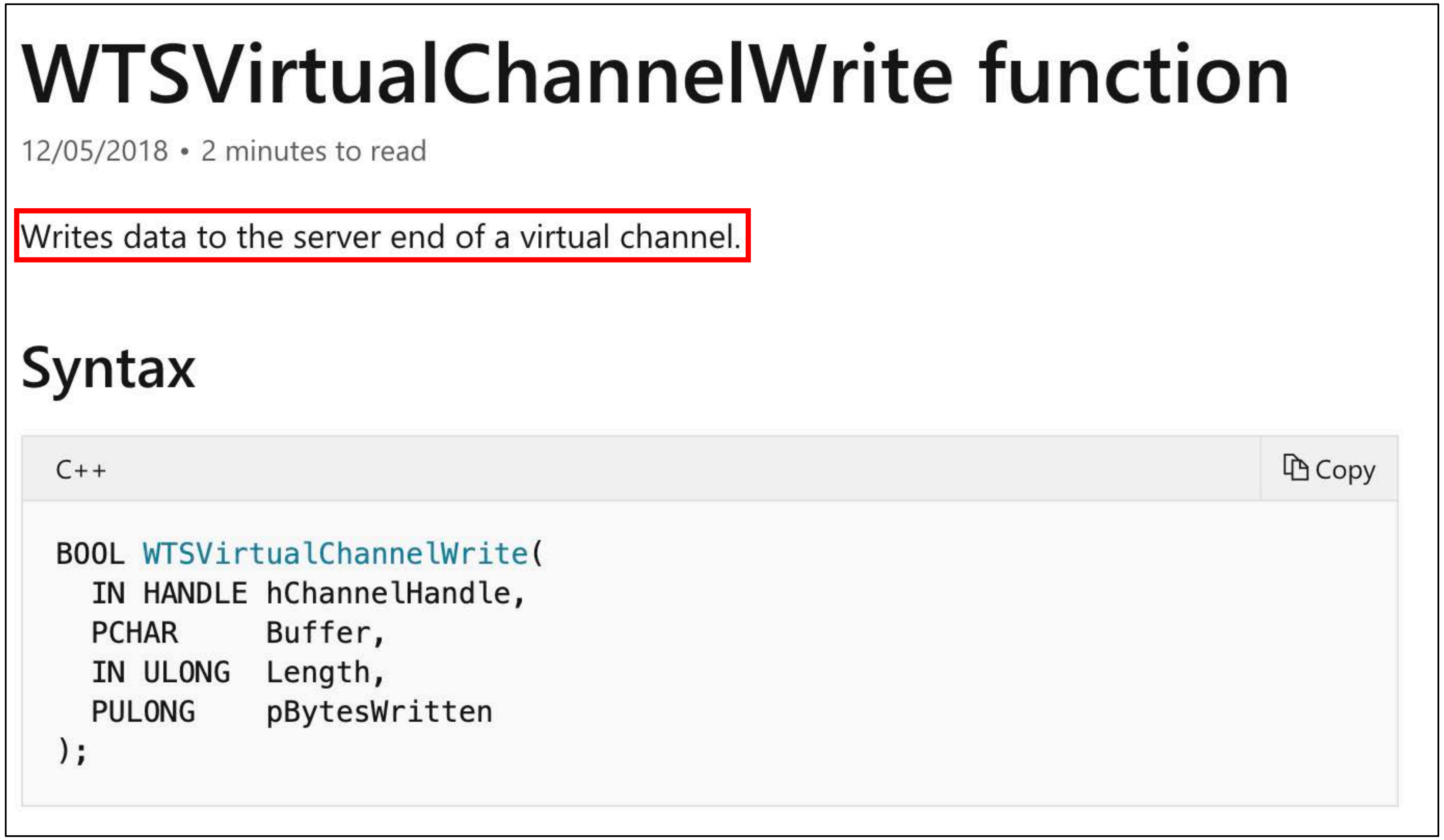
Syntax

C++ Copy

```cpp
BOOL WTSVirtualChannelRead(
  IN HANDLE hChannelHandle,
  IN ULONG  TimeOut,
  PCHAR     Buffer,
  IN ULONG  BufferSize,
  PULONG    pBytesRead
);
```

Virtual Channel Protocol

- Open Server
- Open Channel
- Read Channel
- **Write Channel**
- Close Channel
- Close Server



WTSVirtualChannelWrite function

12/05/2018 • 2 minutes to read

Writes data to the server end of a virtual channel.

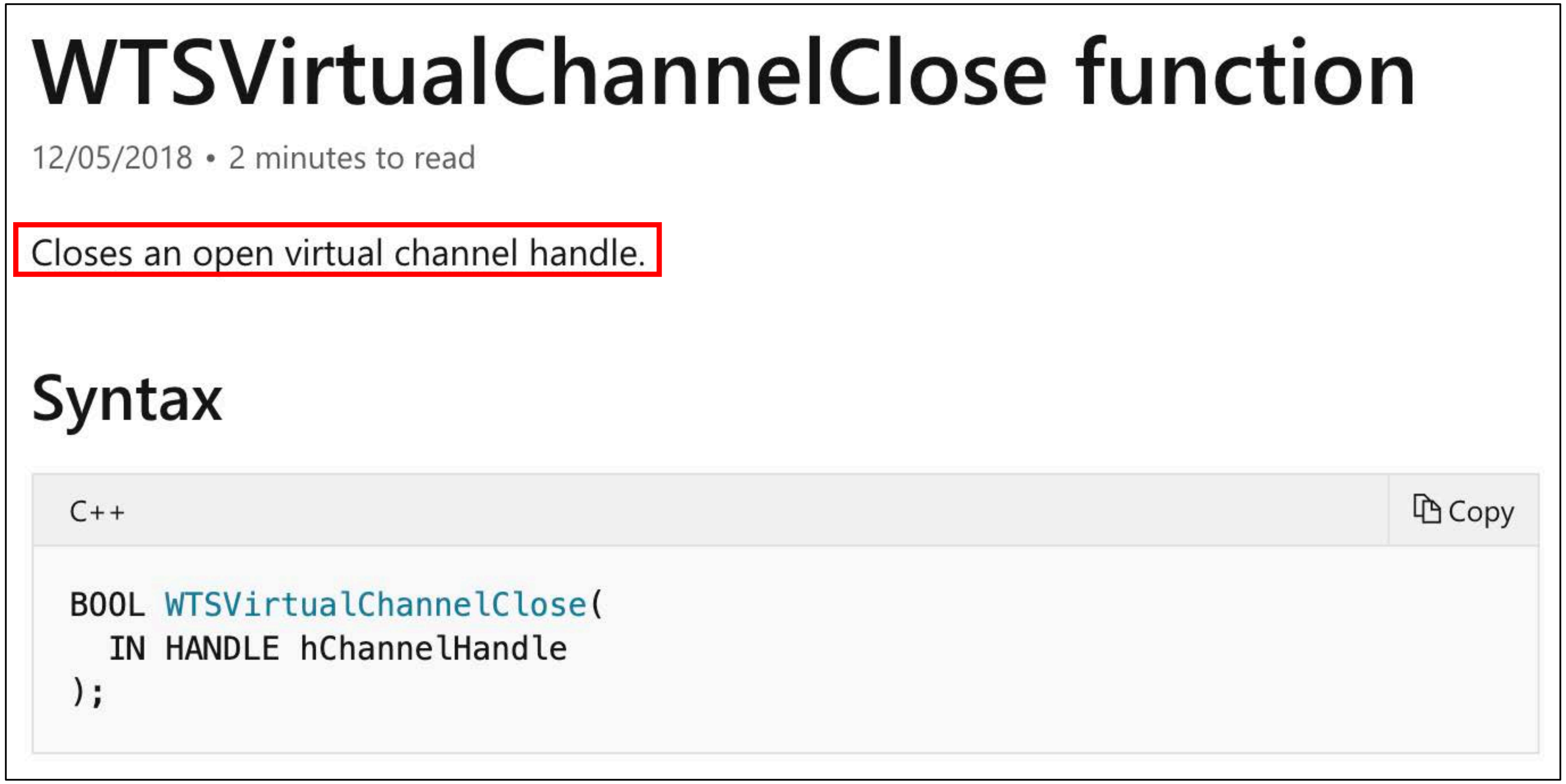
Syntax

C++ Copy

```
BOOL WTSVirtualChannelWrite(
  IN HANDLE hChannelHandle,
  PCHAR     Buffer,
  IN ULONG  Length,
  PULONG    pBytesWritten
);
```

Virtual Channel Protocol

- Open Server
- Open Channel
- Read Channel
- Write Channel
- **Close Channel**
- Close Server



WTSVirtualChannelClose function

12/05/2018 • 2 minutes to read

Closes an open virtual channel handle.

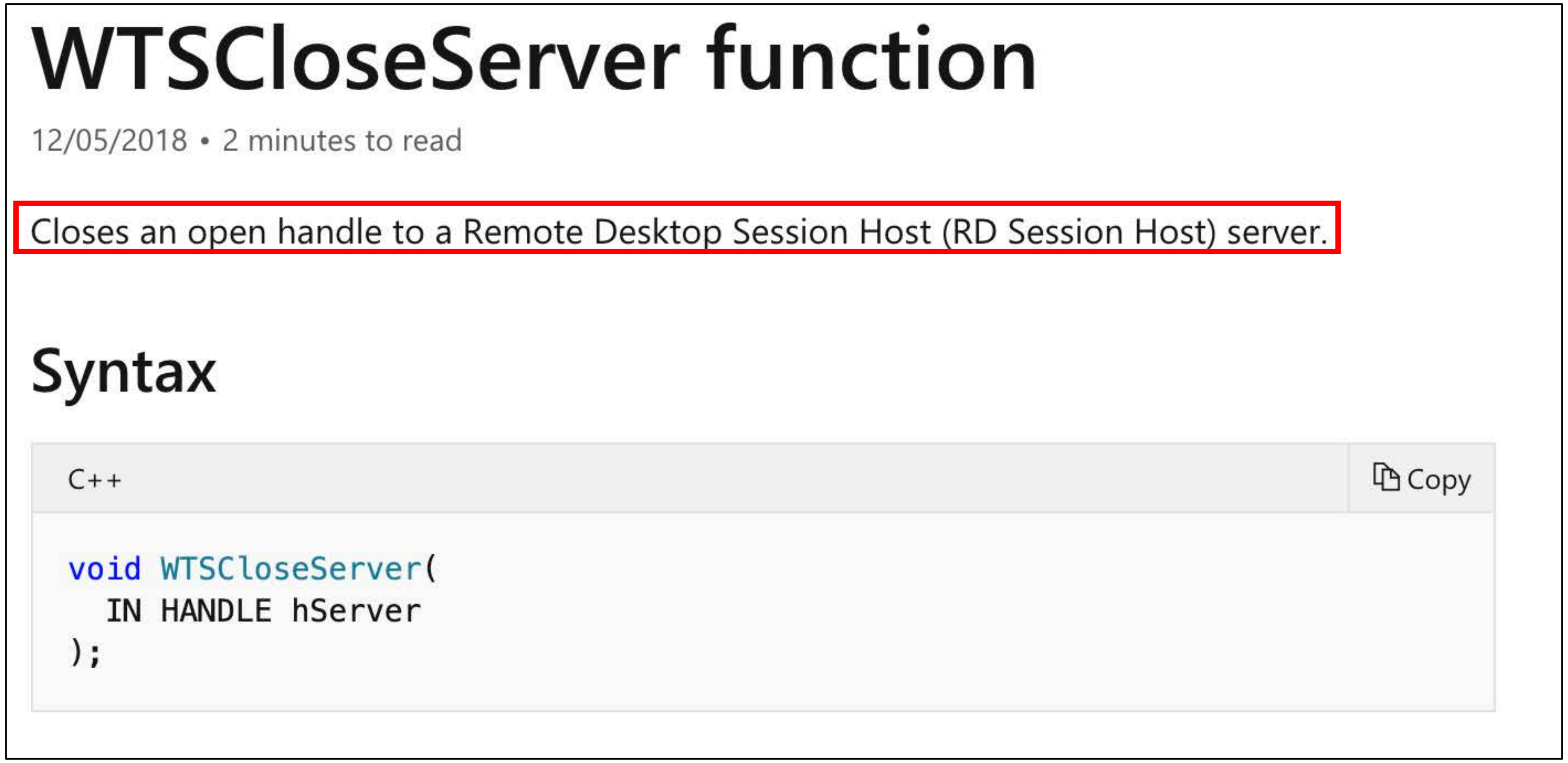
Syntax

C++ Copy

```
BOOL WTSVirtualChannelClose(
  IN HANDLE hChannelHandle
);
```

Virtual Channel Protocol

- Open Server
- Open Channel
- Read Channel
- Write Channel
- Close Channel
- **Close Server**



WTSCloseServer function

12/05/2018 • 2 minutes to read

Closes an open handle to a Remote Desktop Session Host (RD Session Host) server.

Syntax

C++

```
void WTSCloseServer(
  IN HANDLE hServer
);
```

Virtual Channel Protocol

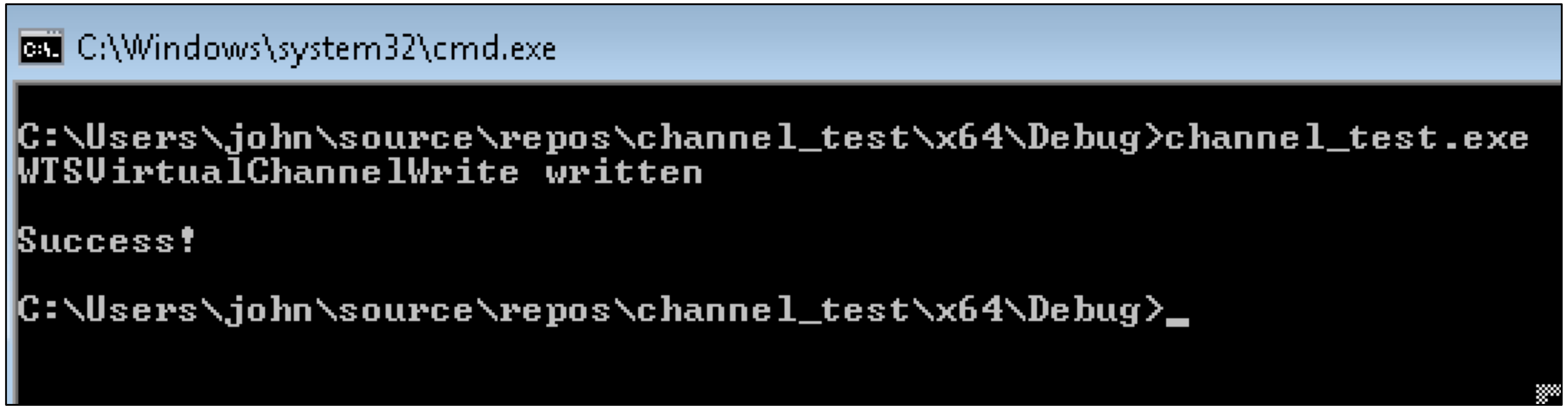
- **Open Server**
- **Open Channel**
- Read Channel
- **Write Channel**
- Close Channel
- Close Server

```
hWTSSERVERHandle = WTSOpenServer(“localhost");
hVirtualChannel = WTSVirtualChannelOpen(
                    hWTSSERVERHandle,
                    WTS_CURRENT_SESSION,
                    "RDPSND”
                    );

buffer[0] = 0xa0;
buffer[1] = 0xa1;
buffer[2] = 0xa2;
buffer[3] = 0xa3;

BufferSize = 4;
bSuccess = WTSVirtualChannelWrite(hVirtualChannel, buffer,
BufferSize, &bytesWritten);
```

Virtual Channel Protocol

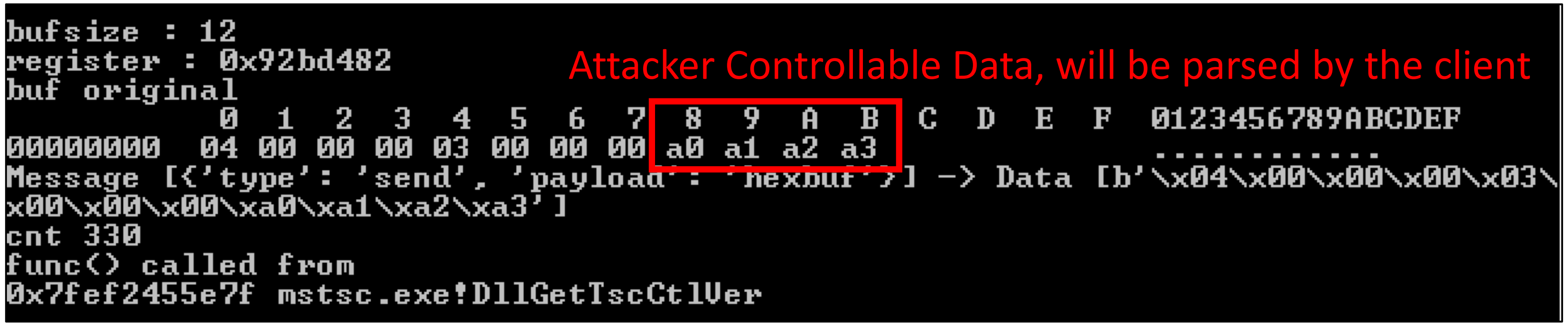
- **Open Server**
- **Open Channel**
- Read Channel
- **Write Channel**
- Close Channel
- Close Server

1. Run virtual channel write test program



```
C:\Users\john\source\repos\channel_test\x64\Debug>channel_test.exe
WTSVirtualChannelWrite written

Success!

C:\Users\john\source\repos\channel_test\x64\Debug>_
```

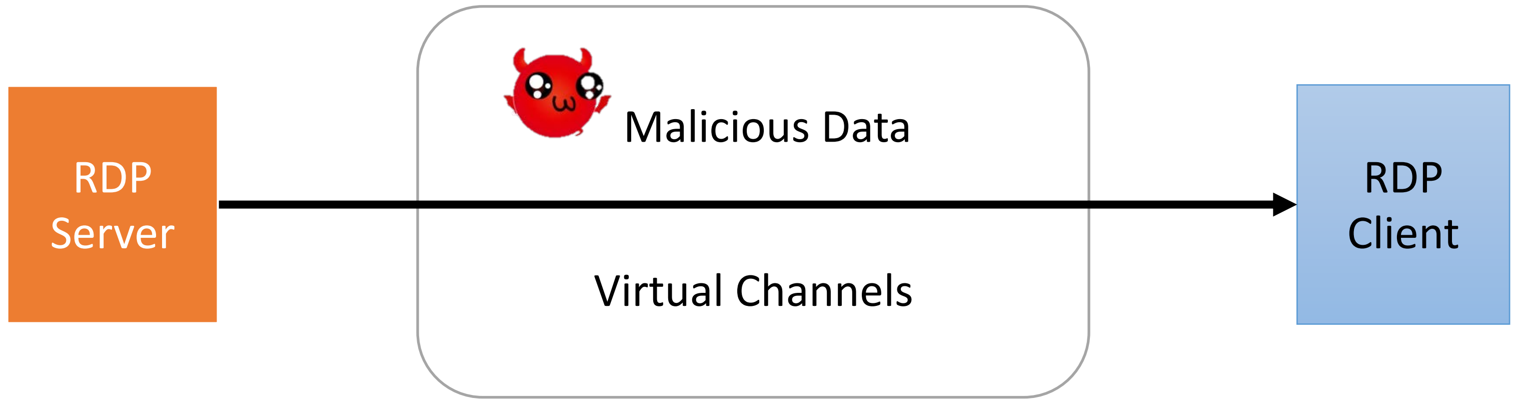
2. Hooking mstscax!CChan::ChannelOnPacketReceived()



```
bufsize : 12
register : 0x92bd482
buf original
           0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F  0123456789ABCDEF
00000000  04 00 00 00 03 00 00 00 a0 a1 a2 a3              ............
Message [{'type': 'send', 'payload': 'hexbuf'}] -> Data [b'\x04\x00\x00\x00\x03\
x00\x00\x00\xa0\xa1\xa2\xa3']
cnt 330
func() called from
0x7fef2455e7f mstsc.exe!DllGetTscCtlVer
```

Attack Scenario via Manipulating the Virtual Channel

- From server to client using virtual channel
  - 1) Open server
  - 2) Open channel
  - 3) Write channel
  - 4) Send Malicious Data to Client

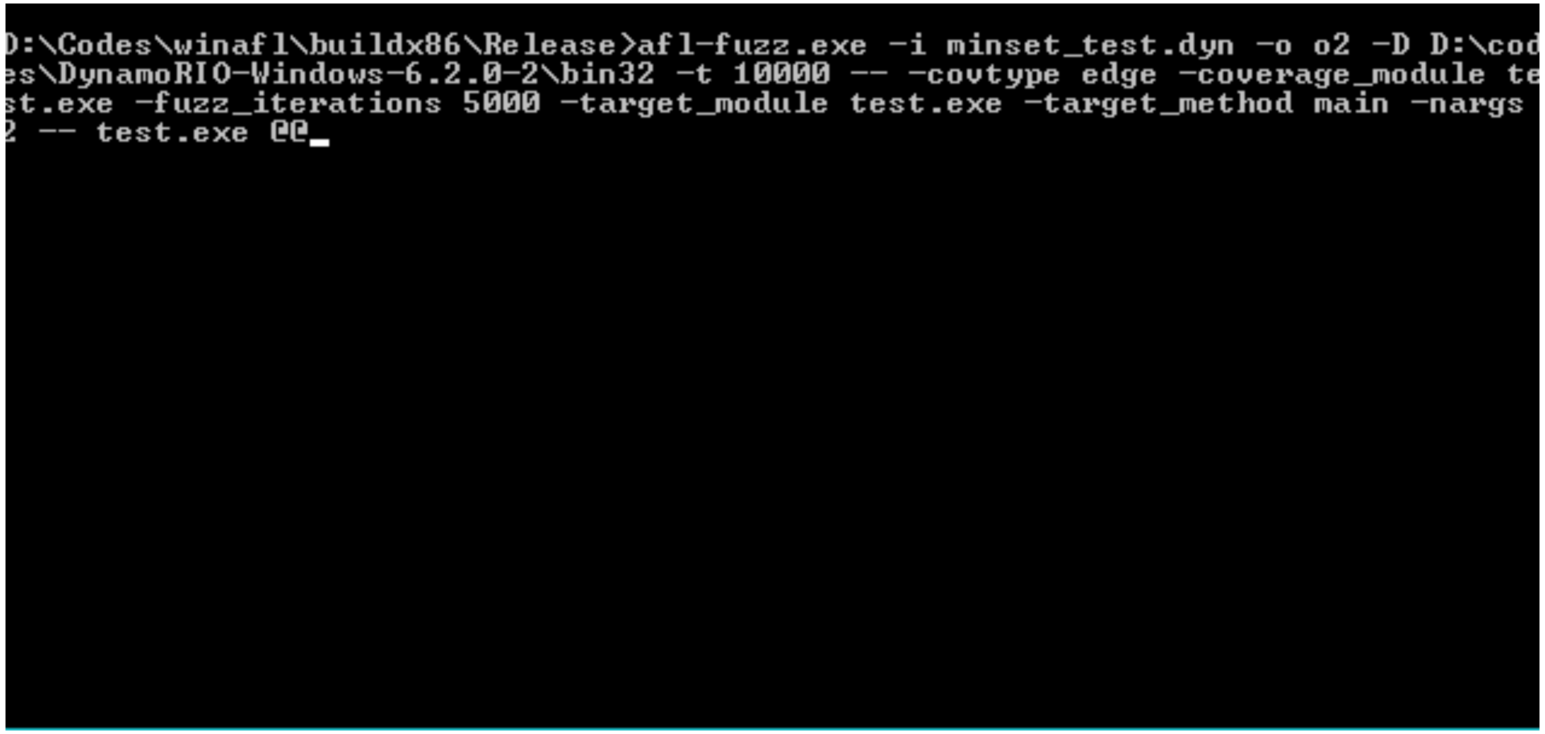


Finding Vulnerabilities Automatically: Build an RDP client fuzzer

- Requirements
  - Need to hook **Virtual Channels**
  - Need to work with a server-client model
  - Need to apply a blackbox fuzzing
  - On Windows
  - But we would like to enable coverage-guided fuzzing (like AFL)!
  - …

Finding Vulnerabilities Automatically: Build an RDP client fuzzer

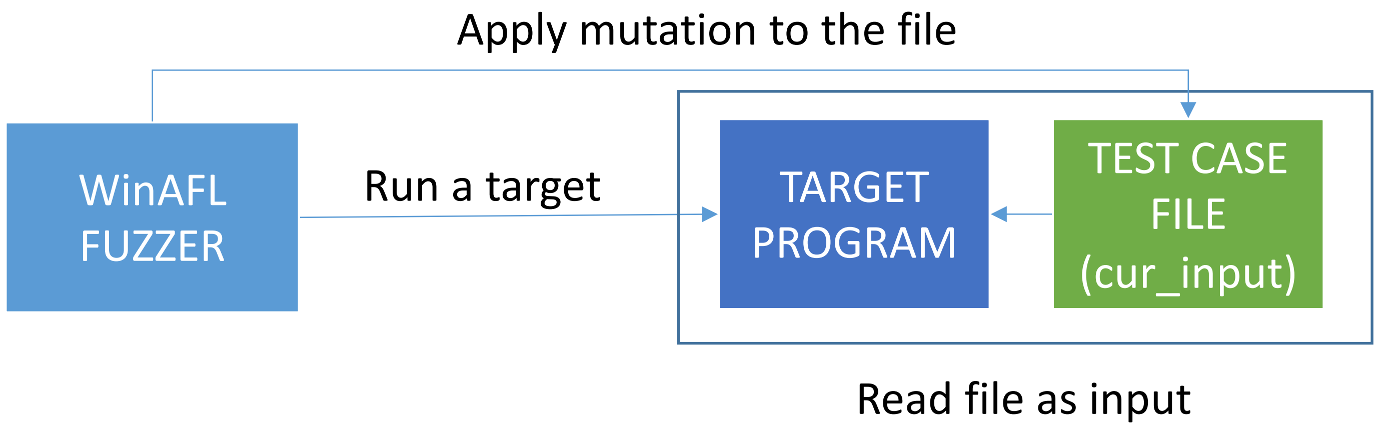
- Available coverage-guided fuzzing tool on Windows
  - WinAFL !



```
D:\Codes\winafl\buildx86\Release>afl-fuzz.exe -i minset_test.dyn -o o2 -D D:\cod
es\DynamoRIO-Windows-6.2.0-2\bin32 -t 10000 -- -covtype edge -coverage_module te
st.exe -fuzz_iterations 5000 -target_module test.exe -target_method main -nargs
2 -- test.exe @@_
```

Finding Vulnerabilities Automatically: Build an RDP client fuzzer

- WinAFL
- A fork of AFL for fuzzing Windows binaries
  - afl-fuzz [afl options] -- [instrumentation options] -- target_cmd_line
    - `afl-fuzz.exe -i in -o out -D C:\work\winafl\DynamoRIO\bin64 -t 20000 -- -coverage_module gdiplus.dll -coverage_module WindowsCodecs.dll -fuzz_iterations 5000 -target_module test_gdiplus.exe -target_offset 0x16e0 -nargs 2 -- test_gdiplus.exe @@`



Finding Vulnerabilities Automatically: Build an RDP client fuzzer

- WinAFL: slightly different with AFL

  -
    ```
    afl-fuzz.exe -i in -o out -D C:\work\winafl\DynamoRIO\bin64 -t 20000 -- -
    coverage_module gdiplus.dll -coverage_module WindowsCodecs.dll -
    fuzz_iterations 5000 -target_module test_gdiplus.exe -target_offset 0x16e0 -
    nargs 2 -- test_gdiplus.exe @@
    ```
- Need to specify target module and offset to hook the function and measure the code coverage

WinAFL ≠ AFL

Finding Vulnerabilities Automatically: Build an RDP client fuzzer

- WinAFL
  - It supports the following instrumentation modes:

- Dynamic instrumentation using DynamoRIO
- Hardware tracing using Intel PT
- Static instrumentation via Syzygy

Finding Vulnerabilities Automatically: Build an RDP client fuzzer

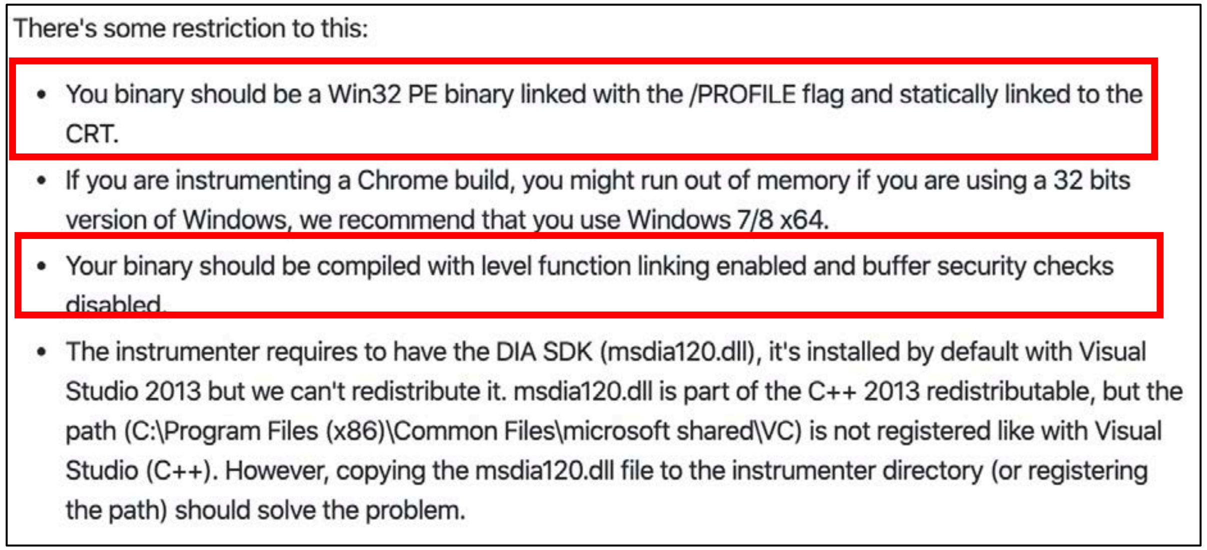
- WinAFL
  - It supports the following instrumentation modes:
- Dynamic instrumentation using DynamoRIO
  - Drawback: slow
- Hardware tracing using Intel PT
- Static instrumentation via Syzygy

Finding Vulnerabilities Automatically: Build an RDP client fuzzer

- WinAFL
  - It supports the following instrumentation modes:
- Dynamic instrumentation using DynamoRIO
- Hardware tracing using Intel PT
  - Drawback: need a CPU that supports PT, and cannot run fuzzer in the VM
- Static instrumentation via Syzygy

Finding Vulnerabilities Automatically: Build an RDP client fuzzer

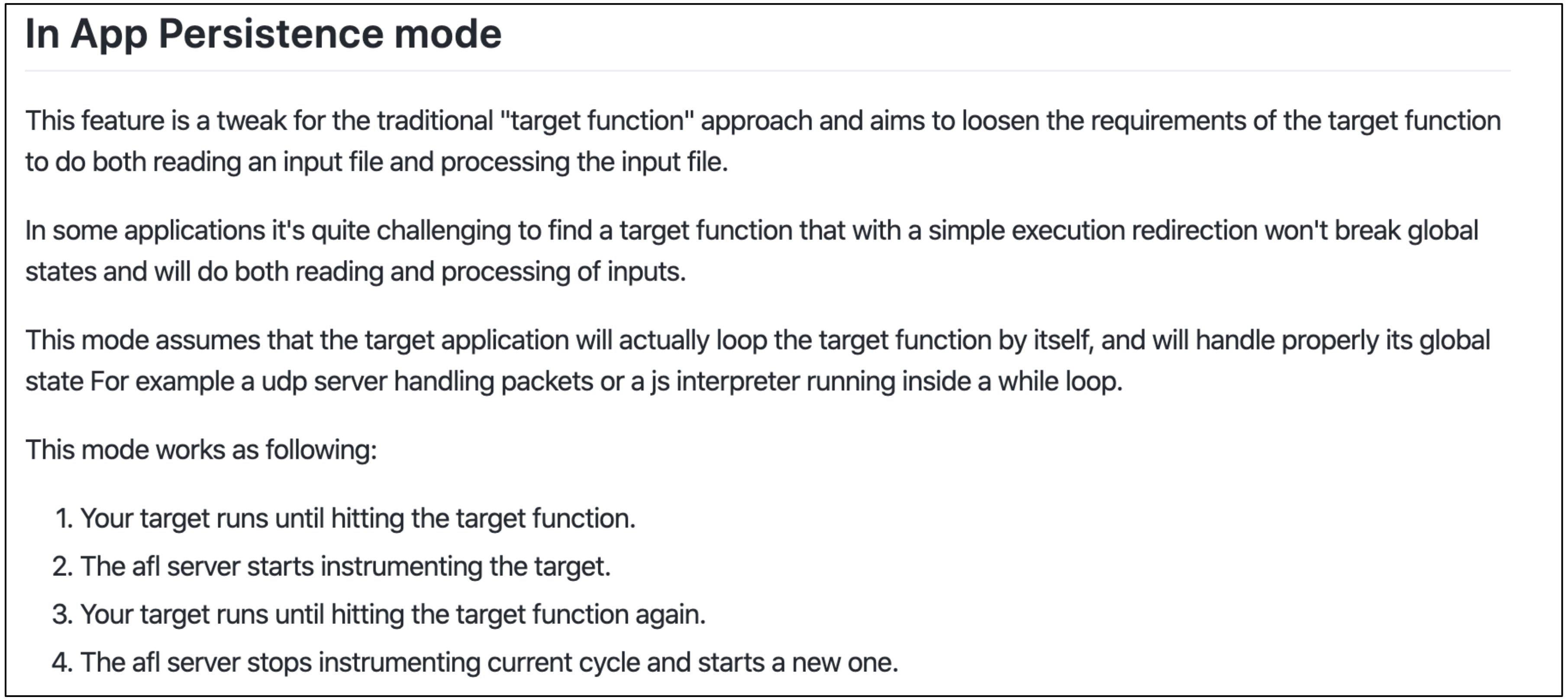
- WinAFL
  - It supports the following instrumentation modes:
- Dynamic instrumentation using DynamoRIO
- Hardware tracing using Intel PT
- Static instrumentation via Syzygy
  - Drawback: some restrictions..


There's some restriction to this:

- You binary should be a Win32 PE binary linked with the /PROFILE flag and statically linked to the CRT.
- If you are instrumenting a Chrome build, you might run out of memory if you are using a 32 bits version of Windows, we recommend that you use Windows 7/8 x64.
- Your binary should be compiled with level function linking enabled and buffer security checks disabled.
- The instrumenter requires to have the DIA SDK (msdia120.dll), it's installed by default with Visual Studio 2013 but we can't redistribute it. msdia120.dll is part of the C++ 2013 redistributable, but the path (C:\Program Files (x86)\Common Files\microsoft shared\VC) is not registered like with Visual Studio (C++). However, copying the msdia120.dll file to the instrumenter directory (or registering the path) should solve the problem.

Finding Vulnerabilities Automatically: Build an RDP client fuzzer

- We decided to work with DynamoRIO with in-app persistent mode



In App Persistence mode

This feature is a tweak for the traditional "target function" approach and aims to loosen the requirements of the target function to do both reading an input file and processing the input file.

In some applications it's quite challenging to find a target function that with a simple execution redirection won't break global states and will do both reading and processing of inputs.

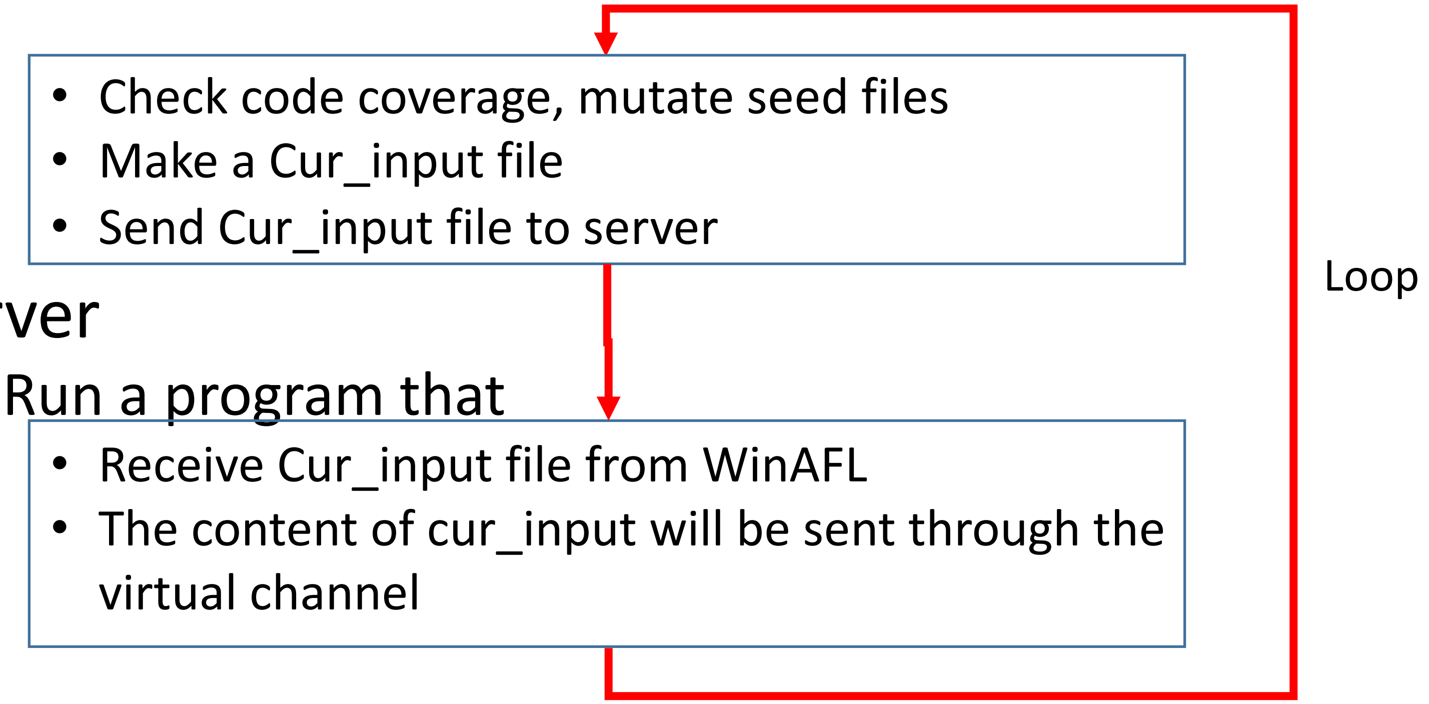
This mode assumes that the target application will actually loop the target function by itself, and will handle properly its global state For example a udp server handling packets or a js interpreter running inside a while loop.

This mode works as following:

1. Your target runs until hitting the target function.
2. The afl server starts instrumenting the target.
3. Your target runs until hitting the target function again.
4. The afl server stops instrumenting current cycle and starts a new one.

Finding Vulnerabilities Automatically: Build an RDP client fuzzer

- The fuzzer architecture for fuzzing RDP client (a modified version of WinAFL)
- Client
  - run winafl with mstsc.exe and hook mstscax.dll (for data receiving function)



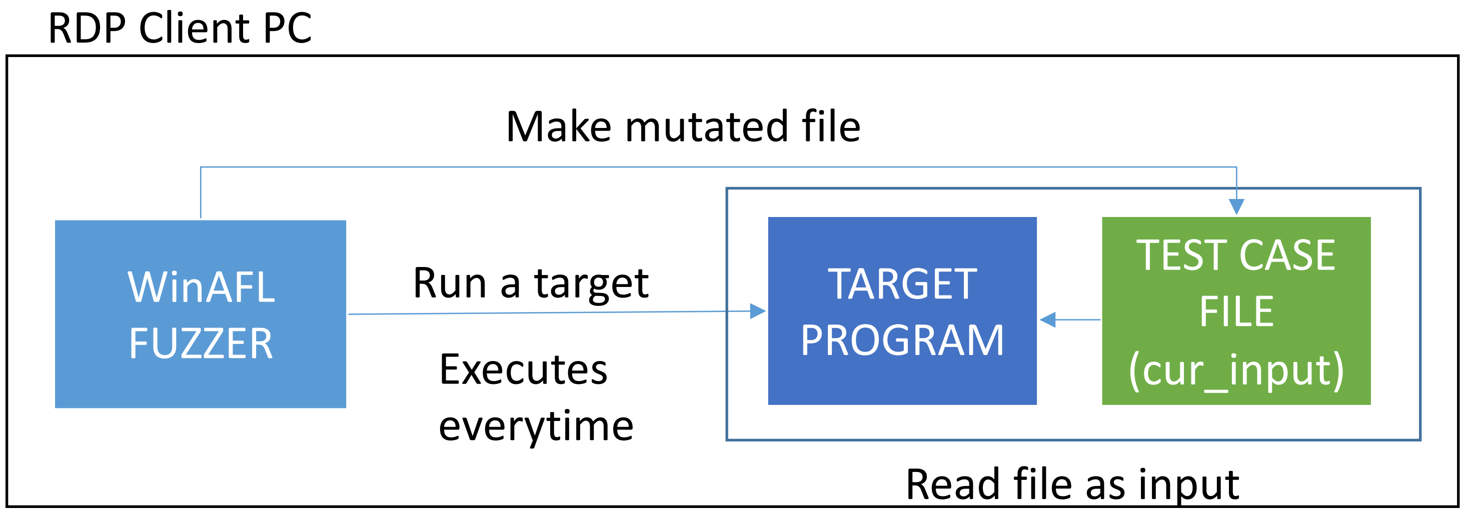
- Check code coverage, mutate seed files
- Make a Cur_input file
- Send Cur_input file to server

- Server
  - Run a program that

- Receive Cur_input file from WinAFL
- The content of cur_input will be sent through the virtual channel

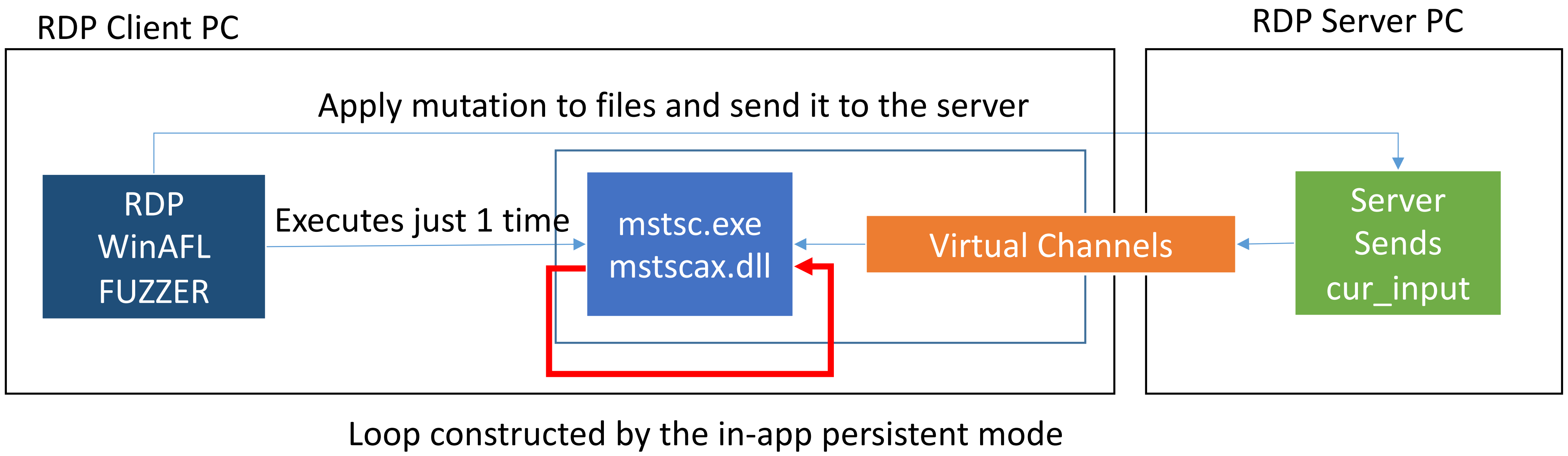
Finding Vulnerabilities Automatically: Build an RDP client fuzzer

- Original Architecture of WinAFL



Finding Vulnerabilities Automatically: Build an RDP client fuzzer

- Modified WinAFL for fuzzing the RDP client

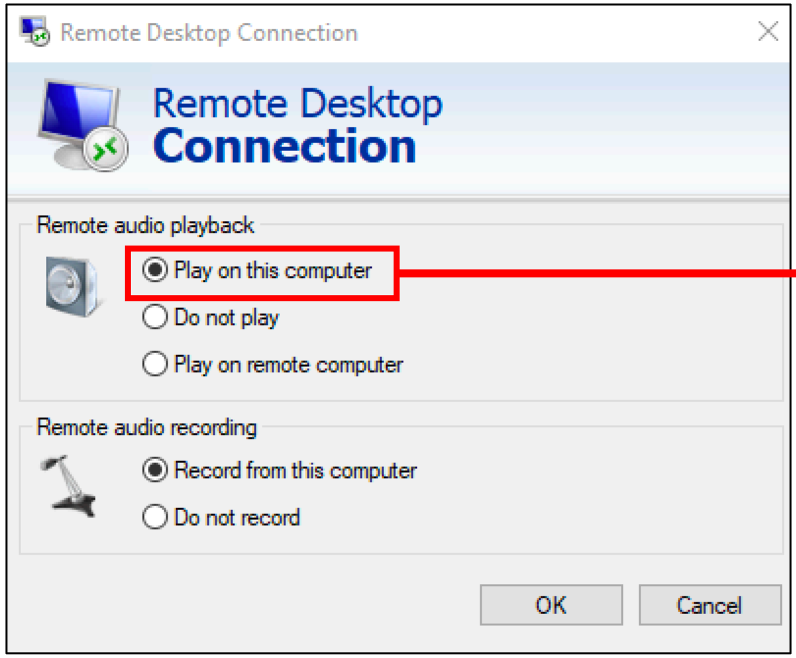


Applying the RDP Client Fuzzer

- Fuzzing mstsc.exe On Windows With WinAFL via Virtual Channels in RDP
- First target : RDPSND
  - A channel enabled by default by mstsc.exe
  - One-way communication as audio playback is run by server and played in the client
  - Very simple protocol
- Note: other channels (Clipboard, etc.) are two-way channels

Applying the RDP Client Fuzzer

- Fuzzing mstsc.exe On Windows With WinAFL
  - RDPSND : Audio Output Virtual Channel



Remote Desktop Connection

Remote Desktop Connection

Remote audio playback

[✓ Play on this computer]

Do not play

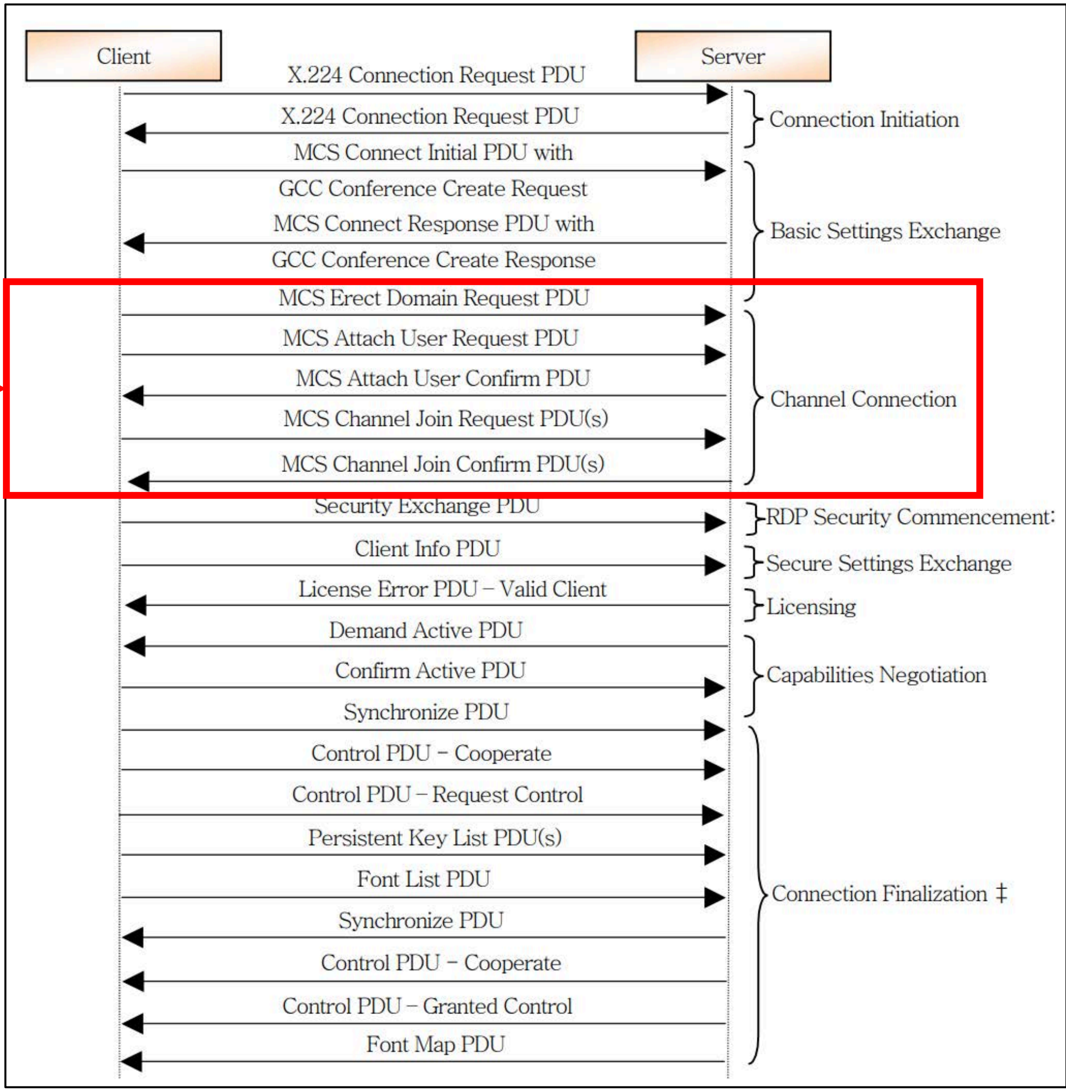
Play on remote computer

Remote audio recording

[✓ Record from this computer]

Do not record

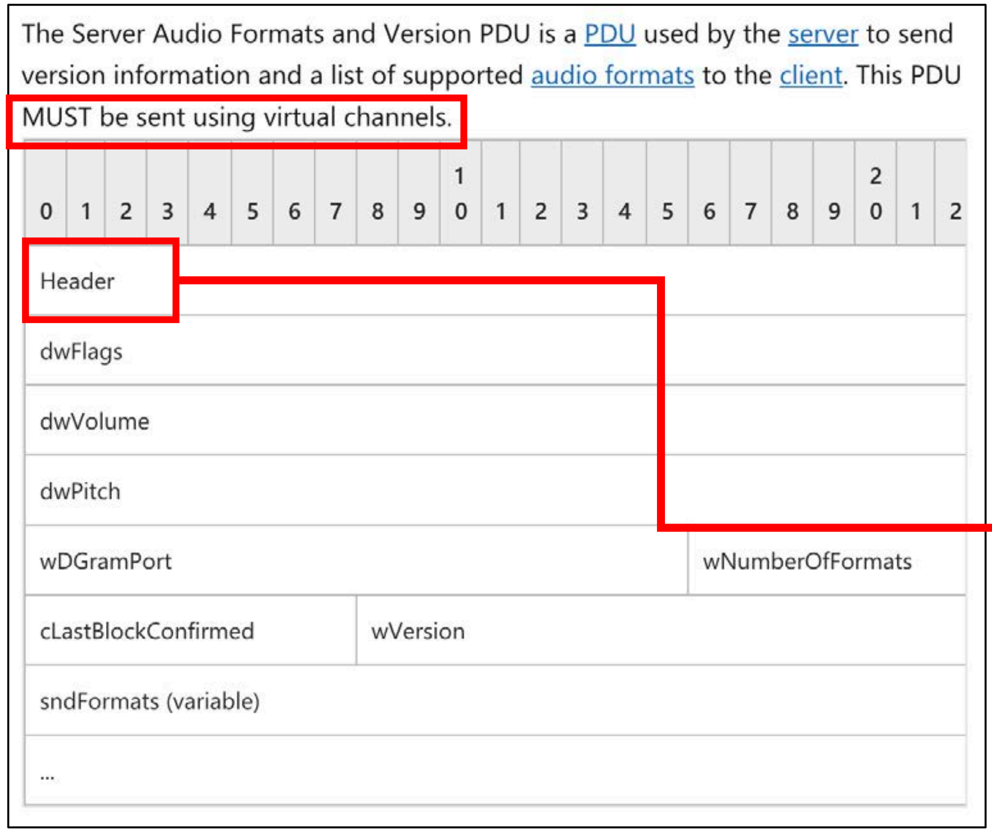
OK Cancel



Client
Server
X.224 Connection Request PDU
X.224 Connection Request PDU
Connection Initiation
MCS Connect Initial PDU with GCC Conference Create Request
MCS Connect Response PDU with GCC Conference Create Response
Basic Settings Exchange
MCS Erect Domain Request PDU
MCS Attach User Request PDU
MCS Attach User Confirm PDU
MCS Channel Join Request PDU(s)
MCS Channel Join Confirm PDU(s)
Channel Connection
Security Exchange PDU
RDP Security Commencement
Client Info PDU
Secure Settings Exchange
License Error PDU – Valid Client
Licensing
Demand Active PDU
Confirm Active PDU
Capabilities Negotiation
Synchronize PDU
Control PDU – Cooperate
Control PDU – Request Control
Persistent Key List PDU(s)
Font List PDU
Synchronize PDU
Control PDU – Cooperate
Control PDU – Granted Control
Font Map PDU
Connection Finalization ‡

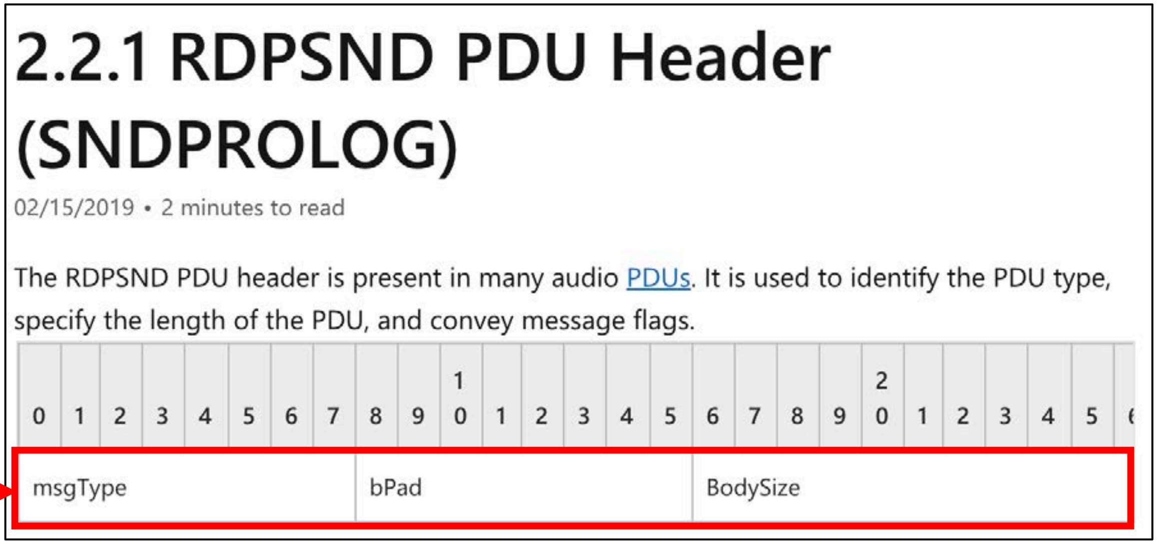
Applying the RDP Client Fuzzer

- Fuzzing mstsc.exe On Windows With WinAFL
  - RDPSND : Audio Output Virtual Channel



The Server Audio Formats and Version PDU is a PDU used by the server to send version information and a list of supported audio formats to the client. This PDU MUST be sent using virtual channels.

0	1	2	3	4	5	6	7	8	9	1 0	1	2	3	4	5	6	7	8	9	2 0	1	2
Header																						
dwFlags																						
dwVolume																						
dwPitch																						
wDGramPort																wNumberOfFormats						
cLastBlockConfirmed								wVersion														
sndFormats (variable)																						
...																						



2.2.1 RDPSND PDU Header (SNDPROLOG)

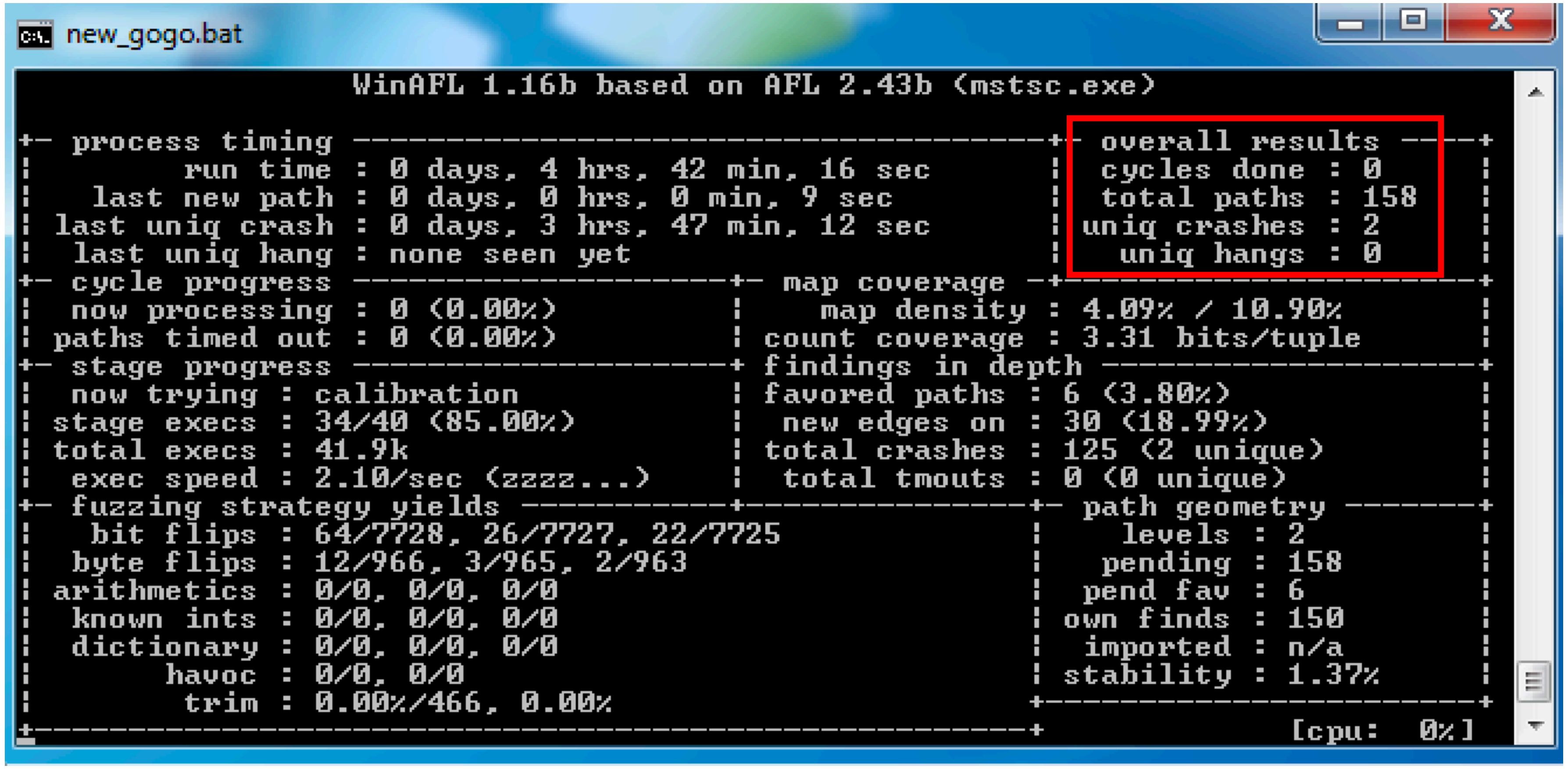
02/15/2019 • 2 minutes to read

The RDPSND PDU header is present in many audio PDUs. It is used to identify the PDU type, specify the length of the PDU, and convey message flags.

0	1	2	3	4	5	6	7	8	9	1 0	1	2	3	4	5	6	7	8	9	2 0	1	2	3	4	5
msgType								bPad								BodySize									

Applying the RDP Client Fuzzer

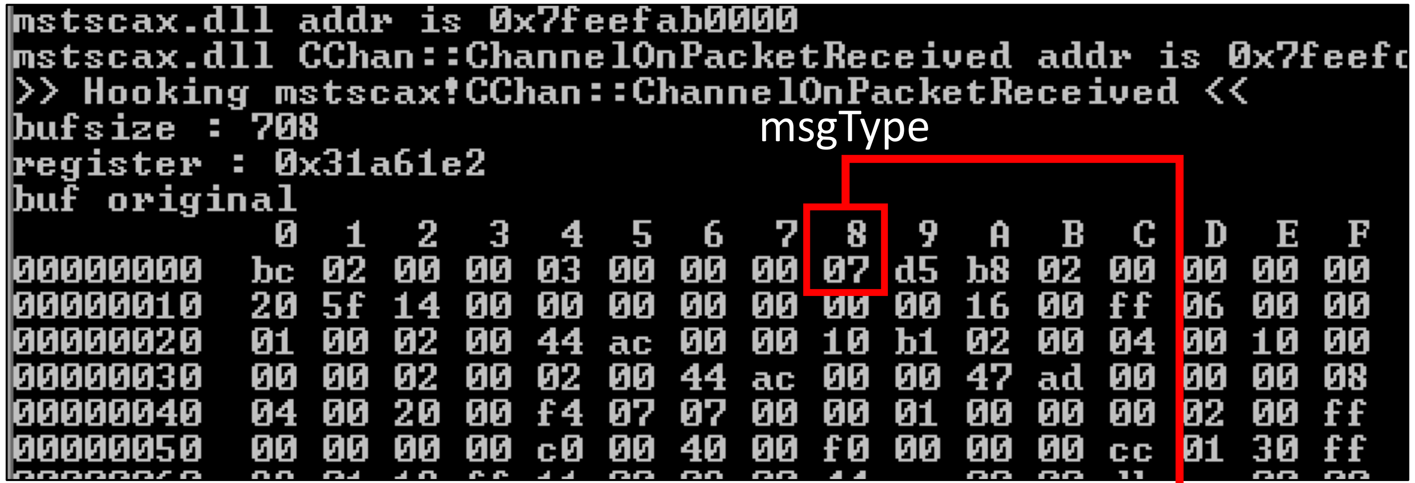
- Fuzzing mstsc.exe On Windows With WinAFL


new_gogo.bat

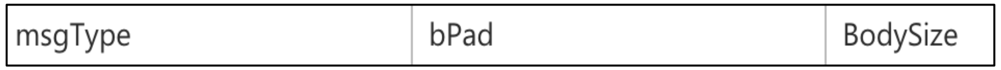
```
                  WinAFL 1.16b based on AFL 2.43b (mstsc.exe)

+- process timing -------------------------------------+- overall results ---+
|        run time : 0 days, 4 hrs, 42 min, 16 sec      |  cycles done : 0    |
|   last new path : 0 days, 0 hrs, 0 min, 9 sec        |  total paths : 158  |
| last uniq crash : 0 days, 3 hrs, 47 min, 12 sec      | uniq crashes : 2    |
|  last uniq hang : none seen yet                      |   uniq hangs : 0    |
+- cycle progress --------------------+- map coverage -+---------------------+
|  now processing : 0 (0.00%)         |    map density : 4.09% / 10.90%      |
| paths timed out : 0 (0.00%)         | count coverage : 3.31 bits/tuple     |
+- stage progress --------------------+ findings in depth -------------------+
|  now trying : calibration           | favored paths : 6 (3.80%)            |
| stage execs : 34/40 (85.00%)        |  new edges on : 30 (18.99%)          |
| total execs : 41.9k                 | total crashes : 125 (2 unique)       |
|  exec speed : 2.10/sec (zzzz...)    |  total tmouts : 0 (0 unique)         |
+- fuzzing strategy yields -----------+---------------+- path geometry ------+
|   bit flips : 64/7728, 26/7727, 22/7725             |    levels : 2        |
|  byte flips : 12/966, 3/965, 2/963                  |   pending : 158      |
| arithmetics : 0/0, 0/0, 0/0                         |  pend fav : 6        |
|  known ints : 0/0, 0/0, 0/0                         | own finds : 150      |
|  dictionary : 0/0, 0/0, 0/0                         |  imported : n/a      |
|       havoc : 0/0, 0/0                              | stability : 1.37%    |
|        trim : 0.00%/466, 0.00%                      +----------------------+
+-----------------------------------------------------+          [cpu:  0%]
```

Applying the RDP Client Fuzzer

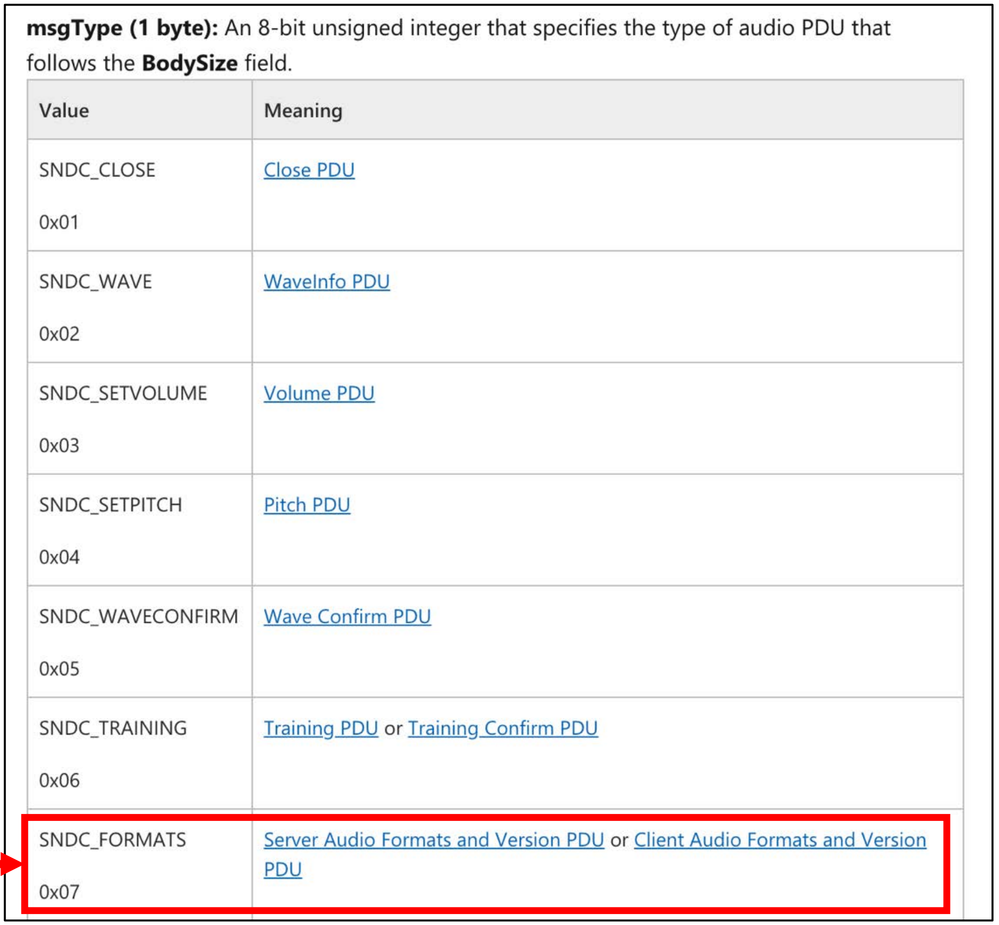
- Fuzzing mstsc.exe On Windows With WinAFL
  - RDPSND : Audio Output Virtual Channel
  - How to get a seed file for fuzzer?
    - Hooking and Logging



```
mstscax.dll addr is 0x7feefab0000
mstscax.dll CChan::ChannelOnPacketReceived addr is 0x7feefc
>> Hooking mstscax!CChan::ChannelOnPacketReceived <<
bufsize : 708
register : 0x31a61e2
buf original
           0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
00000000  bc 02 00 00 03 00 00 00 07 d5 b8 02 00 00 00 00
00000010  20 5f 14 00 00 00 00 00 00 00 16 00 ff 06 00 00
00000020  01 00 02 00 44 ac 00 00 10 b1 02 00 04 00 10 00
00000030  00 00 02 00 02 00 44 ac 00 00 47 ad 00 00 00 08
00000040  04 00 20 00 f4 07 07 00 00 01 00 00 00 02 00 ff
00000050  00 00 00 00 c0 00 40 00 f0 00 00 00 cc 01 30 ff
```



msgType	bPad	BodySize

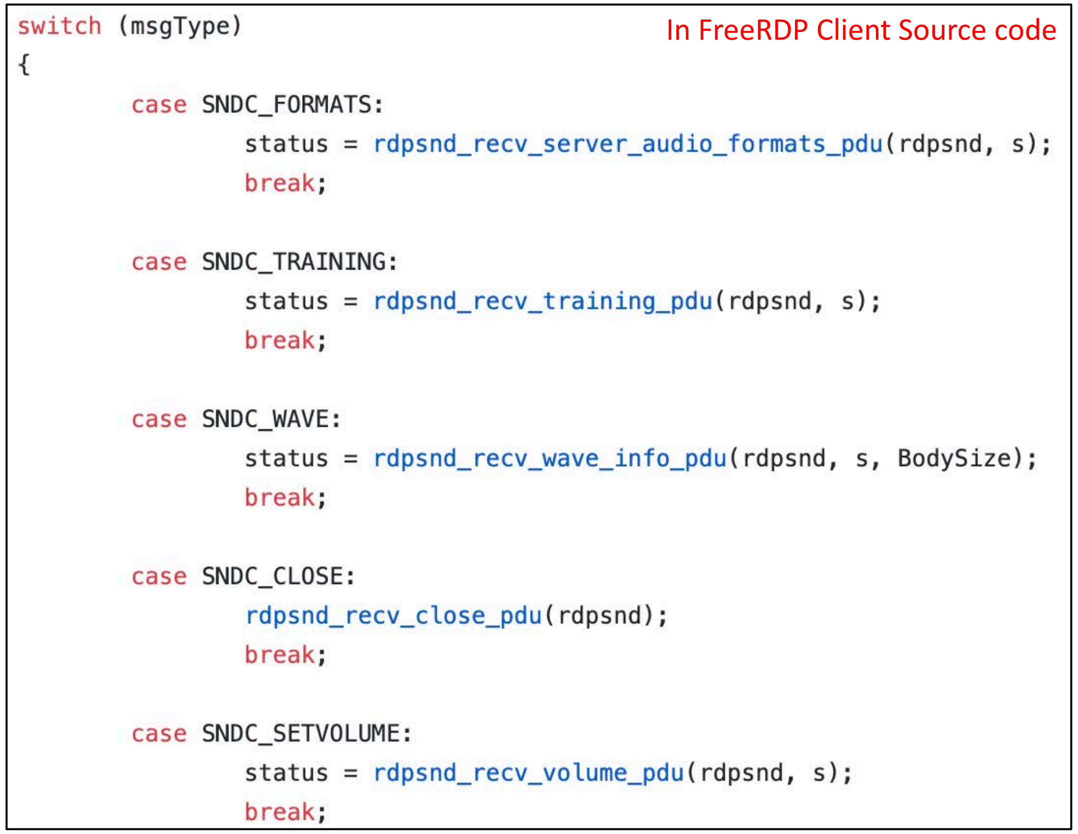


msgType (1 byte): An 8-bit unsigned integer that specifies the type of audio PDU that follows the **BodySize** field.

Value	Meaning
SNDC_CLOSE 0x01	Close PDU
SNDC_WAVE 0x02	WaveInfo PDU
SNDC_SETVOLUME 0x03	Volume PDU
SNDC_SETPITCH 0x04	Pitch PDU
SNDC_WAVECONFIRM 0x05	Wave Confirm PDU
SNDC_TRAINING 0x06	Training PDU or Training Confirm PDU
SNDC_FORMATS 0x07	Server Audio Formats and Version PDU or Client Audio Formats and Version PDU

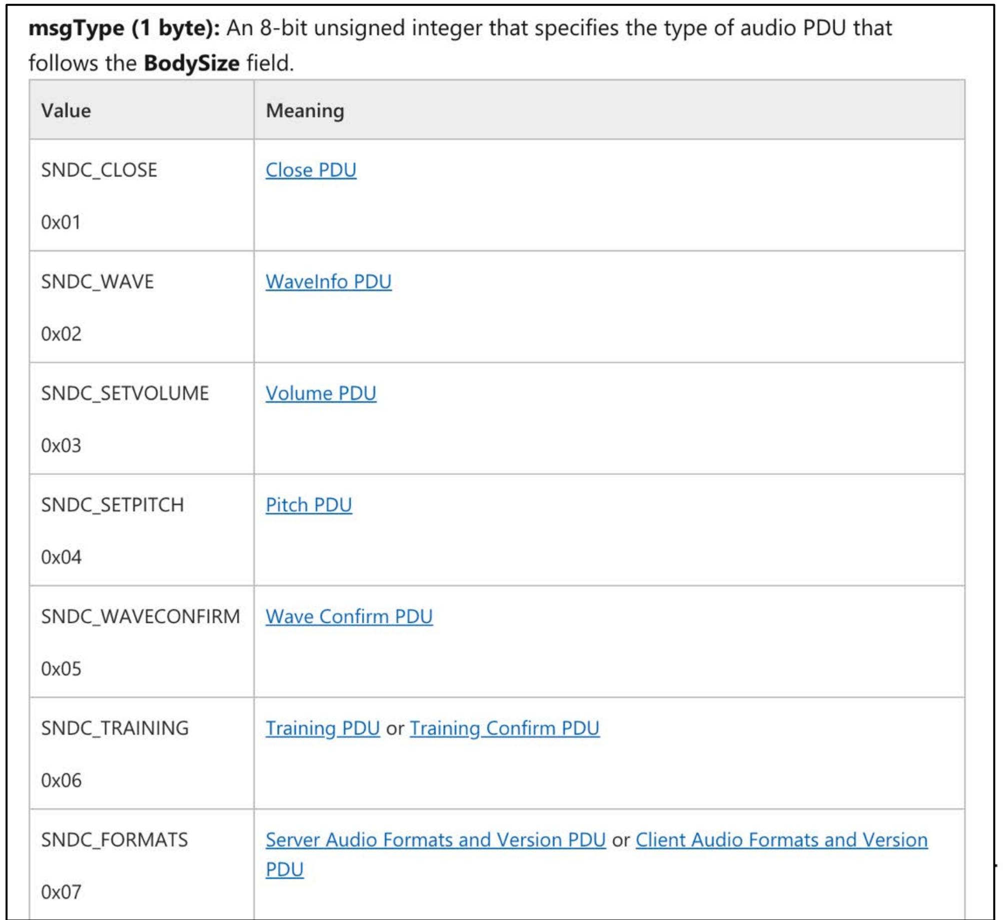
Applying the RDP Client Fuzzer

- Fuzzing mstsc.exe On Windows With WinAFL
  - RDPSND : Audio Output Virtual Channel



In FreeRDP Client Source code

```
switch (msgType)
{
        case SNDC_FORMATS:
            status = rdpsnd_recv_server_audio_formats_pdu(rdpsnd, s);
            break;

        case SNDC_TRAINING:
            status = rdpsnd_recv_training_pdu(rdpsnd, s);
            break;

        case SNDC_WAVE:
            status = rdpsnd_recv_wave_info_pdu(rdpsnd, s, BodySize);
            break;

        case SNDC_CLOSE:
            rdpsnd_recv_close_pdu(rdpsnd);
            break;

        case SNDC_SETVOLUME:
            status = rdpsnd_recv_volume_pdu(rdpsnd, s);
            break;
```

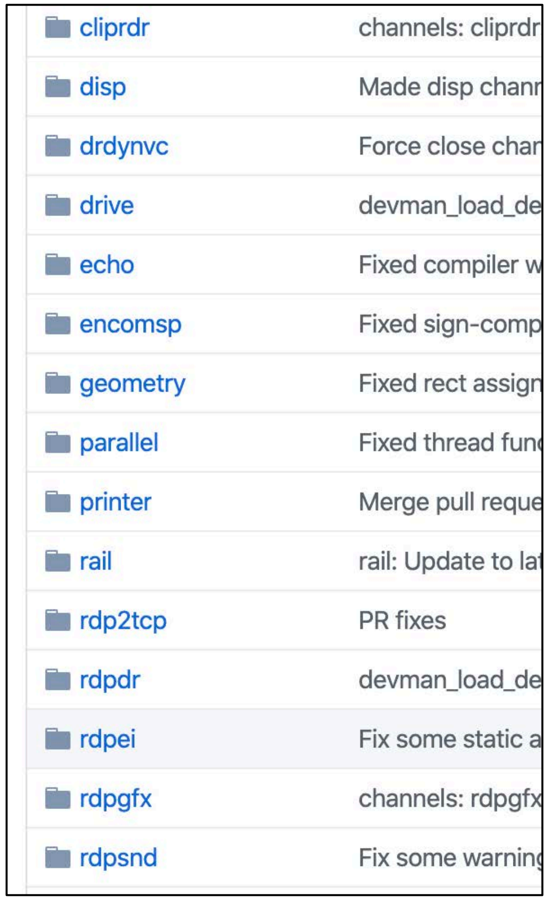


msgType (1 byte): An 8-bit unsigned integer that specifies the type of audio PDU that follows the **BodySize** field.

Value	Meaning
SNDC_CLOSE 0x01	Close PDU
SNDC_WAVE 0x02	WaveInfo PDU
SNDC_SETVOLUME 0x03	Volume PDU
SNDC_SETPITCH 0x04	Pitch PDU
SNDC_WAVECONFIRM 0x05	Wave Confirm PDU
SNDC_TRAINING 0x06	Training PDU or Training Confirm PDU
SNDC_FORMATS 0x07	Server Audio Formats and Version PDU or Client Audio Formats and Version PDU

Applying the RDP Client Fuzzer

- Fuzzing mstsc.exe On Windows With WinAFL
- Next targets
  - Other Channels
    - Drdynvc
    - Printer
    - Smartcard
    - …

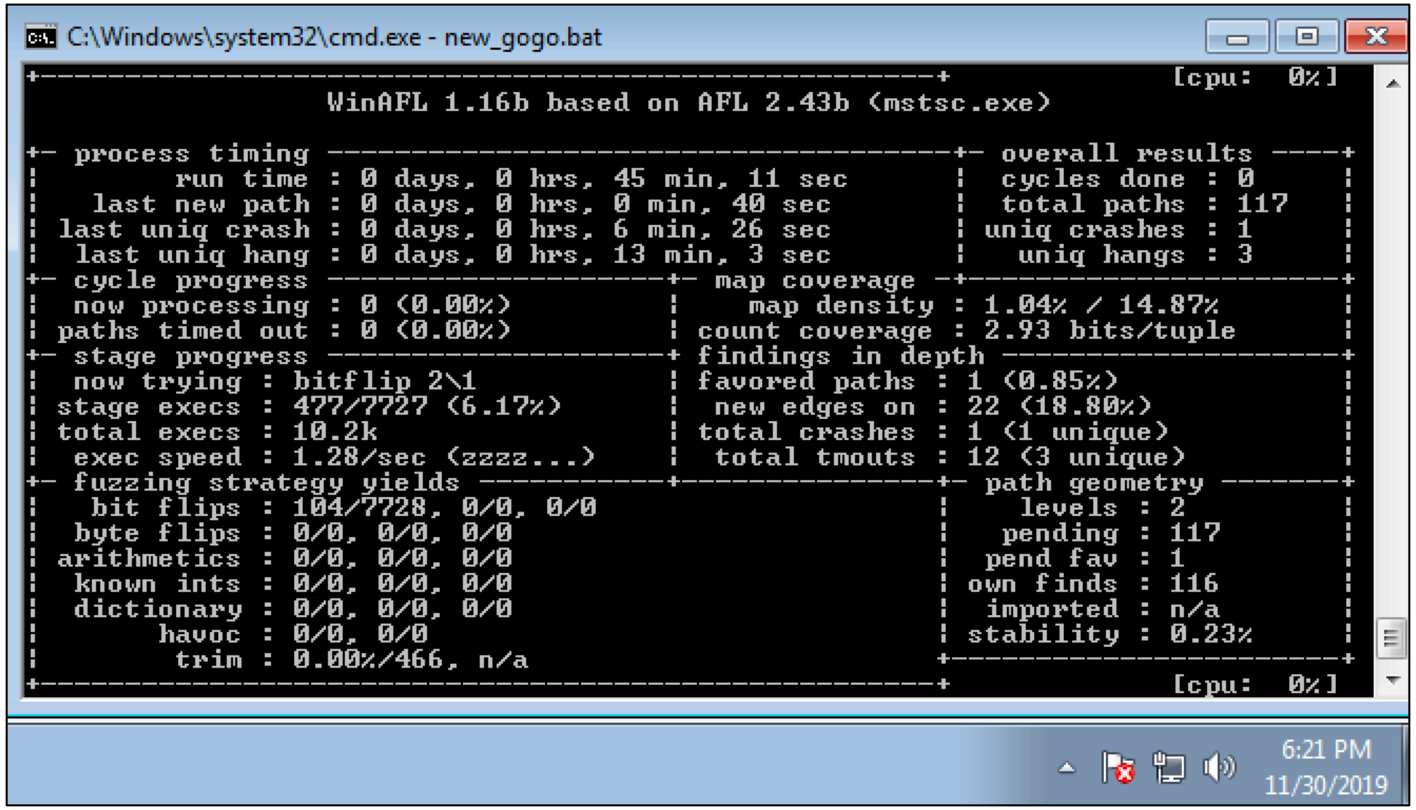


cliprdr	channels: cliprdr
disp	Made disp chann
drdynvc	Force close char
drive	devman_load_de
echo	Fixed compiler w
encomsp	Fixed sign-comp
geometry	Fixed rect assign
parallel	Fixed thread fun
printer	Merge pull reque
rail	rail: Update to la
rdp2tcp	PR fixes
rdpdr	devman_load_de
rdpei	Fix some static a
rdpgfx	channels: rdpgfx
rdpsnd	Fix some warning

In FreeRDP Client Source code

Applying the RDP Client Fuzzer

- Demonstration : RDP Client Fuzzer
  - Running on Windows 7
  - 1 Core
  - 2G Memory
  - 2 VMs are required
    - One for RDP Client
    - One for RDP Server



C:\Windows\system32\cmd.exe - new_gogo.bat

```
+--------------------------------------------------------+   [cpu:  0%]
                WinAFL 1.16b based on AFL 2.43b (mstsc.exe)

+- process timing -------------------------------------+- overall results ----+
|        run time : 0 days, 0 hrs, 45 min, 11 sec      |  cycles done : 0     |
|   last new path : 0 days, 0 hrs, 0 min, 40 sec       |  total paths : 117   |
| last uniq crash : 0 days, 0 hrs, 6 min, 26 sec       | uniq crashes : 1     |
|  last uniq hang : 0 days, 0 hrs, 13 min, 3 sec       |   uniq hangs : 3     |
+- cycle progress --------------------+- map coverage -+----------------------+
|  now processing : 0 (0.00%)         |    map density : 1.04% / 14.87%       |
| paths timed out : 0 (0.00%)         | count coverage : 2.93 bits/tuple      |
+- stage progress --------------------+- findings in depth -------------------+
|  now trying : bitflip 2\1           | favored paths : 1 (0.85%)             |
| stage execs : 477/7727 (6.17%)      |  new edges on : 22 (18.80%)           |
| total execs : 10.2k                 | total crashes : 1 (1 unique)          |
|  exec speed : 1.28/sec (zzzz...)    |  total tmouts : 12 (3 unique)         |
+- fuzzing strategy yields -----------+---------------+- path geometry -------+
|   bit flips : 104/7728, 0/0, 0/0                    |    levels : 2         |
|  byte flips : 0/0, 0/0, 0/0                         |   pending : 117       |
| arithmetics : 0/0, 0/0, 0/0                         |  pend fav : 1         |
|  known ints : 0/0, 0/0, 0/0                         | own finds : 116       |
|  dictionary : 0/0, 0/0, 0/0                         |  imported : n/a       |
|       havoc : 0/0, 0/0                              | stability : 0.23%     |
|        trim : 0.00%/466, n/a                        +-----------------------+
+-----------------------------------------------------+        [cpu:  0%]
```

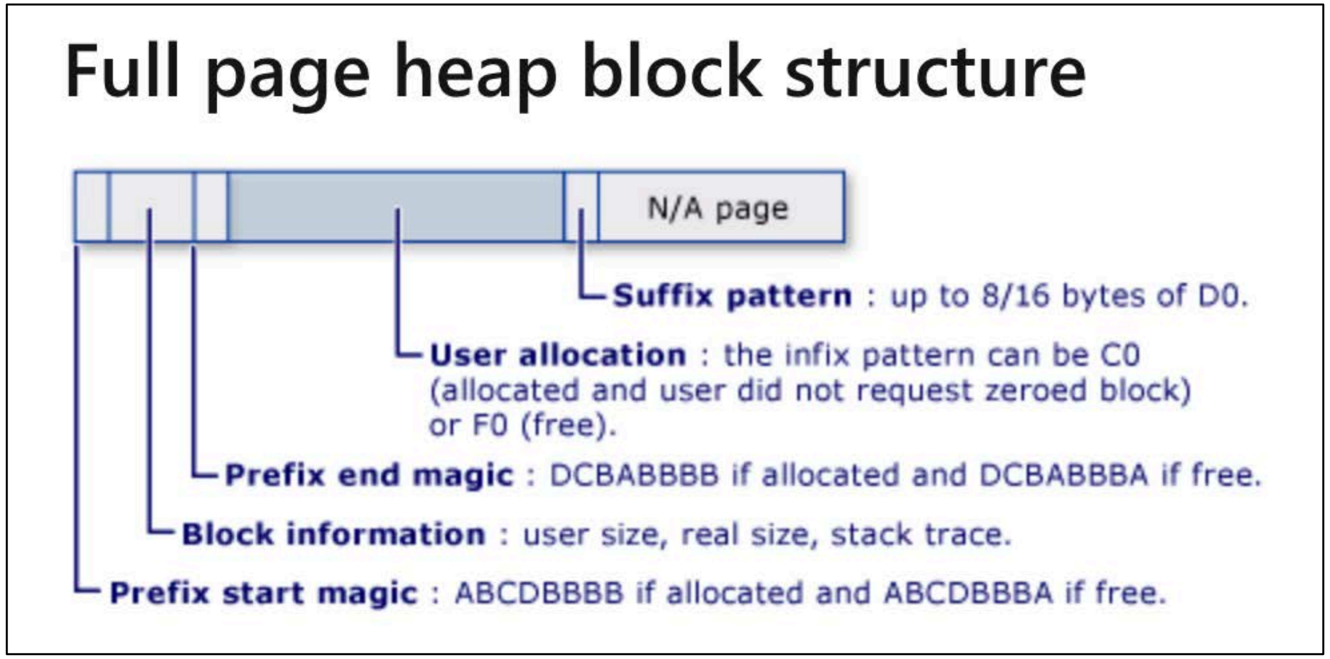
6:21 PM
11/30/2019

Running the RDP Client Fuzzer

- Fuzzing setup
  - Both the RDP Client and the RDP Server runs in each VM
  - Running on Windows 7
  - 1 Core
  - 2G Memory
  - 2 VMs are required
    - One for RDP Client
    - One for RDP Server
  - The first vulnerability was found within 2 hours of running the RDP Client Fuzzer!
    - We keep reporting vulnerabilities to MS

Detecting Heap Vulnerabilities

- We may use PageHeap; allocate each heap object in a new page
  - `gflags.exe /p /enable mstsc.exe /full`
  - Slow, but this will generate crash if any heap error happens during fuzzing..



Full page heap block structure

N/A page
Suffix pattern : up to 8/16 bytes of D0.
User allocation : the infix pattern can be C0 (allocated and user did not request zeroed block) or F0 (free).
Prefix end magic : DCBABBBB if allocated and DCBABBBA if free.
Block information : user size, real size, stack trace.
Prefix start magic : ABCDBBBB if allocated and ABCDBBBA if free.

MSDN Article: The Structure of a Page Heap Block

VULNERABILITY & EXPLOIT

- RDP Client vulnerabilities on Windows in 2019

Category	Title	Author	Date	Ref	Threat No
CVE-2019-1333	Remote Desktop Client Remote Code Execution Vulnerability	Yongil Lee of Diffense	2019.10.08	https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-1333	T1
CVE-2019-1291	Remote Desktop Client Remote Code Execution Vulnerability	Microsoft Platform Security Assurance & Vulnerability Research	2019.09.10	https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-1291	T2
CVE-2019-1290	Remote Desktop Client Remote Code Execution Vulnerability	Microsoft Platform Security Assurance & Vulnerability Research	2019.09.10	https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-1290	T3
CVE-2019-0788	Remote Desktop Client Remote Code Execution Vulnerability	Microsoft Platform Security Assurance & Vulnerability Research	2019.09.10	https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-0788	T4
CVE-2019-0787	Remote Desktop Client Remote Code Execution Vulnerability	Microsoft Platform Security Assurance & Vulnerability Research	2019.09.10	https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-0787	T5
CVE-2019-1108	Remote Desktop Protocol Client Information Disclosure Vulnerability	RDP_HACKER	2019.07.09	https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-1108	T6
CVE-2019-0887	Remote Desktop Client Remote Code Execution Vulnerability	Eyal Itkin of Check Point Research	2019.07.09	https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-0887	T7

Report by Checkpoint

VULNERABILITY & EXPLOIT

- RDP Client vulnerabilities on Windows in 2019

Category	Title	Author	Date	Ref	Threat No
CVE-2019-1333	Remote Desktop Client Remote Code Execution Vulnerability	Yongil Lee of Diffense	2019.10.08	https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-1333	T1
CVE-2019-1291	Remote Desktop Client Remote Code Execution Vulnerability	Microsoft Platform Security Assurance & Vulnerability Research	2019.09.10	https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-1291	T2
CVE-2019-1290	Remote Desktop Client Remote Code Execution Vulnerability	Microsoft Platform Security Assurance & Vulnerability Research	2019.09.10	https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-1290	T3
CVE-2019-0788	Remote Desktop Client Remote Code Execution Vulnerability	Microsoft Platform Security Assurance & Vulnerability Research	2019.09.10	https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-0788	T4
CVE-2019-0787	Remote Desktop Client Remote Code Execution Vulnerability	Microsoft Platform Security Assurance & Vulnerability Research	2019.09.10	https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-0787	T5
CVE-2019-1108	Remote Desktop Protocol Client Information Disclosure Vulnerability	RDP_HACKER	2019.07.09	https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-1108	T6
CVE-2019-0887	Remote Desktop Client Remote Code Execution Vulnerability	Eyal Itkin of Check Point Research	2019.07.09	https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-0887	T7

No comment on: CVE-2019-1108; guess who reported this

VULNERABILITY & EXPLOIT

- RDP Client vulnerabilities on Windows in 2019

Category	Title	Author	Date	Ref	Threat No
CVE-2019-1333	Remote Desktop Client Remote Code Execution Vulnerability	Yongil Lee of Diffense	2019.10.08	https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-1333	T1
CVE-2019-1291	Remote Desktop Client Remote Code Execution Vulnerability	Microsoft Platform Security Assurance & Vulnerability Research	2019.09.10	https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-1291 ??? MSRC	T2
CVE-2019-1290	Remote Desktop Client Remote Code Execution Vulnerability	Microsoft Platform Security Assurance & Vulnerability Research	2019.09.10	https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-1290	T3
CVE-2019-0788	Remote Desktop Client Remote Code Execution Vulnerability	Microsoft Platform Security Assurance & Vulnerability Research	2019.09.10	https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-0788	T4
CVE-2019-0787	Remote Desktop Client Remote Code Execution Vulnerability	Microsoft Platform Security Assurance & Vulnerability Research	2019.09.10	https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-0787	T5
CVE-2019-1108	Remote Desktop Protocol Client Information Disclosure Vulnerability	RDP_HACKER	2019.07.09	https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-1108	T6
CVE-2019-0887	Remote Desktop Client Remote Code Execution Vulnerability	Eyal Itkin of Check Point Research	2019.07.09	https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-0887	T7

VULNERABILITY & EXPLOIT

- RDP Client vulnerabilities on Windows in 2019

He is my boss

Category	Title	Author	Date	Ref	Threat No
CVE-2019-1333	Remote Desktop Client Remote Code Execution Vulnerability	Yongil Lee of Diffense	2019.10.08	https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-1333	T1
CVE-2019-1291	Remote Desktop Client Remote Code Execution Vulnerability	Microsoft Platform Security Assurance & Vulnerability Research	2019.09.10	https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-1291	T2
CVE-2019-1290	Remote Desktop Client Remote Code Execution Vulnerability	Microsoft Platform Security Assurance & Vulnerability Research	2019.09.10	https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-1290	T3
CVE-2019-0788	Remote Desktop Client Remote Code Execution Vulnerability	Microsoft Platform Security Assurance & Vulnerability Research	2019.09.10	https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-0788	T4
CVE-2019-0787	Remote Desktop Client Remote Code Execution Vulnerability	Microsoft Platform Security Assurance & Vulnerability Research	2019.09.10	https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-0787	T5
CVE-2019-1108	Remote Desktop Protocol Client Information Disclosure Vulnerability	RDP_HACKER	2019.07.09	https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-1108	T6
CVE-2019-0887	Remote Desktop Client Remote Code Execution Vulnerability	Eyal Itkin of Check Point Research	2019.07.09	https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-0887	T7

VULNERABILITY & EXPLOIT

- RDP Client vulnerabilities on Windows in 2019

He is my boss

Category	Title	Author	Date	Ref	Threat No
CVE-2019-1333	Remote Desktop Client Remote Code Execution Vulnerability	Yongil Lee of Diffense	2019.10.08	https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-1333	T1
CVE-2019-1291	Remote Desktop Client Remote [illegible]	Microsoft Platform Security [illegible]	[illegible]	https://portal.msrc.microsoft.com/en-US/security- [illegible] 1	T2
CVE-2019-1290	[illegible]	[illegible]	[illegible]	/security- [illegible] 90	T3
CVE-2019-0788	[illegible]	[illegible]	[illegible]	/security- [illegible] 88	T4
CVE-2019-0787	[illegible]	[illegible] Research	[illegible]	/security- [illegible] 87	T5
CVE-2019-1108	Remote Desktop Protocol Client Information Disclosure Vulnerability	RDP_HACKER	2019.07.09	https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-1108	T6
CVE-2019-0887	Remote Desktop Client Remote Code Execution Vulnerability	Eyal Itkin of Check Point Research	2019.07.09	https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2019-0887	T7

For the next, we will describe how to exploit the vulnerability found by our RDP Client Fuzzer, by combining information leak and remote code execution vulnerabilities…

No comment on: CVE-2019-1108; guess who reported this

Vulnerabilities and Exploit

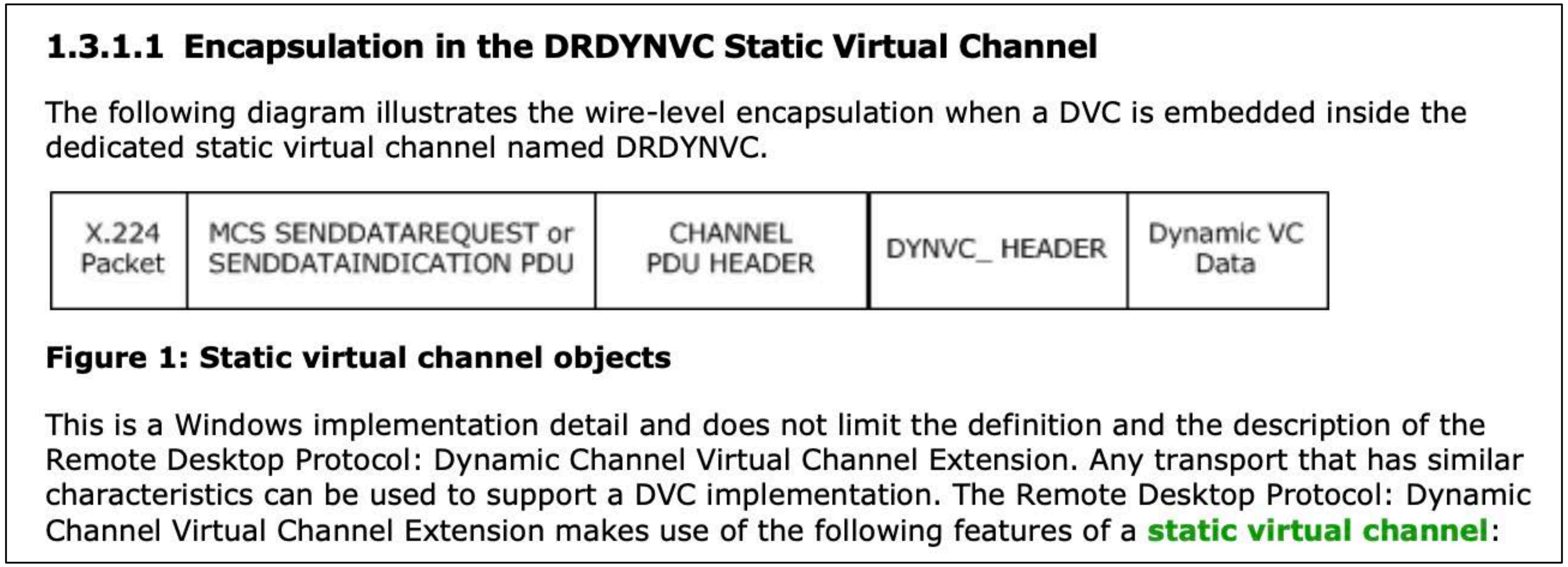
- We will break the ASLR in the RDP client

- We will achieve RCE in the RDP client

- The RCE vulnerability is a heap vulnerability
    - We need to manipulate heap objects
        - Need **heap feng shui**

RDP Heap Feng Shui

- Exploiting a DRDYNVC Channel
- DRDYNVC is a dedicated channel for delivering dynamic data



1.3.1.1 Encapsulation in the DRDYNVC Static Virtual Channel

The following diagram illustrates the wire-level encapsulation when a DVC is embedded inside the dedicated static virtual channel named DRDYNVC.

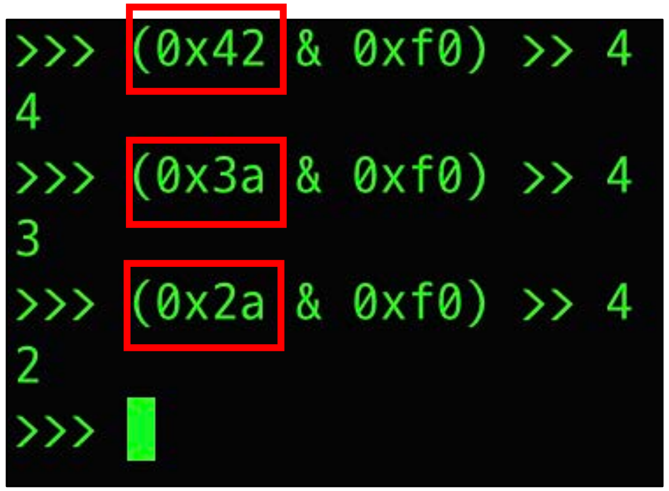
X.224 Packet	MCS SENDDATAREQUEST or SENDDATAINDICATION PDU	CHANNEL PDU HEADER	DYNVC_ HEADER	Dynamic VC Data

Figure 1: Static virtual channel objects

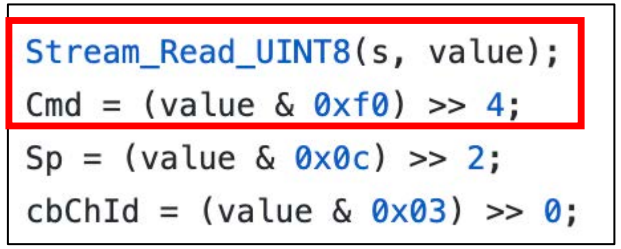
This is a Windows implementation detail and does not limit the definition and the description of the Remote Desktop Protocol: Dynamic Channel Virtual Channel Extension. Any transport that has similar characteristics can be used to support a DVC implementation. The Remote Desktop Protocol: Dynamic Channel Virtual Channel Extension makes use of the following features of a **static virtual channel**:

RDP Heap Feng Shui

- In the DRDYNVC Channel

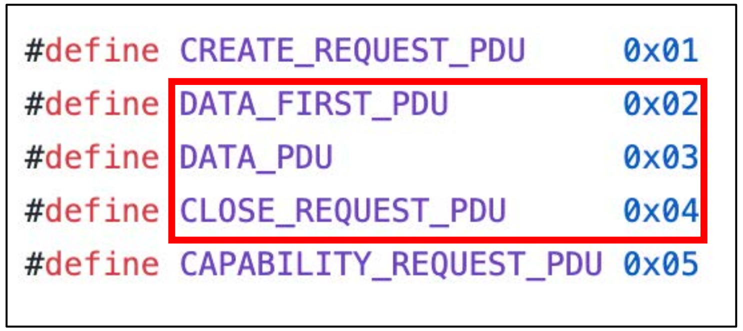


```
>>> (0x42 & 0xf0) >> 4
4
>>> (0x3a & 0xf0) >> 4
3
>>> (0x2a & 0xf0) >> 4
2
>>> 
```

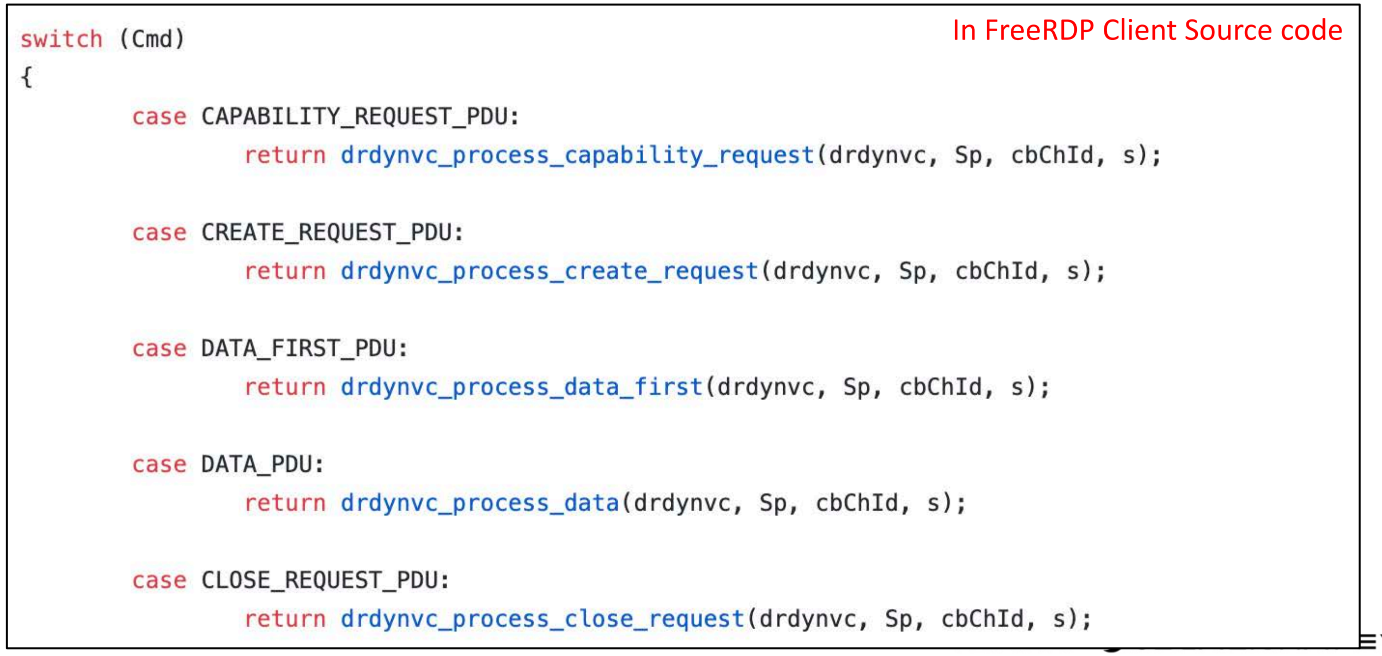


```
Stream_Read_UINT8(s, value);
Cmd = (value & 0xf0) >> 4;
Sp = (value & 0x0c) >> 2;
cbChId = (value & 0x03) >> 0;
```

...



```
#define CREATE_REQUEST_PDU      0x01
#define DATA_FIRST_PDU          0x02
#define DATA_PDU                0x03
#define CLOSE_REQUEST_PDU       0x04
#define CAPABILITY_REQUEST_PDU  0x05
```

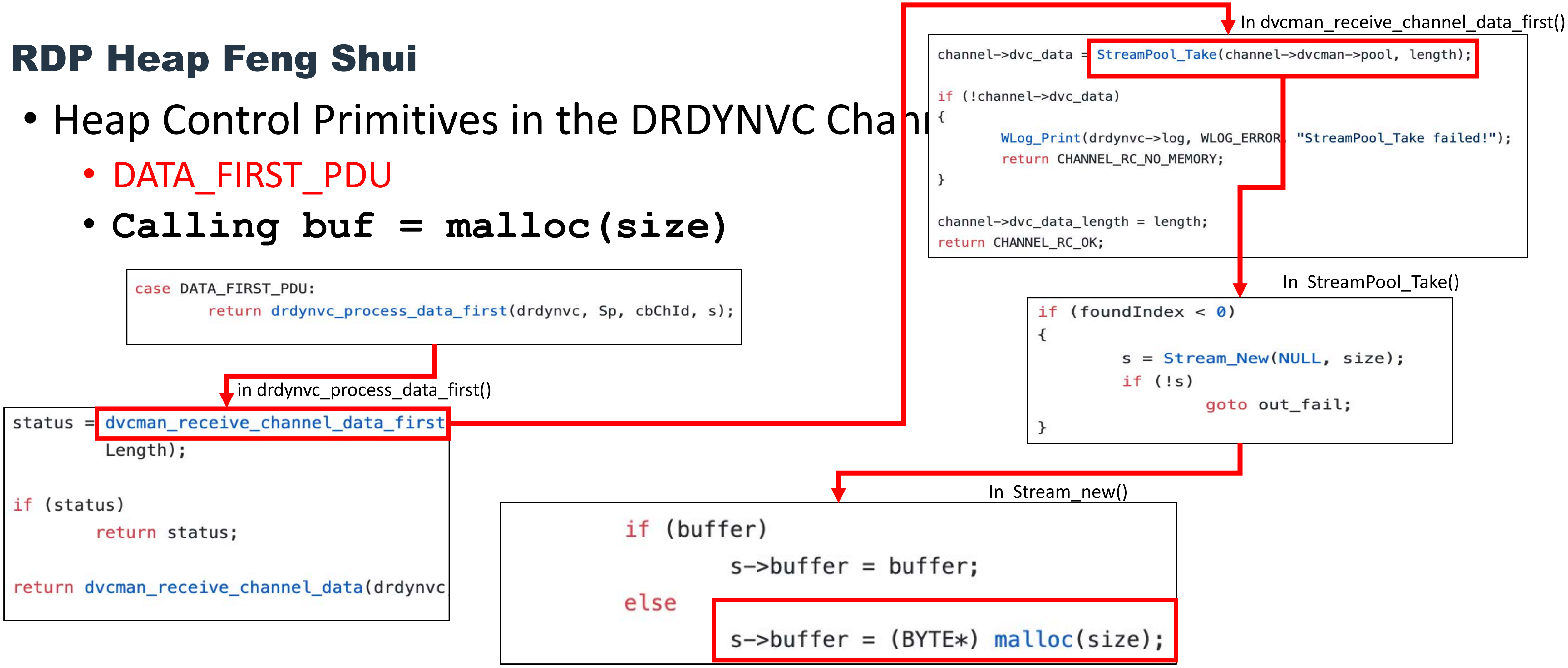


In FreeRDP Client Source code

```
switch (Cmd)
{
        case CAPABILITY_REQUEST_PDU:
                return drdynvc_process_capability_request(drdynvc, Sp, cbChId, s);

        case CREATE_REQUEST_PDU:
                return drdynvc_process_create_request(drdynvc, Sp, cbChId, s);

        case DATA_FIRST_PDU:
                return drdynvc_process_data_first(drdynvc, Sp, cbChId, s);

        case DATA_PDU:
                return drdynvc_process_data(drdynvc, Sp, cbChId, s);

        case CLOSE_REQUEST_PDU:
                return drdynvc_process_close_request(drdynvc, Sp, cbChId, s);
```

RDP Heap Feng Shui

- Heap Control Primitives in the DRDYNVC Chan
  - DATA_FIRST_PDU
  - **`Calling buf = malloc(size)`**



```
case DATA_FIRST_PDU:
        return drdynvc_process_data_first(drdynvc, Sp, cbChId, s);
```

in drdynvc_process_data_first()

```
status = dvcman_receive_channel_data_first
         Length);

if (status)
        return status;

return dvcman_receive_channel_data(drdynvc
```

In dvcman_receive_channel_data_first()

```
channel->dvc_data = StreamPool_Take(channel->dvcman->pool, length);

if (!channel->dvc_data)
{
        WLog_Print(drdynvc->log, WLOG_ERROR, "StreamPool_Take failed!");
        return CHANNEL_RC_NO_MEMORY;
}

channel->dvc_data_length = length;
return CHANNEL_RC_OK;
```

In StreamPool_Take()

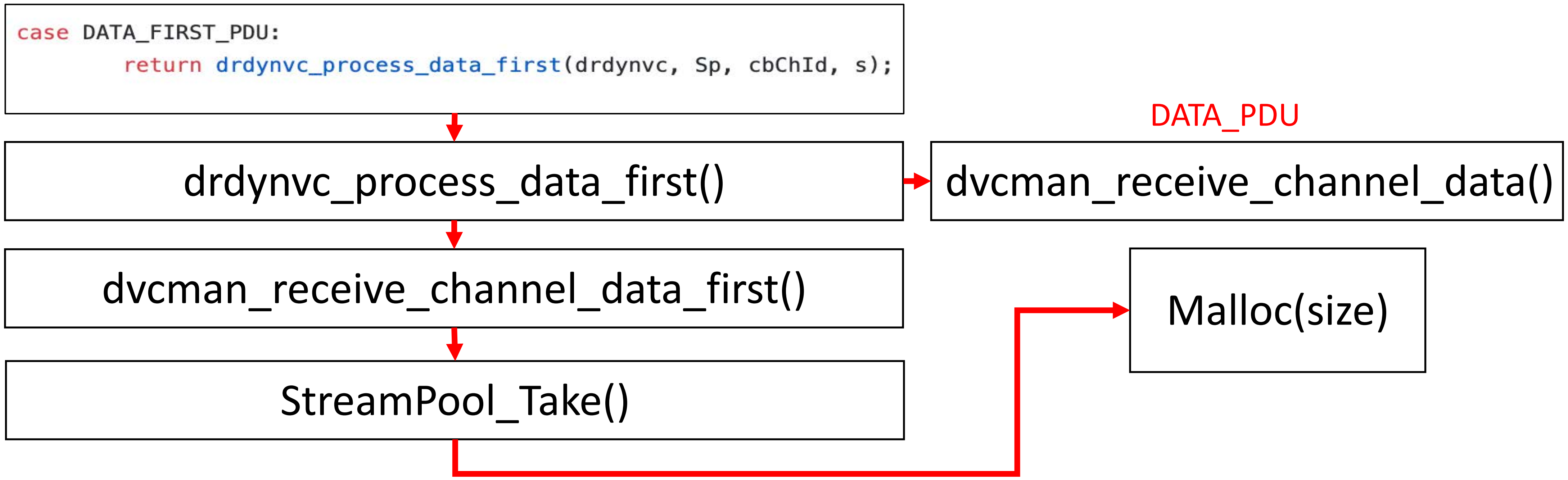
```
if (foundIndex < 0)
{
        s = Stream_New(NULL, size);
        if (!s)
                goto out_fail;
}
```

In Stream_new()

```
if (buffer)
        s->buffer = buffer;
else
        s->buffer = (BYTE*) malloc(size);
```

RDP Heap Feng Shui

- Heap Control Primitives in the DRDYNVC Channel
  - DATA_FIRST_PDU
  - **`Calling buf = malloc(size)`**

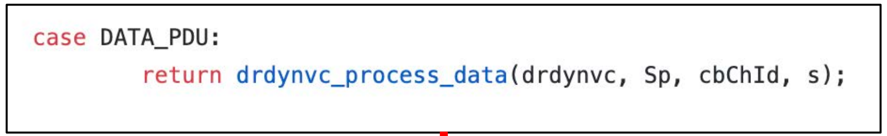


```
case DATA_FIRST_PDU:
        return drdynvc_process_data_first(drdynvc, Sp, cbChId, s);
```

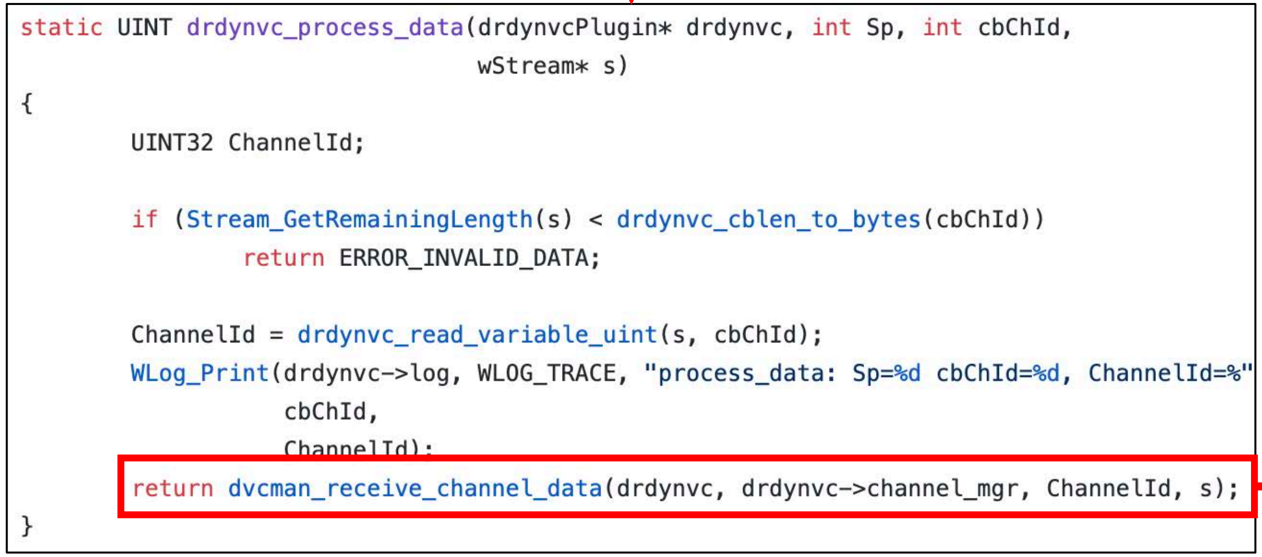
drdynvc_process_data_first()
dvcman_receive_channel_data_first()
StreamPool_Take()
DATA_PDU
dvcman_receive_channel_data()
Malloc(size)

RDP Heap Feng Shui

- Heap Control Primitives in the DRDYNVC Channel
  - DATA_PDU
  - **Calling memcpy(buf,input,size)**

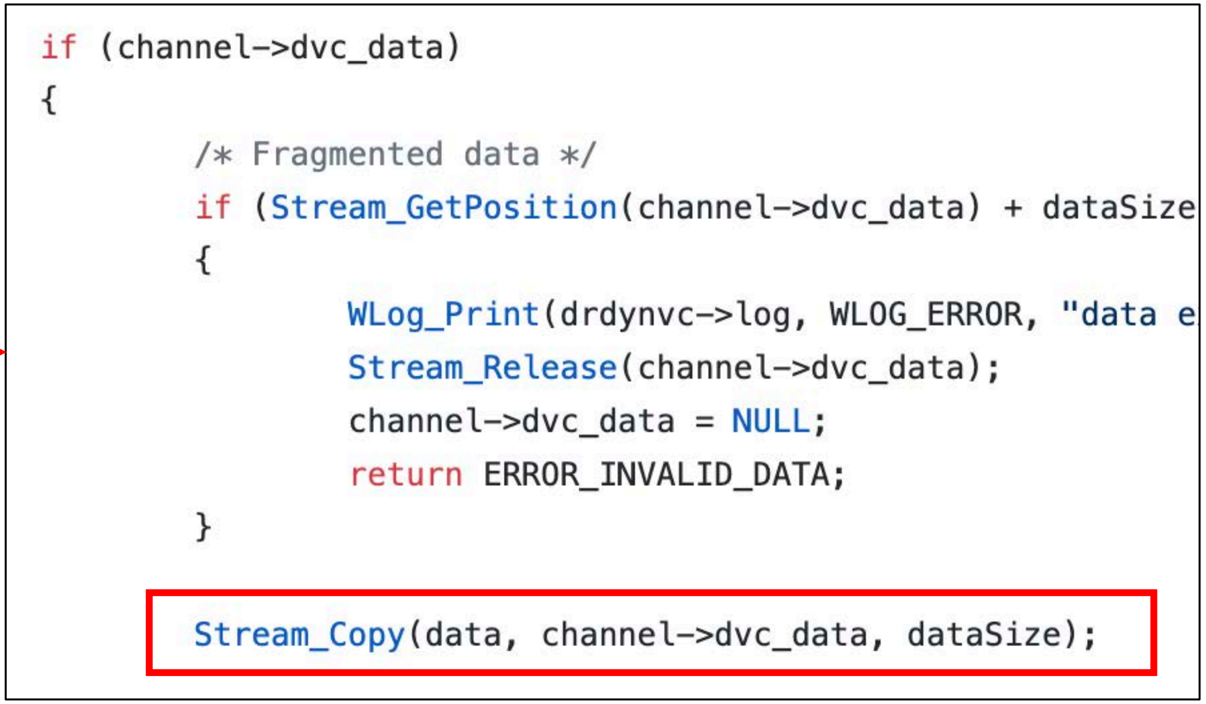


```
case DATA_PDU:
        return drdynvc_process_data(drdynvc, Sp, cbChId, s);
```



```
static UINT drdynvc_process_data(drdynvcPlugin* drdynvc, int Sp, int cbChId,
                                 wStream* s)
{
        UINT32 ChannelId;

        if (Stream_GetRemainingLength(s) < drdynvc_cblen_to_bytes(cbChId))
                return ERROR_INVALID_DATA;

        ChannelId = drdynvc_read_variable_uint(s, cbChId);
        WLog_Print(drdynvc->log, WLOG_TRACE, "process_data: Sp=%d cbChId=%d, ChannelId=%"
                   cbChId,
                   ChannelId);
        return dvcman_receive_channel_data(drdynvc, drdynvc->channel_mgr, ChannelId, s);
}
```

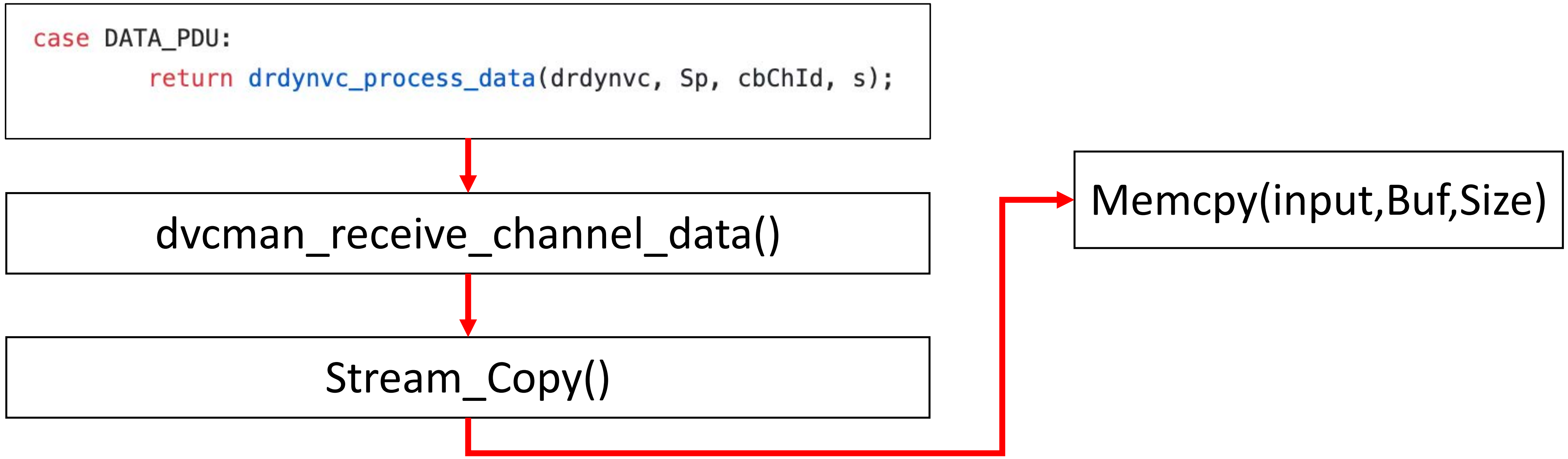
In dvcman_receive_channel_data()



```
if (channel->dvc_data)
{
        /* Fragmented data */
        if (Stream_GetPosition(channel->dvc_data) + dataSize
        {
                WLog_Print(drdynvc->log, WLOG_ERROR, "data e
                Stream_Release(channel->dvc_data);
                channel->dvc_data = NULL;
                return ERROR_INVALID_DATA;
        }

        Stream_Copy(data, channel->dvc_data, dataSize);
```

RDP Heap Feng Shui

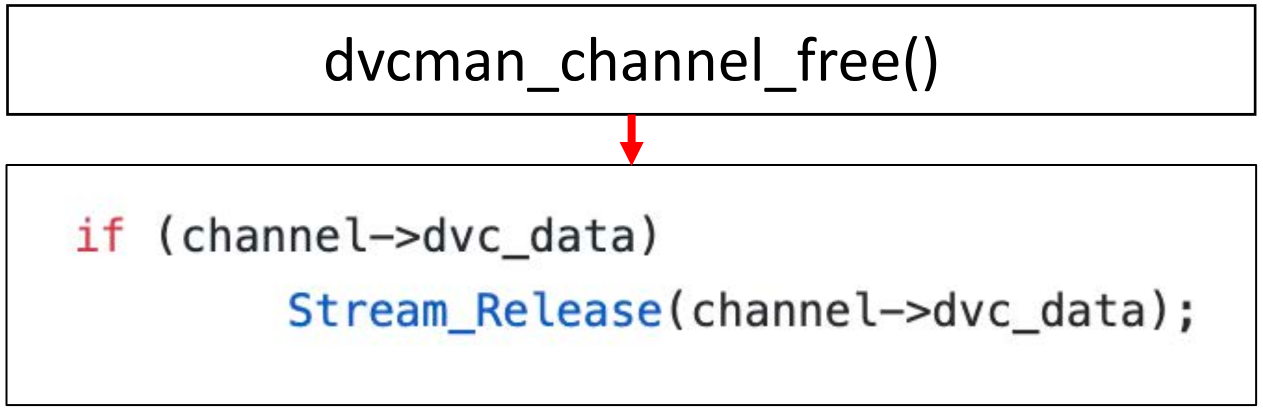
- Heap Control Primitives in the DRDYNVC Channel
  - DATA_PDU
  - **Calling memcpy(buf,input,size)**



```
case DATA_PDU:
        return drdynvc_process_data(drdynvc, Sp, cbChId, s);
```

RDP Heap Feng Shui

- Heap Control Primitives in the DRDYNVC Channel
  - Close Channel
  - **`Calling free(buf)`**



dvcman_channel_free()

```
if (channel->dvc_data)
        Stream_Release(channel->dvc_data);
```

Exploitation

- Combining Information Leak & RCE vulnerabilities
- This part has been removed due to the restriction of one of our speakers
  - NDA…

- But, after having an arbitrary control over malloc(), memcpy(), and free()
  - We can do feng shui to spray the data and manipulate heap objects to achieve arbitrary code execution
  - Fairly standard after following RDP Heap Feng Shui primitives

Please do not be disappointed, we will show you a demo of the full-chain RCE exploit!

DEMO

- Exploiting an Information Leak vulnerability (guess which CVE it is!)
- Use of Uninitialized memory; affected to:
  - Mstsc.exe
  - Windows 10 app
  - Android App
  - iOS App
  - macOS App
- All five RDP clients (MS official)!

Our assumption on sharing the code base among those seems true! Fuzz mstsc.exe and pwn four more!

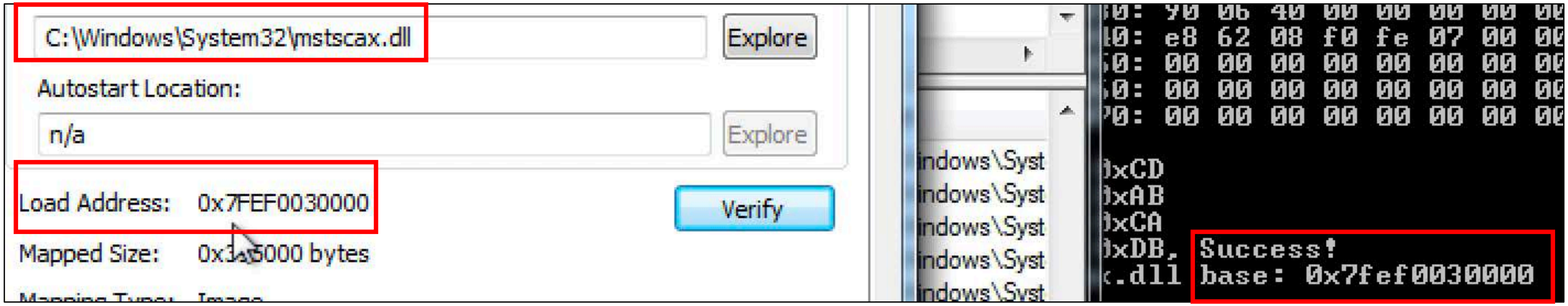
DEMO

- Exploiting an Information Leak vulnerability (guess which CVE it is!)
- Use of Uninitialized memory; affected to:
  - Mstsc.exe
  - Windows 10 app
  - Android App
  - iOS App
  - macOS App
- All five RDP clients (MS official)!

Our assumption on sharing the code base among those seems true! Fuzz mstsc.exe and pwn four more!

DEMO

- Breaking ASLR (guess which CVE it is!)
  - Mstsc.exe on windows 7
  - We will demonstrate how we can break the ASLR of the client machine by launching a memory leak attack.



C:\Windows\System32\mstscax.dll

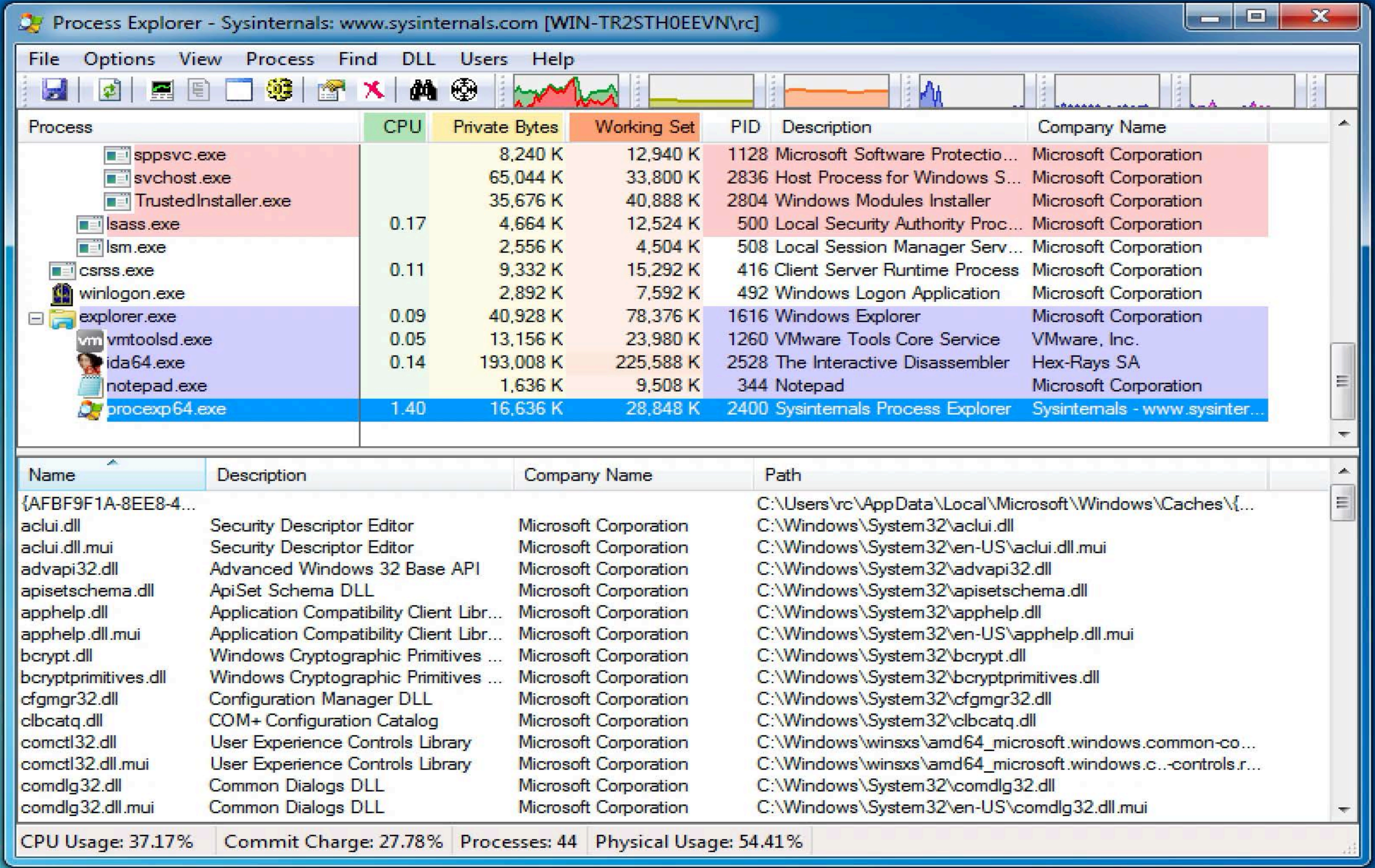
Autostart Location: n/a

Load Address: 0x7FEF0030000

Mapped Size: 0x1?5000 bytes

```
indows\Syst
indows\Syst
indows\Syst
indows\Syst
indows\Syst
```

```
0:  90 06 40 00 00 00 00 00
0:  e8 62 08 f0 fe 07 00 00
0:  00 00 00 00 00 00 00 00
0:  00 00 00 00 00 00 00 00
0:  00 00 00 00 00 00 00 00

0xCD
0xAB
0xCA
0xDB,  Success!
.dll   base: 0x7fef0030000
```


Process Explorer - Sysinternals: www.sysinternals.com [WIN-TR2STH0EEVN\rc]
File
Options
View
Process
Find
DLL
Users
Help

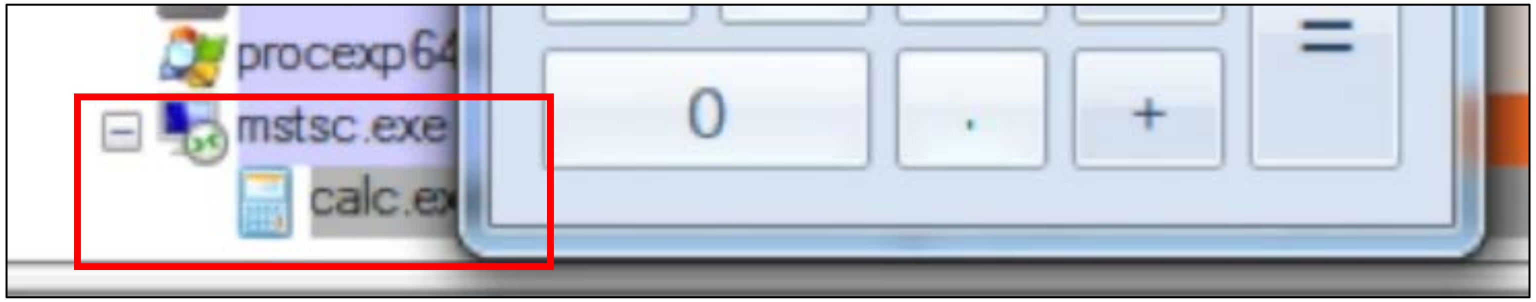
Process	CPU	Private Bytes	Working Set	PID	Description	Company Name
sppsvc.exe		8,240 K	12,940 K	1128	Microsoft Software Protectio...	Microsoft Corporation
svchost.exe		65,044 K	33,800 K	2836	Host Process for Windows S...	Microsoft Corporation
TrustedInstaller.exe		35,676 K	40,888 K	2804	Windows Modules Installer	Microsoft Corporation
lsass.exe	0.17	4,664 K	12,524 K	500	Local Security Authority Proc...	Microsoft Corporation
lsm.exe		2,556 K	4,504 K	508	Local Session Manager Serv...	Microsoft Corporation
csrss.exe	0.11	9,332 K	15,292 K	416	Client Server Runtime Process	Microsoft Corporation
winlogon.exe		2,892 K	7,592 K	492	Windows Logon Application	Microsoft Corporation
explorer.exe	0.09	40,928 K	78,376 K	1616	Windows Explorer	Microsoft Corporation
vmtoolsd.exe	0.05	13,156 K	23,980 K	1260	VMware Tools Core Service	VMware, Inc.
ida64.exe	0.14	193,008 K	225,588 K	2528	The Interactive Disassembler	Hex-Rays SA
notepad.exe		1,636 K	9,508 K	344	Notepad	Microsoft Corporation
procexp64.exe	1.40	16,636 K	28,848 K	2400	Sysinternals Process Explorer	Sysinternals - www.sysinter...

Name	Description	Company Name	Path
{AFBF9F1A-8EE8-4...			C:\Users\rc\AppData\Local\Microsoft\Windows\Caches\{...
aclui.dll	Security Descriptor Editor	Microsoft Corporation	C:\Windows\System32\aclui.dll
aclui.dll.mui	Security Descriptor Editor	Microsoft Corporation	C:\Windows\System32\en-US\aclui.dll.mui
advapi32.dll	Advanced Windows 32 Base API	Microsoft Corporation	C:\Windows\System32\advapi32.dll
apisetschema.dll	ApiSet Schema DLL	Microsoft Corporation	C:\Windows\System32\apisetschema.dll
apphelp.dll	Application Compatibility Client Libr...	Microsoft Corporation	C:\Windows\System32\apphelp.dll
apphelp.dll.mui	Application Compatibility Client Libr...	Microsoft Corporation	C:\Windows\System32\en-US\apphelp.dll.mui
bcrypt.dll	Windows Cryptographic Primitives ...	Microsoft Corporation	C:\Windows\System32\bcrypt.dll
bcryptprimitives.dll	Windows Cryptographic Primitives ...	Microsoft Corporation	C:\Windows\System32\bcryptprimitives.dll
cfgmgr32.dll	Configuration Manager DLL	Microsoft Corporation	C:\Windows\System32\cfgmgr32.dll
clbcatq.dll	COM+ Configuration Catalog	Microsoft Corporation	C:\Windows\System32\clbcatq.dll
comctl32.dll	User Experience Controls Library	Microsoft Corporation	C:\Windows\winsxs\amd64_microsoft.windows.common-co...
comctl32.dll.mui	User Experience Controls Library	Microsoft Corporation	C:\Windows\winsxs\amd64_microsoft.windows.c..-controls.r...
comdlg32.dll	Common Dialogs DLL	Microsoft Corporation	C:\Windows\System32\comdlg32.dll
comdlg32.dll.mui	Common Dialogs DLL	Microsoft Corporation	C:\Windows\System32\en-US\comdlg32.dll.mui

CPU Usage: 37.17% | Commit Charge: 27.78% | Processes: 44 | Physical Usage: 54.41%

DEMO

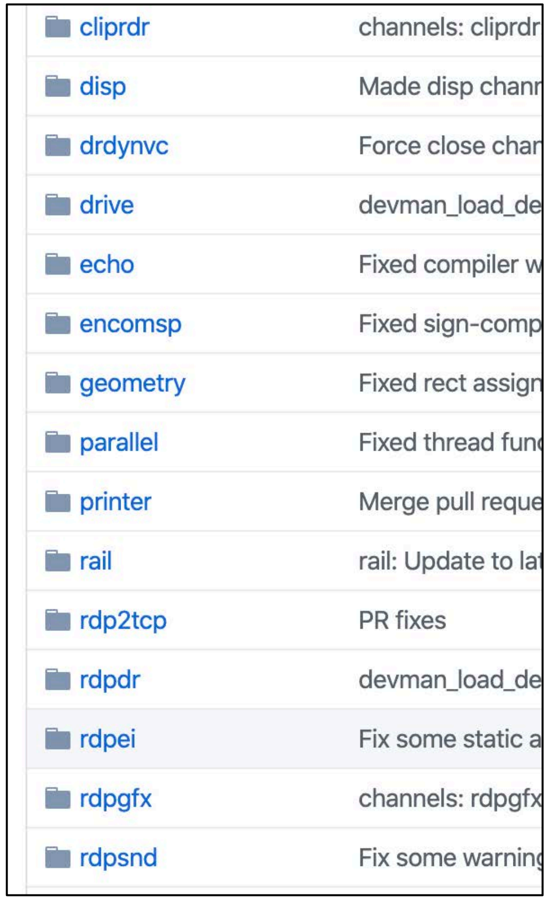
- Remote Code Execution
  - Mstsc.exe on windows 7
  - Using RDP Heap Feng Shui
    - Leak memory to break ASLR
    - Allocate heap objects in a pretty good construction
    - Hijacking a vtable of an object and launch Return-oriented Programming (ROP)
      - Pops a calc.exe!



[illegible]

FUTURE WORK

- Fuzzing on other channels
  - Drdynvc
  - Printer
  - Smartcard
  - …
- Need to handle two-way communication
  - Modifying the program at the server is required



cliprdr	channels: cliprdr
disp	Made disp chann
drdynvc	Force close char
drive	devman_load_de
echo	Fixed compiler w
encomsp	Fixed sign-comp
geometry	Fixed rect assign
parallel	Fixed thread fun
printer	Merge pull reque
rail	rail: Update to la
rdp2tcp	PR fixes
rdpdr	devman_load_de
rdpei	Fix some static a
rdpgfx	channels: rdpgfx
rdpsnd	Fix some warnin

FUTURE WORK & Conclusion

- We have learned coverage-guided fuzzing to
  - Windows Application
  - Server-client model
  - Applications with no source code available
  - Without modifying the binary program (no requirement for syzygy)

FUTURE WORK & Conclusion

- We have learned coverage-guided fuzzing to
  - Windows Application
  - Server-client model
  - Applications with no source code available
  - Without modifying the binary program (no requirement for syzygy)

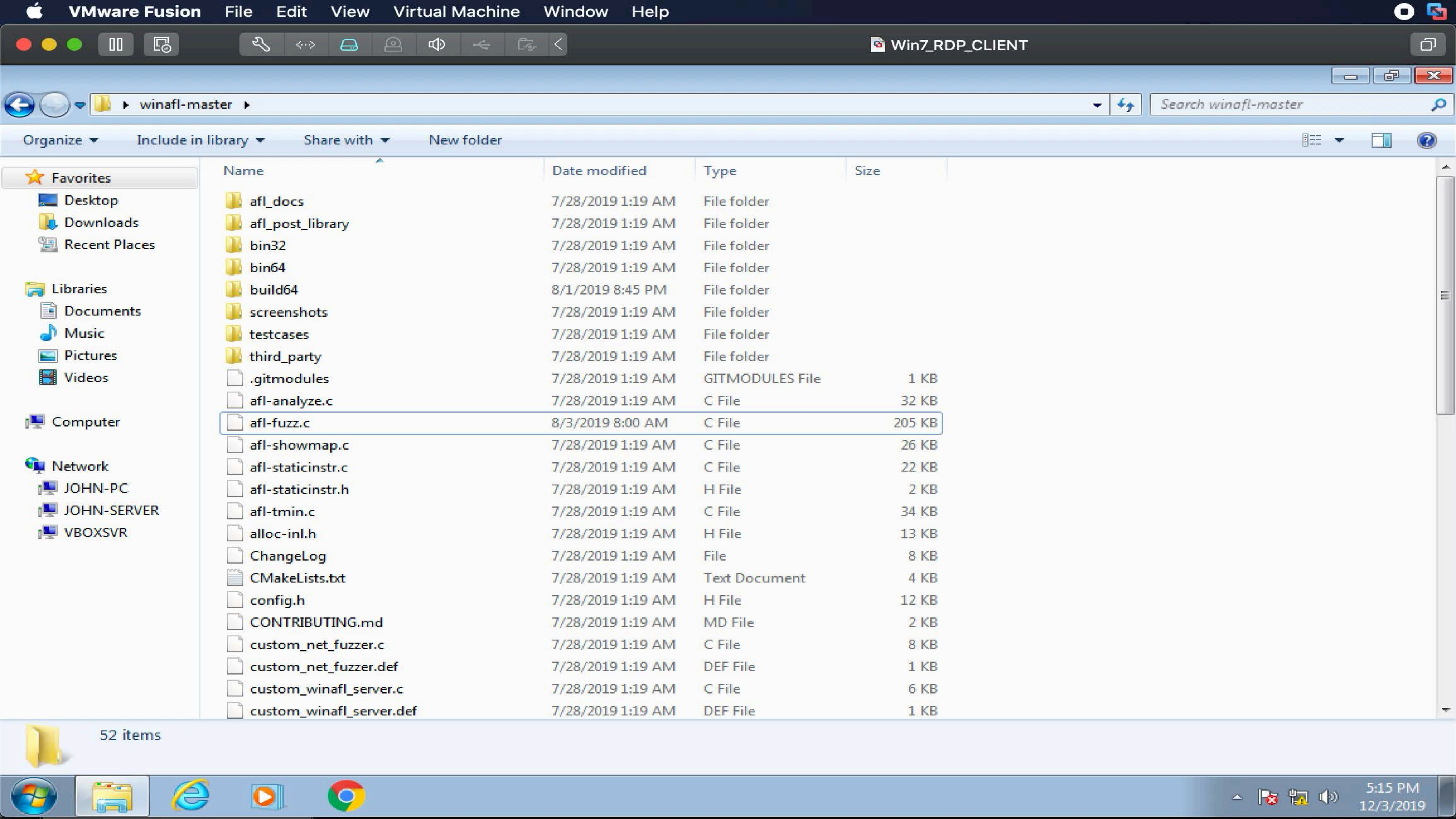
- Don’t be afraid the application runs over server-client and does not accept file input

THANKS

- Q&A

DEMO

- RDP CLIENT FUZZER



Name	Date modified	Type	Size
afl_docs	7/28/2019 1:19 AM	File folder	
afl_post_library	7/28/2019 1:19 AM	File folder	
bin32	7/28/2019 1:19 AM	File folder	
bin64	7/28/2019 1:19 AM	File folder	
build64	8/1/2019 8:45 PM	File folder	
screenshots	7/28/2019 1:19 AM	File folder	
testcases	7/28/2019 1:19 AM	File folder	
third_party	7/28/2019 1:19 AM	File folder	
.gitmodules	7/28/2019 1:19 AM	GITMODULES File	1 KB
afl-analyze.c	7/28/2019 1:19 AM	C File	32 KB
afl-fuzz.c	8/3/2019 8:00 AM	C File	205 KB
afl-showmap.c	7/28/2019 1:19 AM	C File	26 KB
afl-staticinstr.c	7/28/2019 1:19 AM	C File	22 KB
afl-staticinstr.h	7/28/2019 1:19 AM	H File	2 KB
afl-tmin.c	7/28/2019 1:19 AM	C File	34 KB
alloc-inl.h	7/28/2019 1:19 AM	H File	13 KB
ChangeLog	7/28/2019 1:19 AM	File	8 KB
CMakeLists.txt	7/28/2019 1:19 AM	Text Document	4 KB
config.h	7/28/2019 1:19 AM	H File	12 KB
CONTRIBUTING.md	7/28/2019 1:19 AM	MD File	2 KB
custom_net_fuzzer.c	7/28/2019 1:19 AM	C File	8 KB
custom_net_fuzzer.def	7/28/2019 1:19 AM	DEF File	1 KB
custom_winafl_server.c	7/28/2019 1:19 AM	C File	6 KB
custom_winafl_server.def	7/28/2019 1:19 AM	DEF File	1 KB

52 items

5:15 PM
12/3/2019

THANKS

- Q&A