

# Holding the Stick

# · RDPFUZZ ·

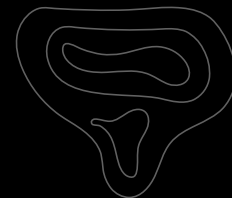
# At Both Ends

Or Ben-Porath & Shaked Reiner

CyberArk Labs




# Who are we



Shaked Reiner

Principal Security Researcher

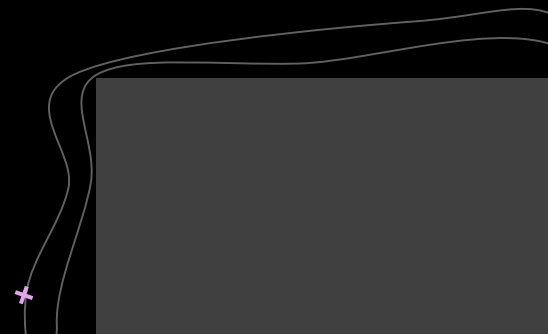
 @ShakReiner



Or Ben-Porath

Security Researcher

 @OrBenPorath



# Agenda



## 01 RDP


How the protocol works

## 02 Fuzzing RDP

Our fuzzing process

## 03 Results

Fuzzing Stats



## 04 Summary

Conslusion and future work

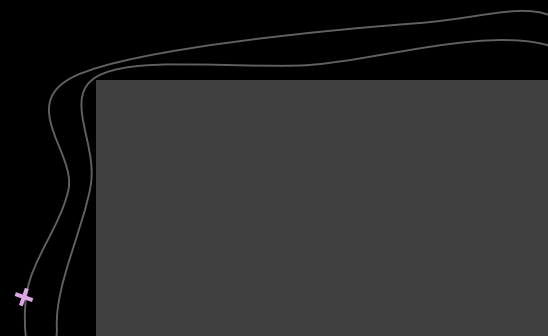


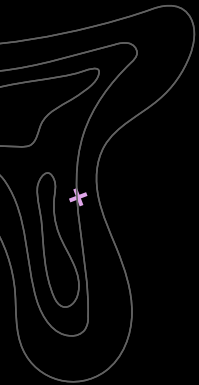


# 01

## RDP

Why RDP? What's the attack surface? How does it work?





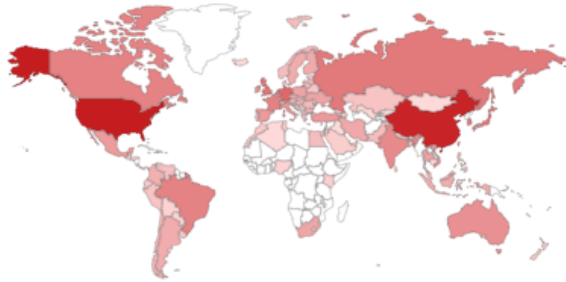
## TOTAL RESULTS

---

4,464,020

## TOP COUNTRIES

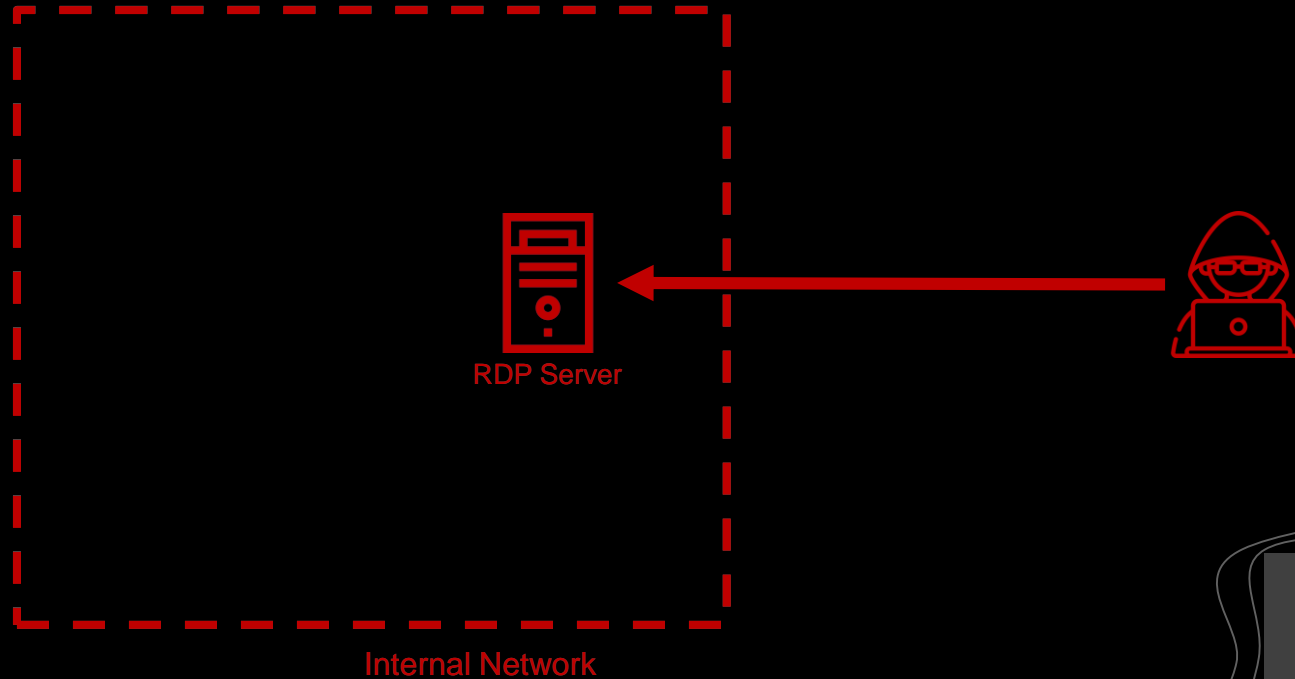
---



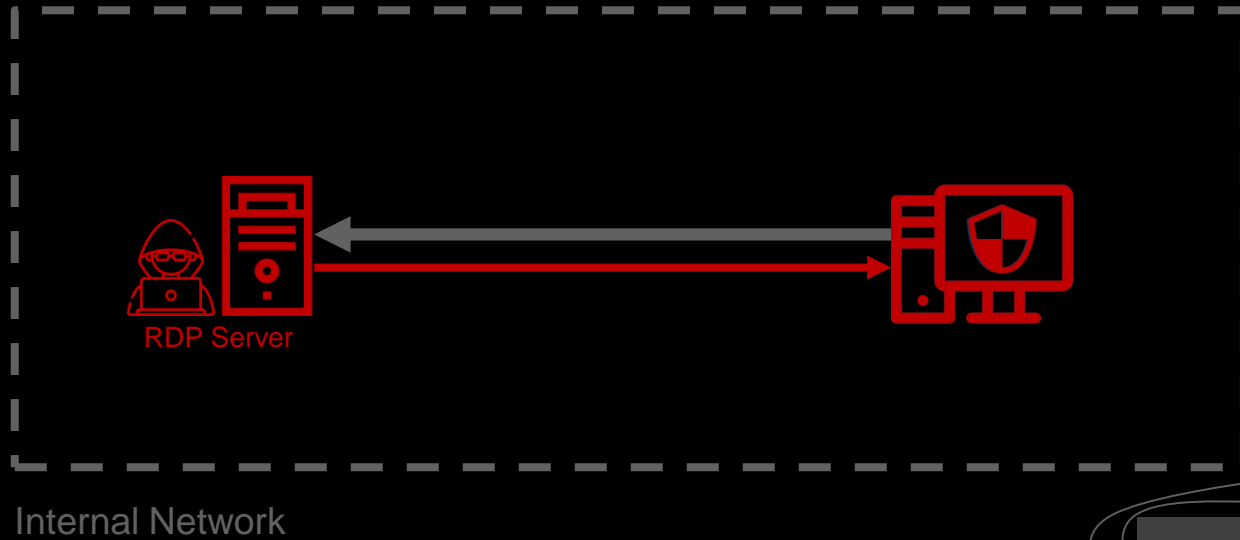
<b>United States</b>	<b>1,452,039</b>
<b>China</b>	<b>1,158,306</b>
<b>Germany</b>	<b>193,084</b>
<b>Netherlands</b>	<b>126,017</b>
<b>United Kingdom</b>	<b>118,304</b>

# RDP In the Wild

# RDP Attack Vector #1



# RDP Attack Vector #2





# Examples



## DejaBlue

Int overflow

RCE



## Reverse RDP

Path traversal in clipboard

Run arbitrary applications

# Attack Surface

```
PS C:\Users\shak> gci -Include *.exe, *.dll, *.sys -Recurse C:\Windows\ -ErrorAction SilentlyContinue |  
?{[System.Diagnostics.FileVersionInfo]::GetVersionInfo($_).FileDescription -match "RDP|Remote Desktop"}  
| Measure-Object | select count
```

Count

-----

191



How Does RDP Work?



# Read Surface

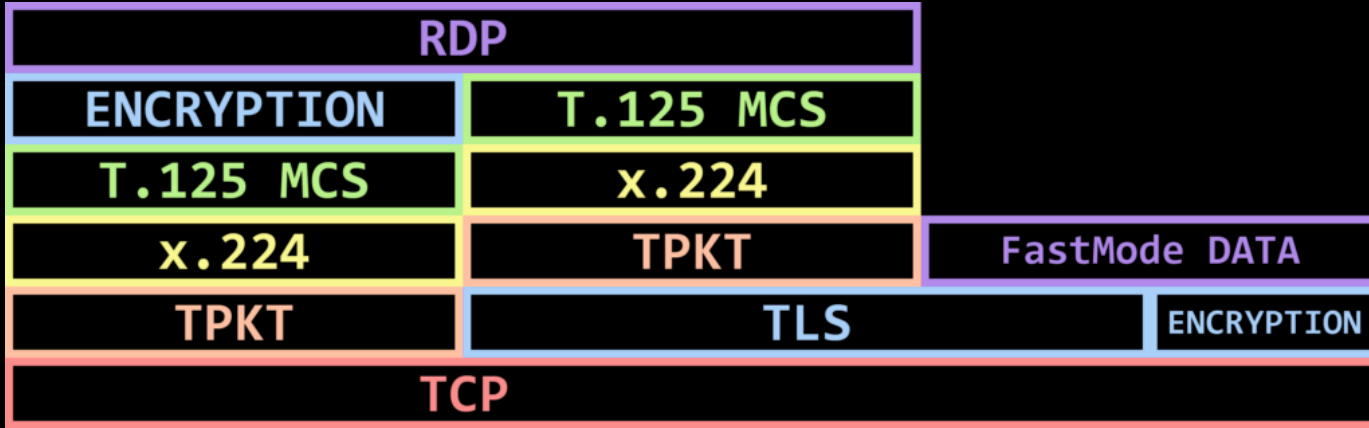
```
PS C:\research\RDP\RDP-SPECS> gci -r *MS-RDP*.pdf | Measure-Object | select count
```

```
Count
```

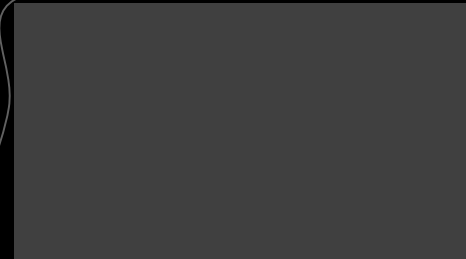
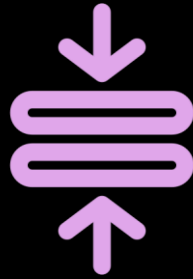
```
-----
```

```
30
```

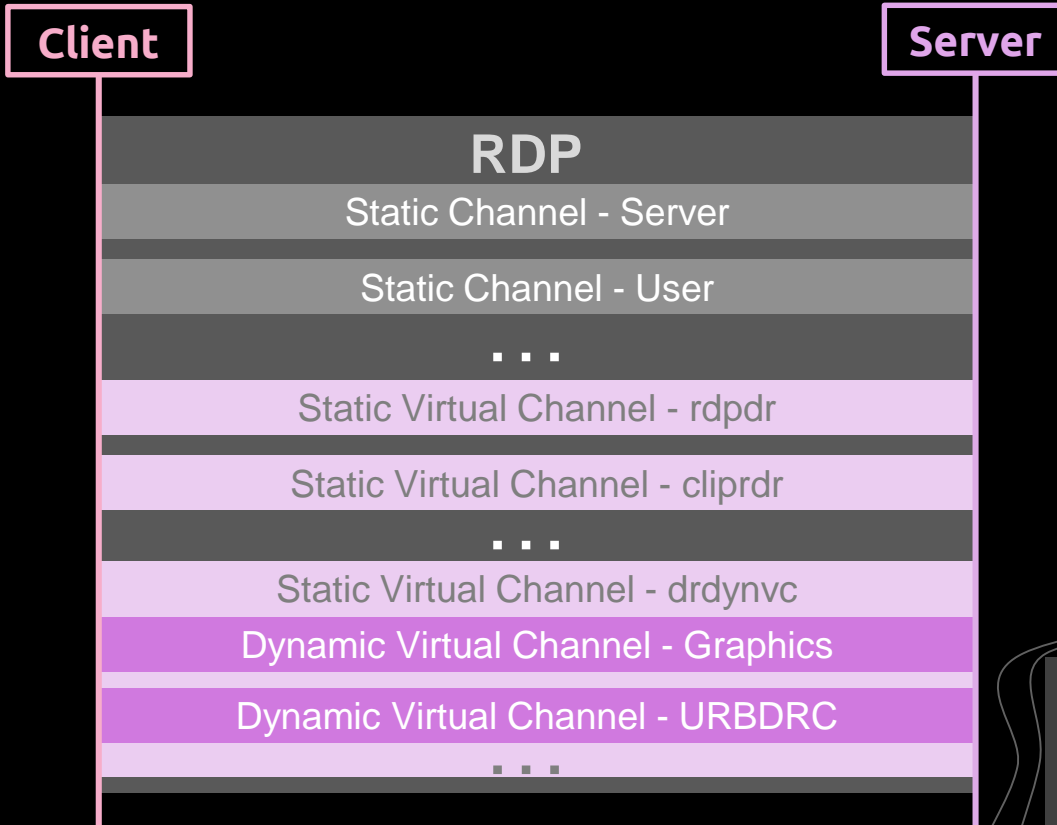
# Protocol Stack



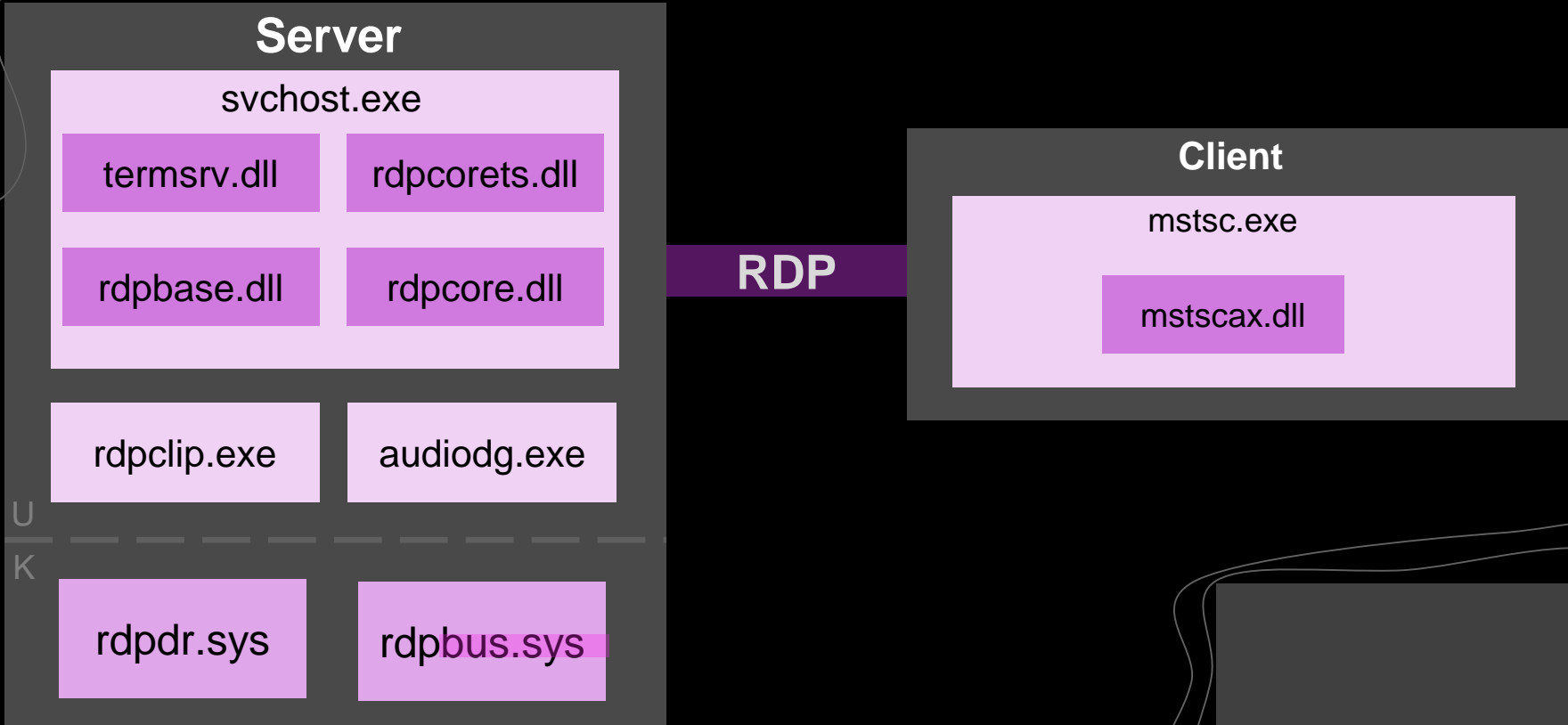
# General Info



# RDP Connection



# RDP Components





# RDP Components

## Server

svchost.exe

termsrv.dll

rdp

rdpbase.dll

rdp

rdpclip.exe

aud

rdpdr.sys

rdp

### Functions

Function name	Seg
CRdpAudioController::StopCloseTimer(void)	.tex
CRdpAudioController::UpdateAndGetDataBufferedInDeviceInfo(u...	.tex
CRdpAudioController::UpdateDataBufferedInDevice(ulong)	.tex
CRdpAudioController::`scalar deleting destructor'(uint)	.tex
CRdpAudioController::sendQualityMode(void)	.tex
CRdpAudioController::vcwaveGetDevCaps(SNDFORMATMSG *)	.tex
CRdpAudioController::~CRdpAudioController(void)	.tex
CRdpAudioPlaybackChannelCallback::CRdpAudioPlaybackChan...	.tex
CRdpAudioPlaybackChannelCallback::CloseChannel(void)	.tex
CRdpAudioPlaybackChannelCallback::CreateInstance(RdpXInterf...	.tex
CRdpAudioPlaybackChannelCallback::OnClose(void)	.tex
CRdpAudioPlaybackChannelCallback::OnDataReceived(ulong,uc...	.tex
CRdpAudioPlaybackChannelCallback::Terminate(void)	.tex
CRdpAudioPlaybackChannelCallback::`scalar deleting destructor'...	.tex
CRdpAudioPlaybackChannelCallback::~CRdpAudioPlaybackCha...	.tex
CRdpAudioPlaybackDVCPluginCRdpAudioPlaybackDVCPlugin/...	.tex

Line 12446 of 18947

## Client

mstsc.exe

mstscax.dll

U  
K



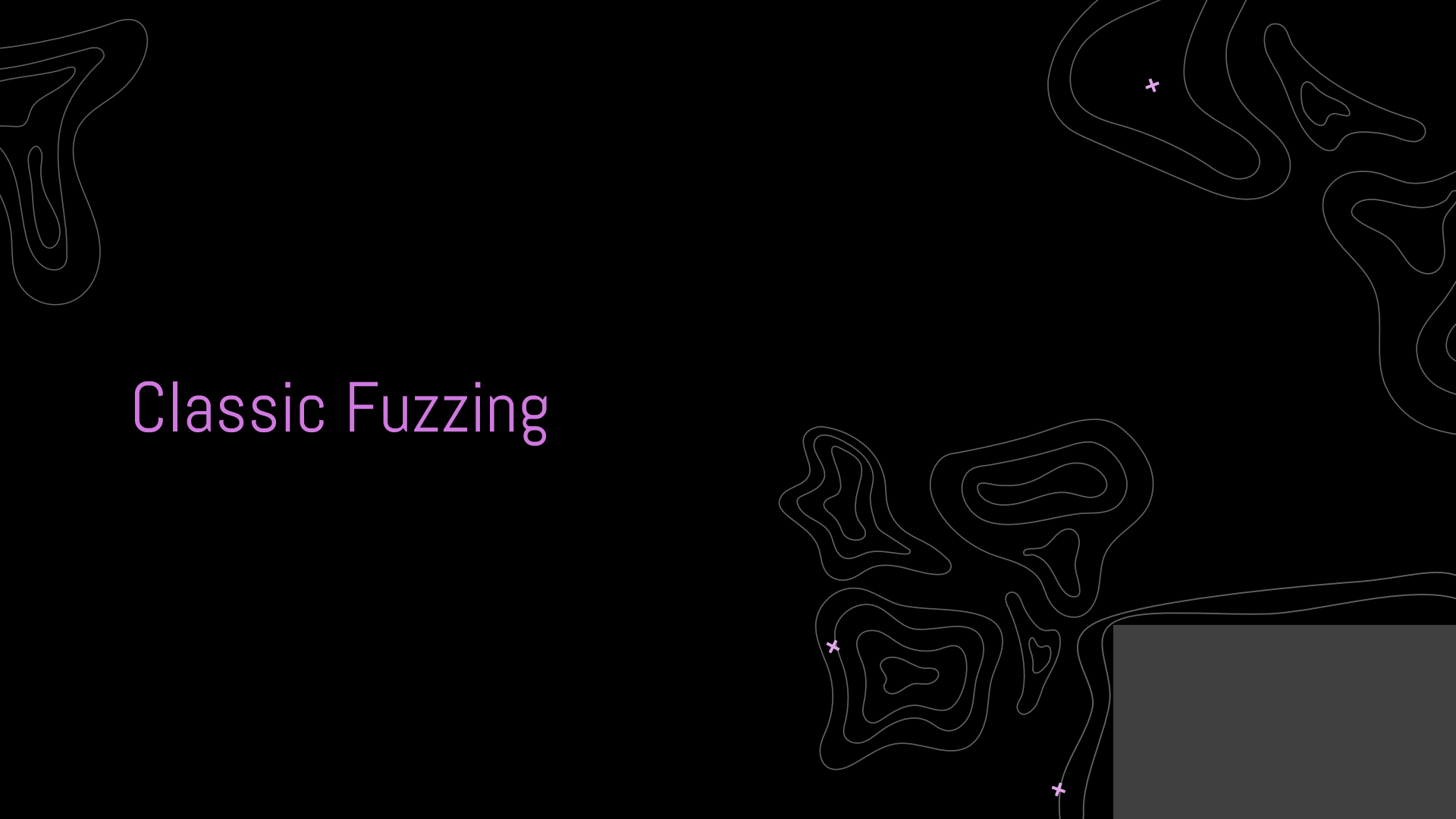
# 02

## RDPFuZZ

Fuzzing RDP



# Classic Fuzzing



# Coverage-guided Fuzzing Setup



# Fuzzing RDP



# Fuzzing Options



## Open-source

Use a modified open-source client/server



## Custom

Write our own client and server



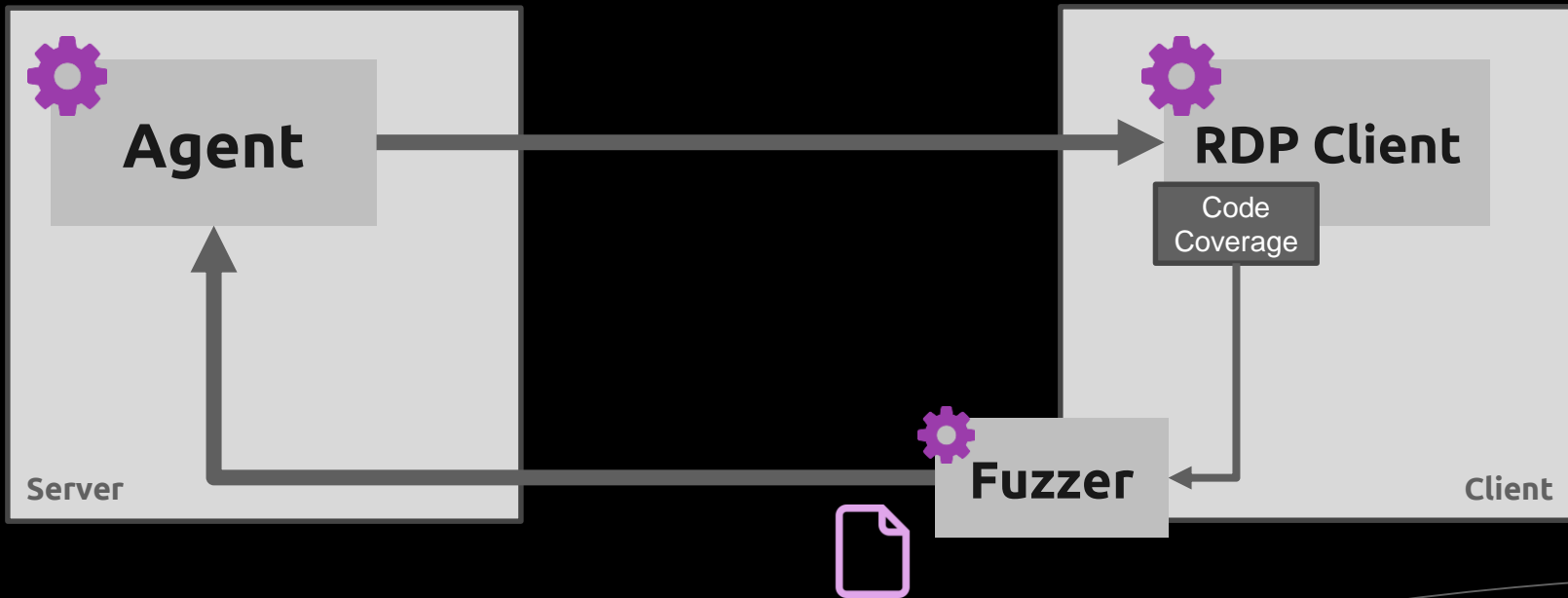
## Snapshot

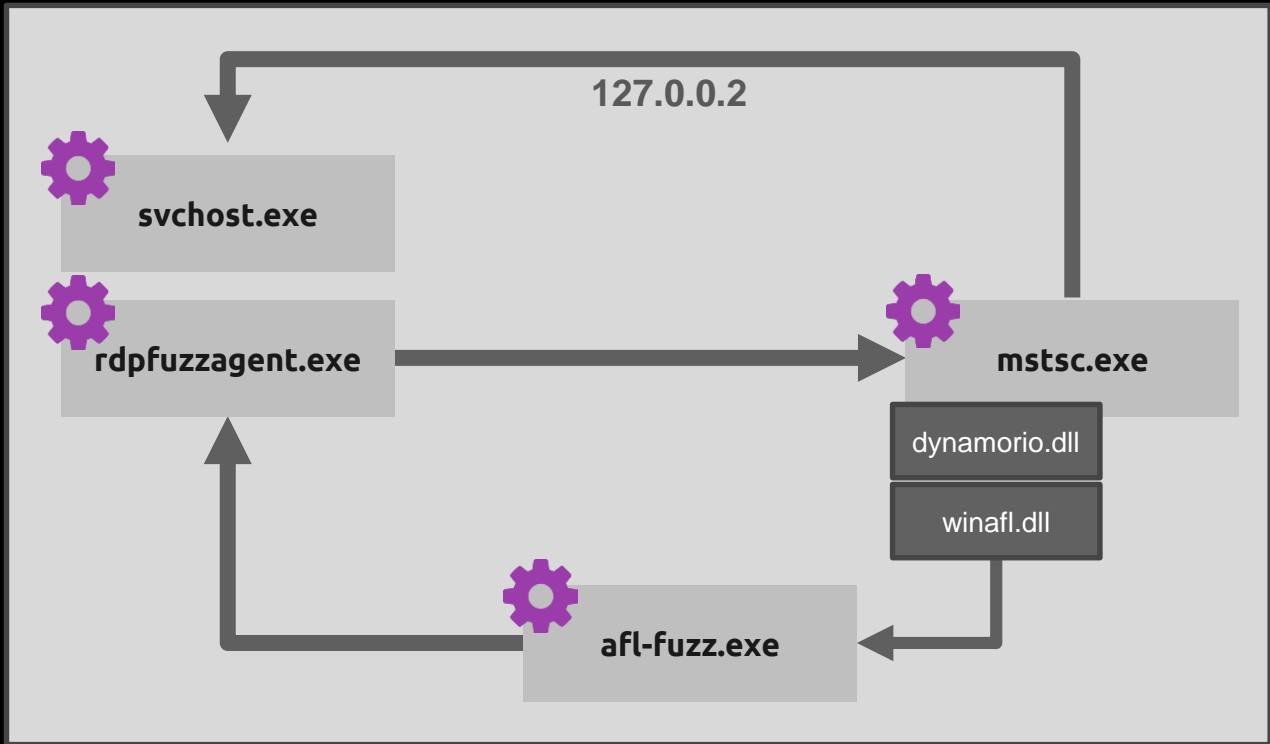
Use a snapshot-based fuzzer



## Use existing

Tap into the client and server using legitimate APIs or code injection

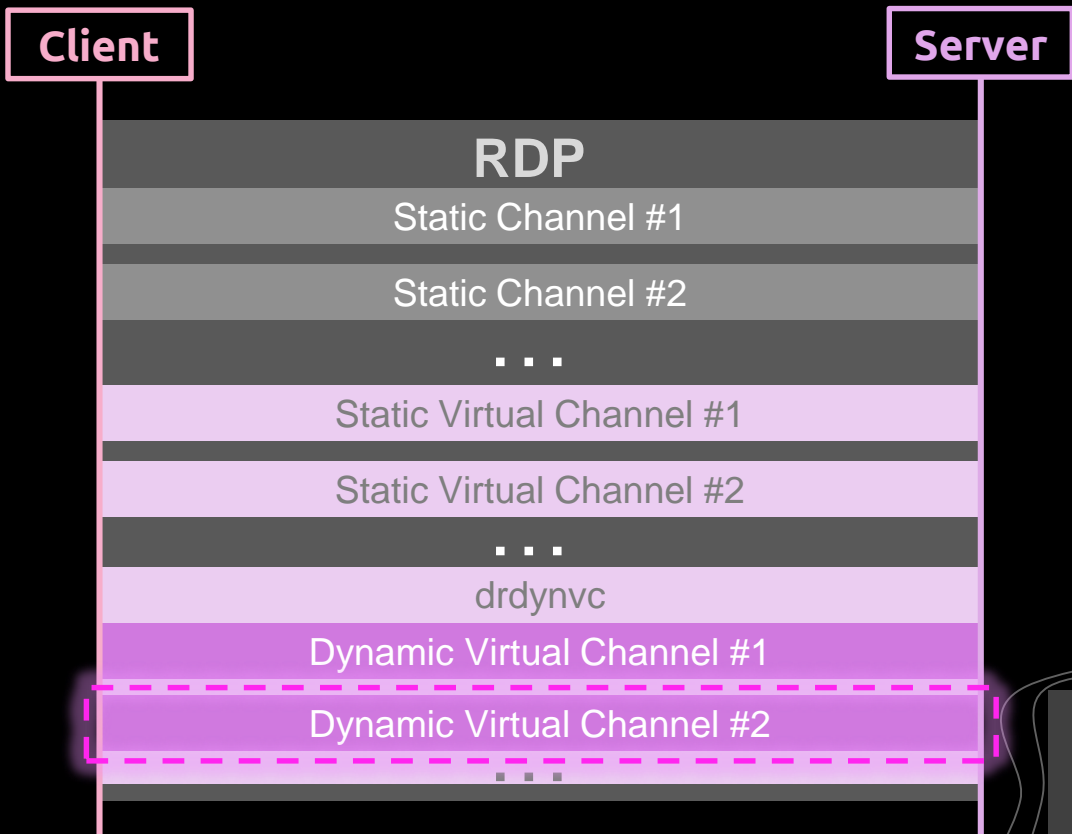




x 10



# RDP Connection



# DR Attach



# Background Fuzzing

```
172.22.87.240 - Remote Desktop Connection

american fuzzy lop 2.49b (fuzzer1)

process timing
run time : 1 days, 16 hrs, 24 min, 6 sec
last new path : 0 days, 0 hrs, 0 min, 43 sec
last uniq crash : 0 days, 15 hrs, 50 min, 54 sec
last uniq hang : 0 days, 0 hrs, 12 min, 7 sec

cycle progress
now processing : 17 (0.48%)
paths timed out : 0 (0.00%)

stage progress
now trying : bitFlip 1/1
stage execs : 74.2k/113k (65.66%)
total execs : 21.8M
exec speed : 153.1/sec

overall results
cycles done : 0
total paths : 3570
uniq crashes : 2
uniq hangs : 254

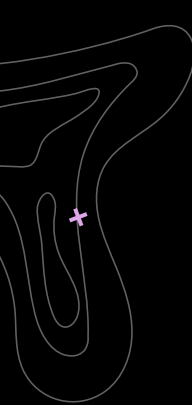
map coverage
map density : 6.59% / 15.77%
count coverage : 5.54 bits/tuple

findings in depth
favored paths : 308 (8.63%)
new edges on : 330 (9.24%)
total crashes : 2 (2 unique)
total trouts : 4191 (294 unique)

fuzzing strategy yields
bit flips : 1355/816k, 148/816k, 67/816k
byte flips : 0/102k, 0/101k, 4/101k
arithmetics : 308/5.67M, 1/300k, 0/0
known ints : 22/529k, 2/2.82M, 3/4.49M
dictionary : 0/0, 0/0, 139/5.10M
havoc : 6/4040, 0/0
triple : 0.00%/11.7k, 0.00%

path geometry
levels : 2
pending : 3563
pend fail : 308
non finds : 2111
unreported : 1435
stability : 84.24%

[cpu000:101k]
```



20 00 00 44  
03 41 42 43

20 00 00 44  
03 41 42 43

FF 00 00 44  
03 41 42 43



## 2.2.1 RDPSND PDU Header (SNDPROLOG)

10/30/2020 - 2 minutes to read

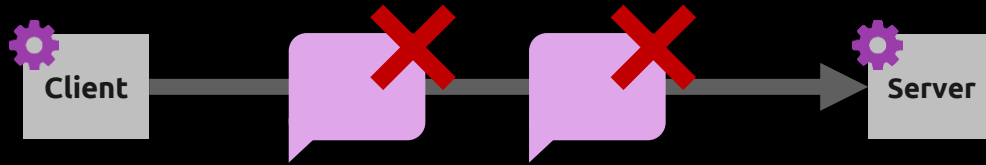
The RDPSND PDU header is present in many audio PDUs. It is used to identify the PDU type, specify the length of the PDU, and convey message flags.



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

Value	Meaning
SND_C_CLOSE 0x01	Close PDU
SND_C_WAVE 0x02	Waveinfo PDU
SND_C_SETVOLUME 0x03	Volume PDU
SND_C_SETPITCH 0x04	Pitch PDU
SND_C_WAVECONFIRM 0x05	Wave Confirm PDU
SND_C_TRAINING 0x06	Training PDU or Training Confirm PDU
SND_C_FORMATS 0x07	Server Audio Formats and Version PDU or Client Audio Formats and Version PDU
SND_C_CRYPTKEY 0x08	Crypt Key PDU
SND_C_WAVEENCRYPT 0x09	Wave Encrypt PDU
SND_C_UDPWAVE 0x0A	UDP Wave PDU
SND_C_UDPWAVELAST 0x0B	UDP Wave Last PDU
SND_C_QUALITYMODE 0x0C	Quality Mode PDU
SND_C_WAVE2 0x0D	Wave2 PDU

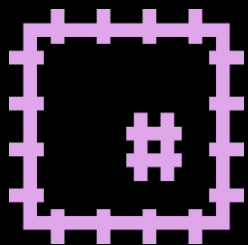
# Statefulness



# Statefulness



# Statefulness



Code patches



Grammar enforcement



# Code Patches

```
loc_16A7C1276:  
cmp     dword ptr [rdi+88h], 0  
jz      loc_16A7C1382
```

```
Error Logging
```

```
loc_16A7C12C4:          ; this  
lea     rcx, [rdi-30h]  
mov     edx, esi        ; int  
call   ?ReconnectSoftwareMode@RdpGfxClientChannel@@AEA@J@Z ; RdpGfxClientChannel::ReconnectSoftwareMode(long)  
mov     esi, eax  
test    eax, eax  
jns     short loc_16A7C1313
```

```
Exit
```

mstscax!RdpGfxClientChannel::OnDataReceived

## ◆ drwrap\_replace()

```
DR_EXPORT bool drwrap_replace ( app_pc original,  
                               app_pc replacement,  
                               bool  override  
                               )
```

Replaces the application function that starts at the address `original` with the code at the address `replacement`.

Only one replacement is supported per target address. If a replacement already exists for `original`, this function fails unless `override` is true, in which case it replaces the prior replacement. To remove a replacement, pass NULL for `replacement` and **true** for `override`. When removing or replacing a prior replacement, existing replaced code in the code cache will be flushed lazily: i.e., there may be some execution in other threads after this call is made.

Only the first target replacement address in a basic block will be honored. All code after that address is removed.

When replacing a function, it is up to the user to ensure that the replacement mirrors the calling convention and other semantics of the original function. The replacement code will be executed as application code, NOT as client code.

### Note

The priority of the app2app pass used here is `DRMGR_PRIORITY_APP2APP_DRWRAP` and its name is `DRMGR_PRIORITY_NAME_DRWRAP`.

### Returns

whether successful.

# Grammar Enforcement



## Documentation

Reading the docs



## RE

Analyzing the  
conditions within  
the code



## Tracing

Analyzing failed  
executions

# Grammar Enforcement



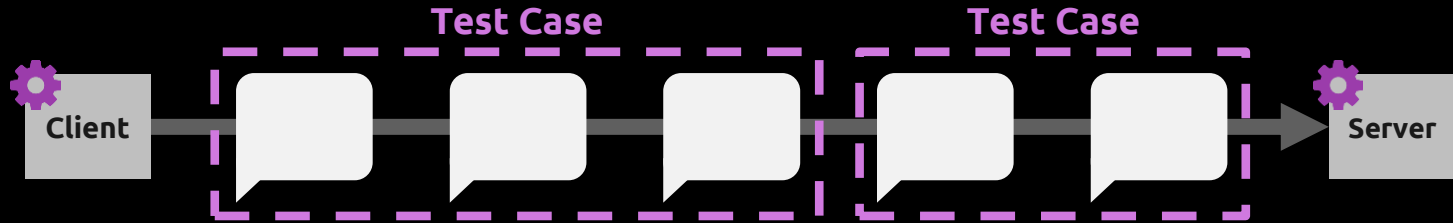
## Limit Fuzzer

Grammar narrows down  
the input space

# Multi-input



# Multi-input



## 2.2.2.9 RDPGFX\_CREATE\_SURFACE\_PDU

04/07/2021 • 2 minutes to read

The `RDPGFX_CREATE_SURFACE_PDU` message is used to instruct the client to create a surface of a given width, height, and pixel format.

0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
header																															
...																															
surfaceId																width															
height																pixelFormat															

## 2.2.2.15 RDPGFX\_MAP\_SURFACE\_TO\_OUTPUT\_PDU

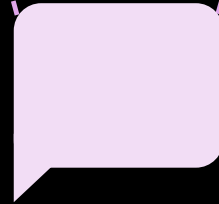
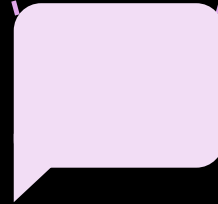
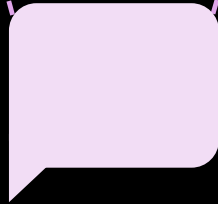
02/14/2019 • 2 minutes to read

The `RDPGFX_MAP_SURFACE_TO_OUTPUT_PDU` message is sent by the server to instruct the client to map a surface to a rectangular area of the `Graphics Output Buffer` (section 3.3.1.7) ADM element.

0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
header																															
...																															
surfaceId																reserved															
outputOriginX																															
outputOriginY																															

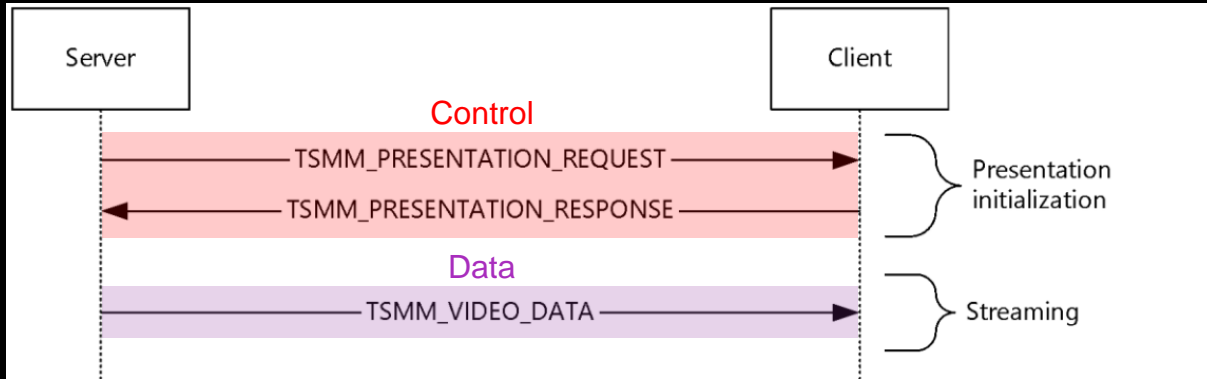
## Test Case

\_\_cmd07 <1st PDU data> \_\_cmd02 <2nd PDU data> \_\_cmd03 <3rd PDU data>

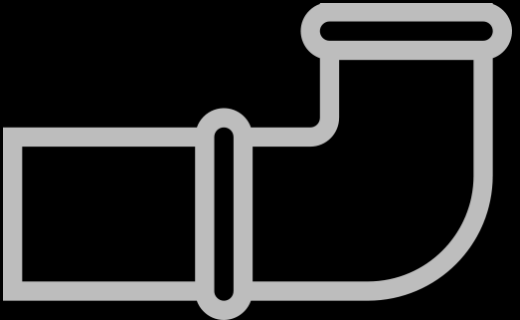




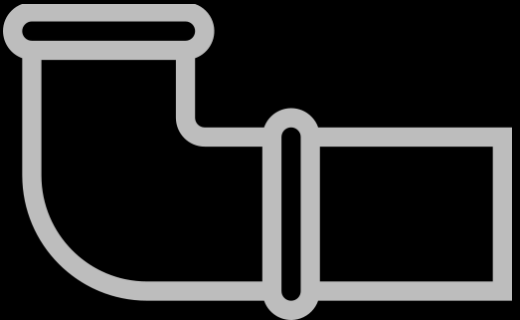
# Multi-channel Input



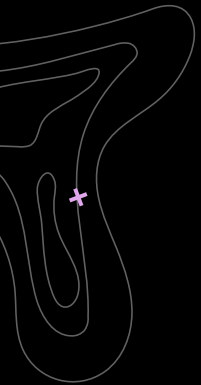
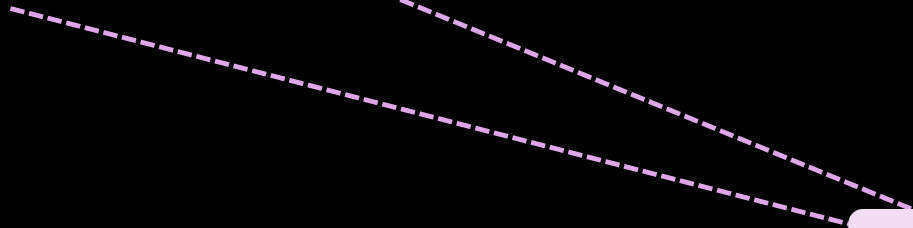
\_\_cmd07 <1st PDU data> \_\_\_cmd02 <2nd PDU data> \_\_\_cmd03 <3rd PDU data>



**Video::Data**

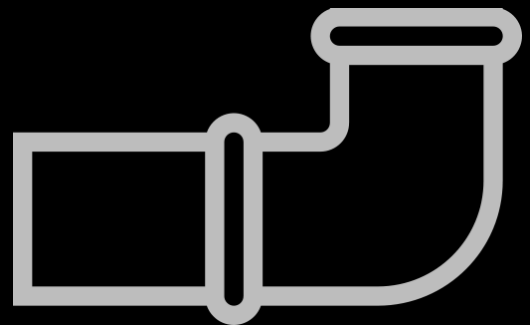


**Video::Control**

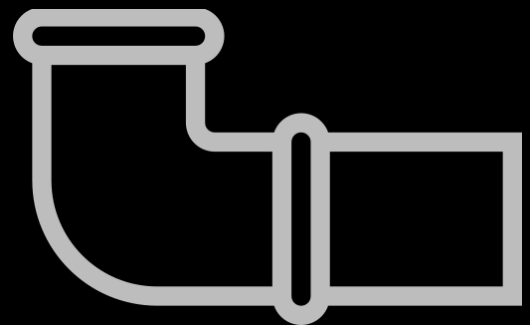




\_\_cmd07 <1st PDU data>    \_\_cmd02 <2nd PDU data>    \_\_cmd03 <3rd PDU data>



**Video::Data**

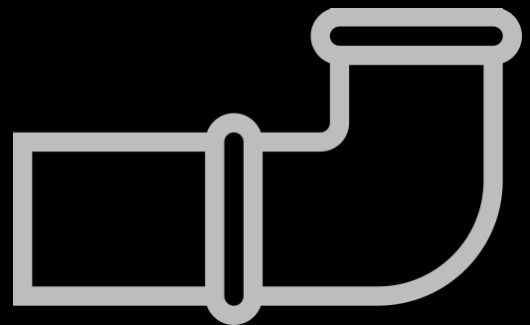
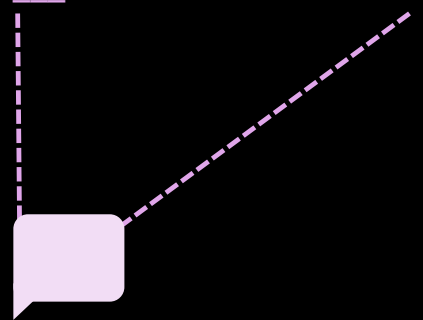


**Video::Control**

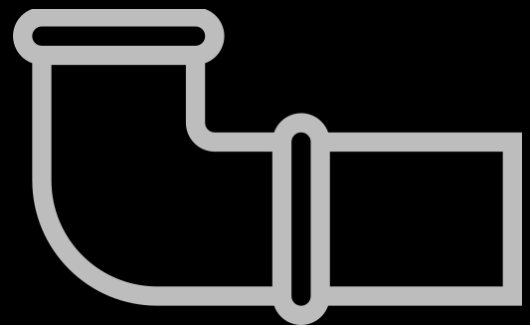




\_\_cmd07 <1st PDU data> \_\_cmd02 <2nd PDU data> \_\_cmd03 <3rd PDU data>



**Video::Data**

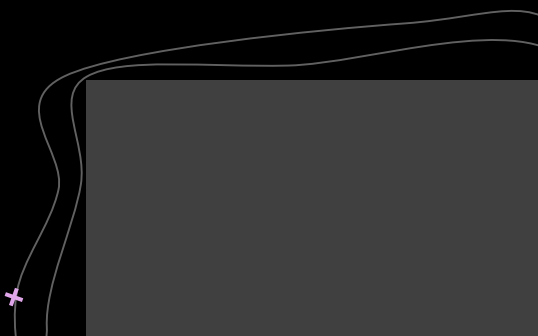


**Video::Control**





# Locating Target Functions

- + Too many components
  - + PDBDownloader to download all relevant .pdb files
  - + grep/sls to get all the `C<class-name>::OnDataReceived` functions
- 

**A FEW  
FUZZING ITERATIONS LATER**

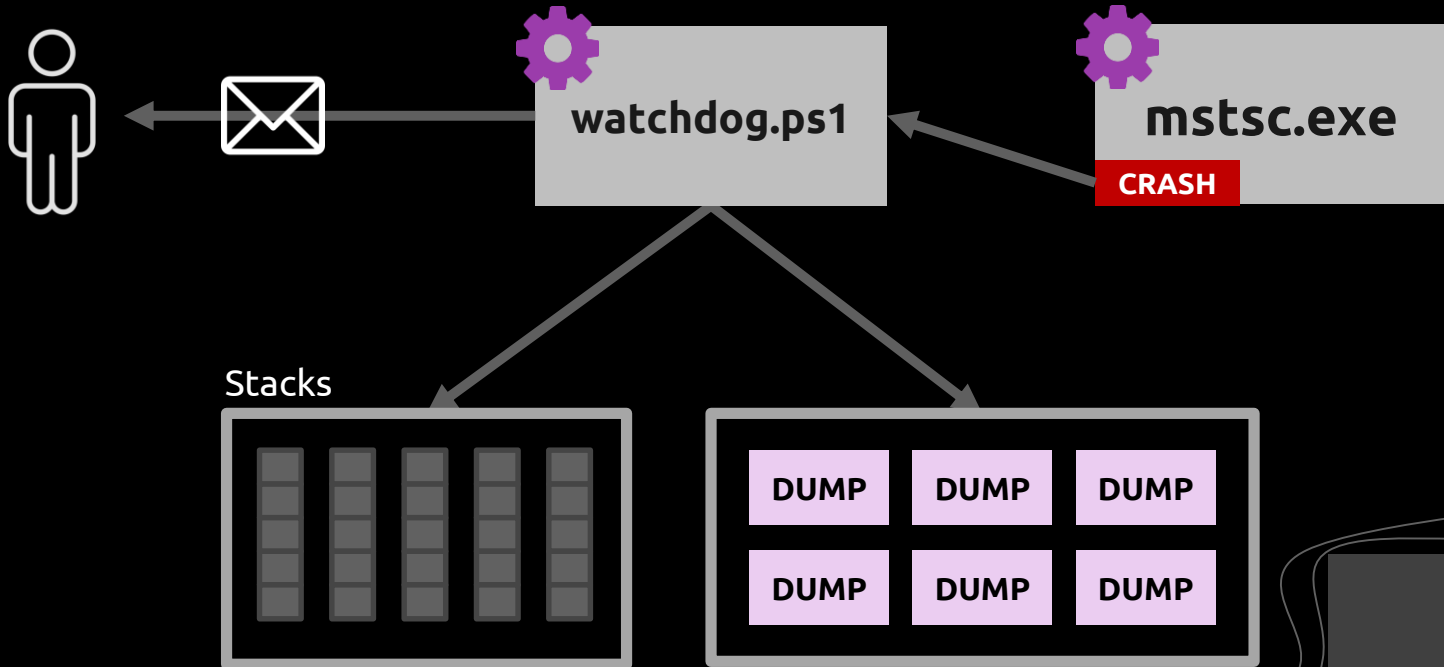


# Reproduction Issues



Sorry, no crash

# Automatic Crash Analysis







From rdpfuzzmonitor

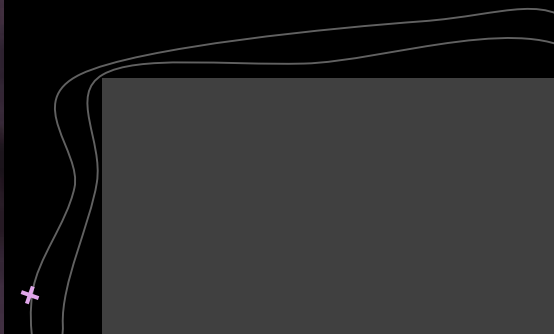
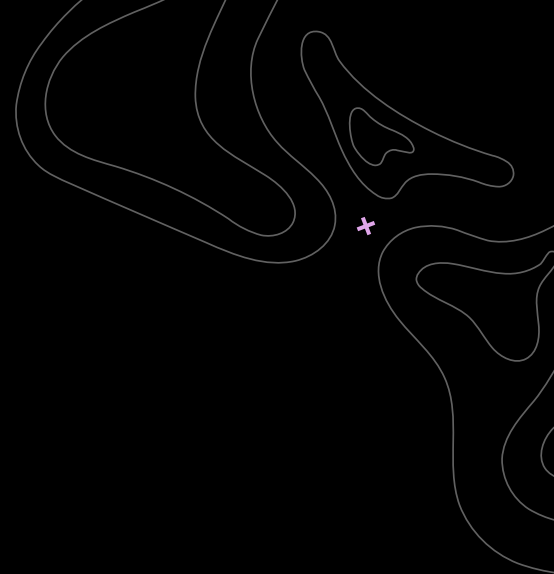
Subject NEW CRASH in AS\_SERVER4 :)

Body

```
mstscax!CRdpAudioController::OnWaveData+0x281
mstscax!CRdpAudioController::DataArrived+0x72f
mstscax!CRdpAudioPlaybackChannelCallback::OnDataReceived+0x433
mstscax!CDynVCChannel::InvokeCallback+0x1b0
mstscax!CDynVCChannel::OnData+0x3aa
mstscax!CDynVCPlugin::OnStaticDataReceived+0x14f
mstscax!CStaticChannelCallback::OnDataReceived+0x24
mstscax!CCommonVCChannel::OpenProcEx+0x31e
mstscax!CCommonVCChannel::static_OpenProcEx+0xc6
mstscax!CChan::ChannelOnPacketReceived+0x179
mstscax!CSL::SLReceivedDataPacket+0x110
mstscax!CSL::OnPacketReceived+0x19d
mstscax!CMCS::MCSRecvData+0x20f
mstscax!CMCS::OnDataAvailable+0xdd
mstscax!CTSx224Filter::OnDataAvailable+0x138
mstscax!CTscSslFilter::OnDataAvailable+0xda
mstscax!CTSFilterTransport::OnDataAvailable_TransportEvent+0x63
mstscax!CTSTransportStack::OnDataAvailable+0x106
mstscax!CTSTcpTransport::AsyncOnReadCompletedAsyncCallback::Invoke+0x60
mstscax!CTSMsg::Invoke+0xdc
mstscax!CTSThread::RunAllQueueEvents+0x219
mstscax!CTSThread::internalMsgPump+0x91
mstscax!CTSThread::internalThreadMsgLoop+0x14d
mstscax!CTSThread::ThreadMsgLoop+0x1c
mstscax!CRCV::RCVMain+0x170
mstscax!CTSThread::TSStaticThreadEntry+0x258
mstscax!PAL_System_Win32_ThreadProcWrapper+0x32
KERNEL32!BaseThreadInitThunk+0x14
ntdll!RtlUserThreadStart+0x21
```

# 03

## Results

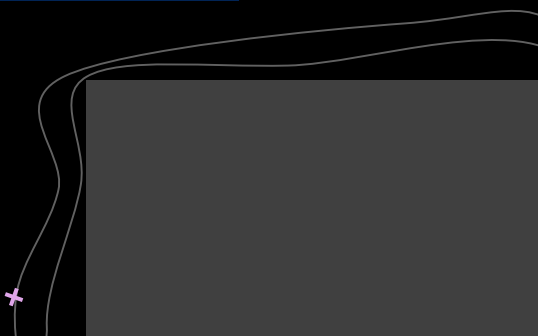




## Summary stats

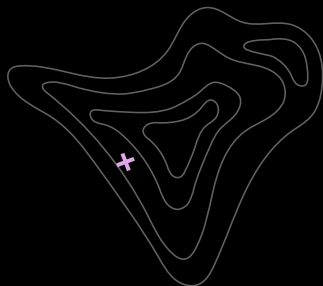
=====

```
Fuzzers alive : 10
Total run time : 0 days, 3 hours
  Total execs : 0 million
Cumulative speed : 41.39 execs/sec
  Pending paths : 154 faves, 664 total
Maximal coverage : 3.84%
Average coverage : 3.01%
Average stability : 15.56%
Pending per fuzzer : 15.40 faves, 66.40 total (on average)
  Crashes found : 17 locally unique
```



# ~1 month

Fuzzing duration



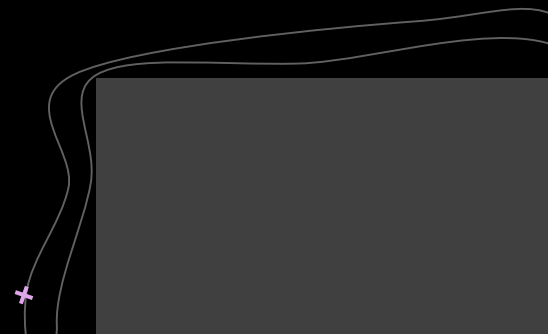
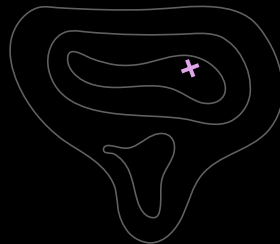
# 5

bugs



# 15

Channels fuzzed





**SHOW ME WHAT YOU GOT**

# AUDIO\_PLAYBACK Channel



```


0:000> g
ModLoad: 00007ff8`ca9d0000 00007ff8`ca9f4000 C:\Windows\SYSTEM32\edputil.dll
(ac9c.e600): Access violation - code c0000005 (first/second chance not available)
First chance exceptions are reported before any exception handling.
This exception may be expected and handled.
Time Travel Position: CFC0:0
mstscax!CRdpAudioController::OnWaveData+0x281:
00007ff8`5314b9e1 0fb739 movzx edi,word ptr [rcx] ds:800022b7`167fc3cf=????
0:003> k
# Child-SP RetAddr Call Site
00 00000053`1f17ec50 00007ff8`5314b4cf mstscax!CRdpAudioController::OnWaveData+0x281
01 00000053`1f17ed10 00007ff8`531831e3 mstscax!CRdpAudioController::DataArrived+0x72f
02 00000053`1f17ed90 00007ff8`53160070 mstscax!CRdpAudioPlaybackChannelCallback::OnDataReceived+0x433
03 00000053`1f17edf0 00007ff8`5315354a mstscax!CDynVCChannel::InvokeCallback+0x1b0
04 00000053`1f17ee70 00007ff8`53152bf7 mstscax!CDynVCChannel::OnData+0x3aa
05 00000053`1f17ef20 00007ff8`53152a94 mstscax!CDynVCPlugin::OnStaticDataReceived+0x14f
06 00000053`1f17ef90 00007ff8`53165c2e mstscax!CStaticChannelCallback::OnDataReceived+0x24
07 00000053`1f17efd0 00007ff8`531658b6 mstscax!CCommonVCChannel::OpenProcEx+0x31e
08 00000053`1f17f010 00007ff8`53124c71 mstscax!CCommonVCChannel::static_OpenProcEx+0xc6
09 00000053`1f17f060 00007ff8`53124784 mstscax!CChan::ChannelOnPacketReceived+0x179
0a 00000053`1f17f320 00007ff8`53123f4d mstscax!CSL::SLReceivedDataPacket+0x110
0b 00000053`1f17f390 00007ff8`53149a3f mstscax!CSL::OnPacketReceived+0x19d
0c 00000053`1f17f410 00007ff8`53148bcd mstscax!CMCS::MCSRecvData+0x20f
0d 00000053`1f17f490 00007ff8`5314fea8 mstscax!CMCS::OnDataAvailable+0xdd
0e 00000053`1f17f520 00007ff8`53157aaa mstscax!CTSx224Filter::OnDataAvailable+0x138
0f 00000053`1f17f5b0 00007ff8`53178233 mstscax!CTScs51Filter::OnDataAvailable+0xda
10 00000053`1f17f600 00007ff8`5316b296 mstscax!CTSFilterTransport::OnDataAvailable_TransportEvent+0x63
11 00000053`1f17f640 00007ff8`530fcdc0 mstscax!CTSTransportStack::OnDataAvailable+0x106
12 00000053`1f17f6c0 00007ff8`53102db4 mstscax!CTSTcpTransport::AsyncOnReadCompletedAsyncCallback::Invoke+0x60
13 00000053`1f17f6f0 00007ff8`531045d9 mstscax!CTSMsg::Invoke+0xdc
14 00000053`1f17f720 00007ff8`531030c9 mstscax!CTSThread::RunAllQueueEvents+0x219
15 00000053`1f17f7a0 00007ff8`5310415d mstscax!CTSThread::internalMsgPump+0x91
16 00000053`1f17f810 00007ff8`5319951c mstscax!CTSThread::internalThreadMsgLoop+0x14d
17 00000053`1f17fab0 00007ff8`5352e904 mstscax!CTSThread::ThreadMsgLoop+0x1c
18 00000053`1f17faf0 00007ff8`533e1428 mstscax!CRCV::RCVMain+0x170
19 00000053`1f17fb50 00007ff8`533e9b42 mstscax!CTSThread::TSStaticThreadEntry+0x258
1a 00000053`1f17fbb0 00007ff8`d81c7034 mstscax!PAL_System_Win32_ThreadProcWrapper+0x32
1b 00000053`1f17fbe0 00007ff8`d847d0d1 KERNEL32!BaseThreadInitThunk+0x14
1c 00000053`1f17fc10 00000000`00000000 ntdll!RtlUserThreadStart+0x21

```

# Crashing Input


```
- offset -   0 1  2 3  4 5  6 7  8 9  A B  C D  E F  0123456789ABCDEF
0x00000000  5f5f 5f63 6d64 3037 6f02 0000 90d8 deeb  __cmd07o.....
0x00000010  6f02 0000 270c 1a00 ff08 00b2 06a1 0200  o...'.....
0x00000020  5f5f 5f63 6d64 3064 0000 0500 00ff 0800  __cmd0d.....
0x00000030  b206 a102 0044 ac00 00c0 5d00 0004 0010  ....D....]....
0x00000040  0000 0006 a17e 1100 5f5f 5f63 6d64 3037  ....~..__cmd07
0x00000050  6f00 0000 06a1 0200 44ac 0000 0000 0200  o.....D.....
0x00000060  4001 0200 0100 401f 0000 0010 0000 803e  @.....@.....>
0x00000070  5f5f 5f63 6d64 3064 0000 0500 00ff 0800  __cmd0d.....
0x00000080  b206 a102 0044 ac00 00c0 5d00 0004 0010  ....D....]....
0x00000090  0000 0006 a17e 1100  ....~..
```





```
// mstscax!CRdpAudioController::OnWaveData
last_format = pThis->last_format;
format_from_msg = *((unsigned __int16*)msg + 3);

if (last_format != format_from_msg)
{
    /* ... */
    // Treat format change
    v5 = CRdpAudioController::OnNewFormat(pThis, (__int64)format_from_msg);
    /* ... */
    pThis->last_format = last_format = format_from_msg;
}
formats_array = (AUDIO_FORMAT**)pThis->formatArray;
/* ... */
current_wFormatTag = formats_array[last_format]->wFormatTag;
```

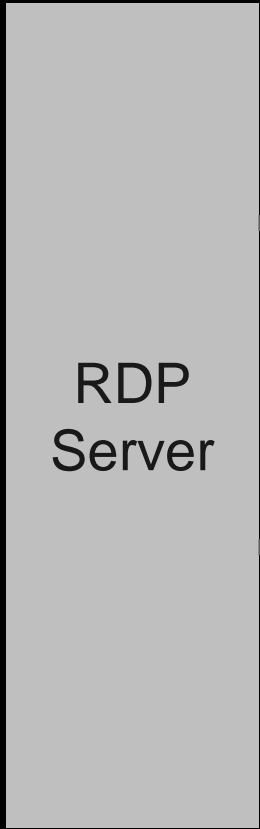




type: formats  
NumFormats: 3

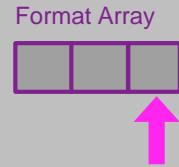
A purple speech bubble containing the text 'type: formats' and 'NumFormats: 3'. A grey arrow points from the RDP Server to this bubble, and another grey arrow points from the bubble to the RDP Client.





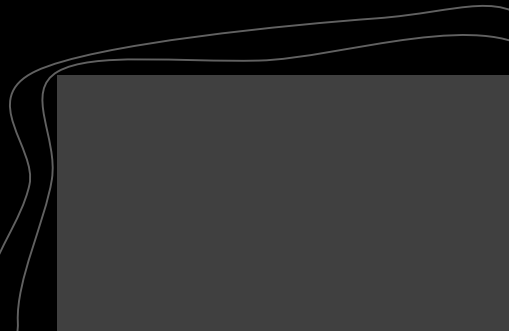
type: formats  
NumFormats: 3

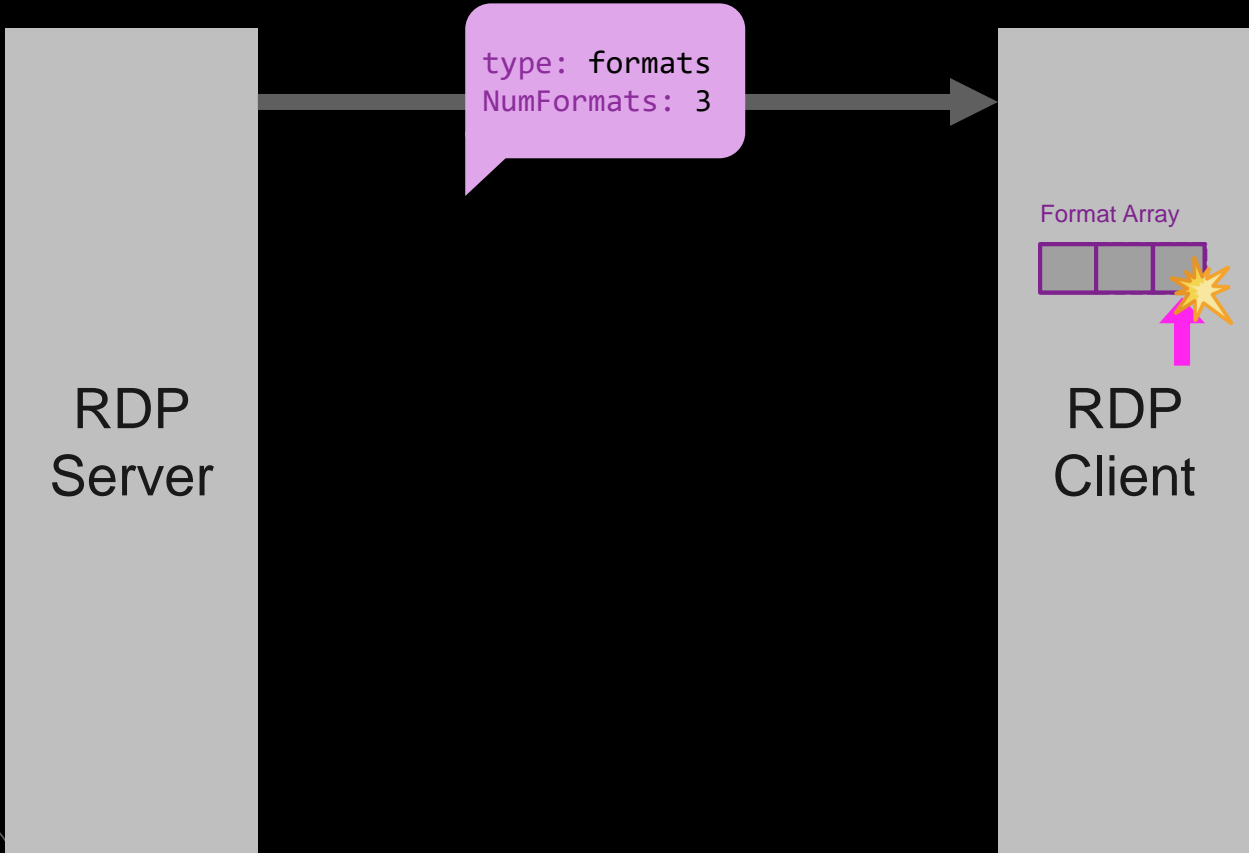
A purple speech bubble containing the text 'type: formats' and 'NumFormats: 3'. An arrow points from this bubble to the RDP Client's Format Array.



type: data  
format: 3  
00 11 22 33  
22 33 44 55

A pink speech bubble containing the text 'type: data', 'format: 3', and two lines of hex data: '00 11 22 33' and '22 33 44 55'. An arrow points from this bubble to the RDP Client.



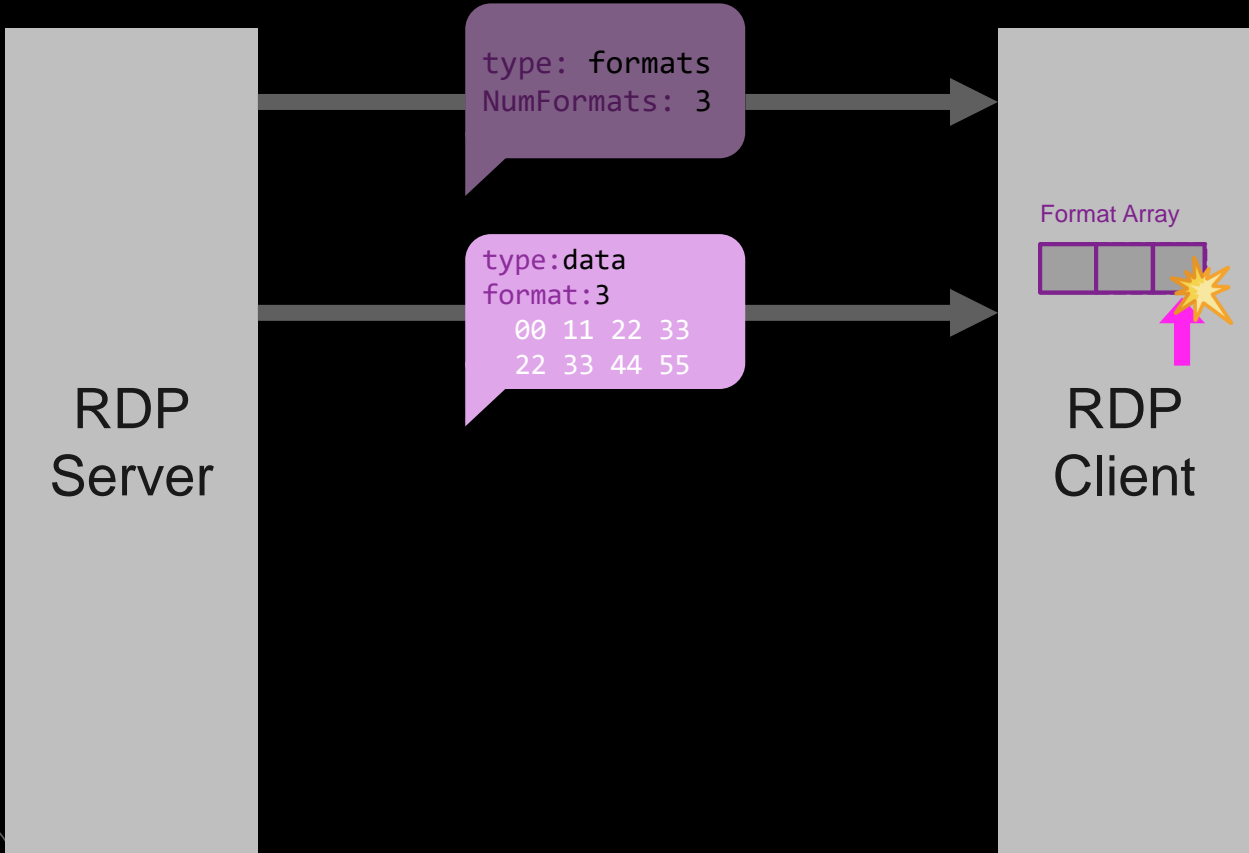


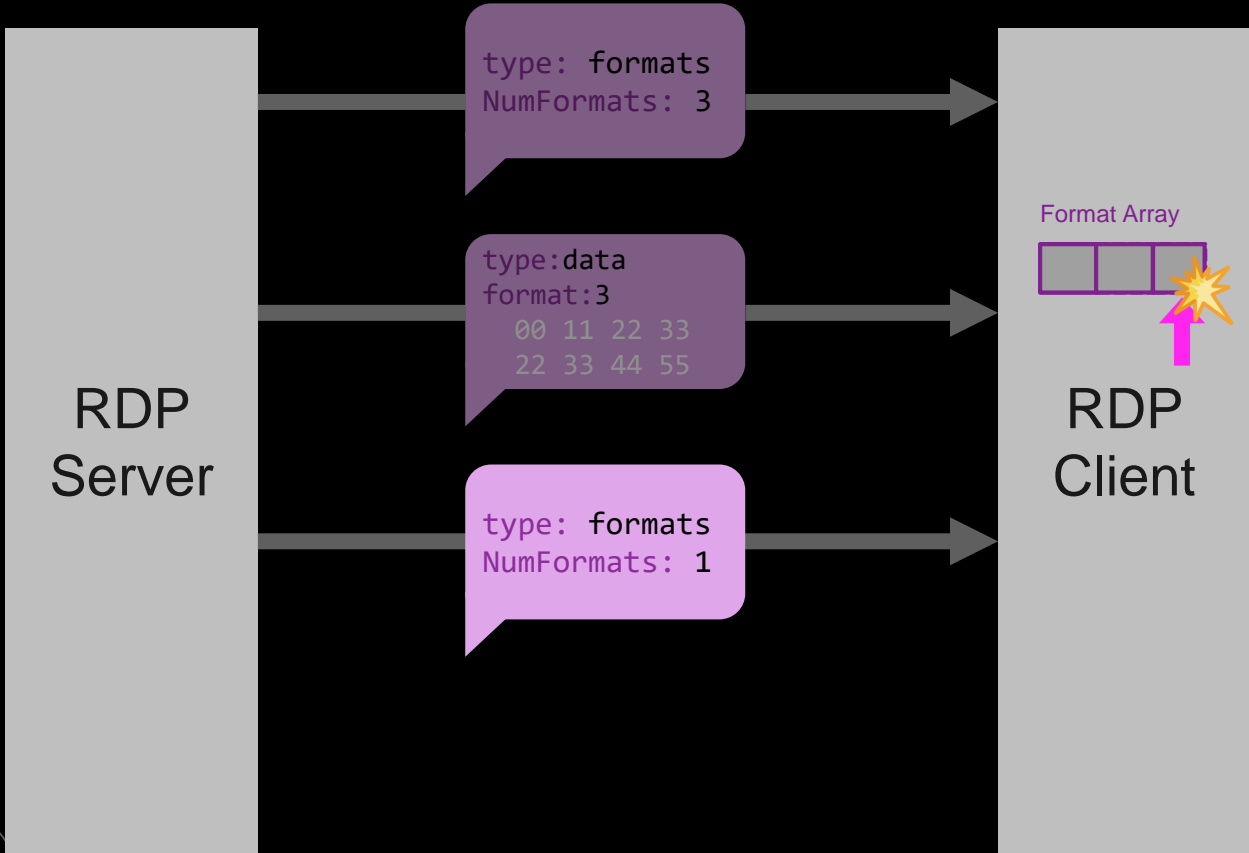
RDP  
Server

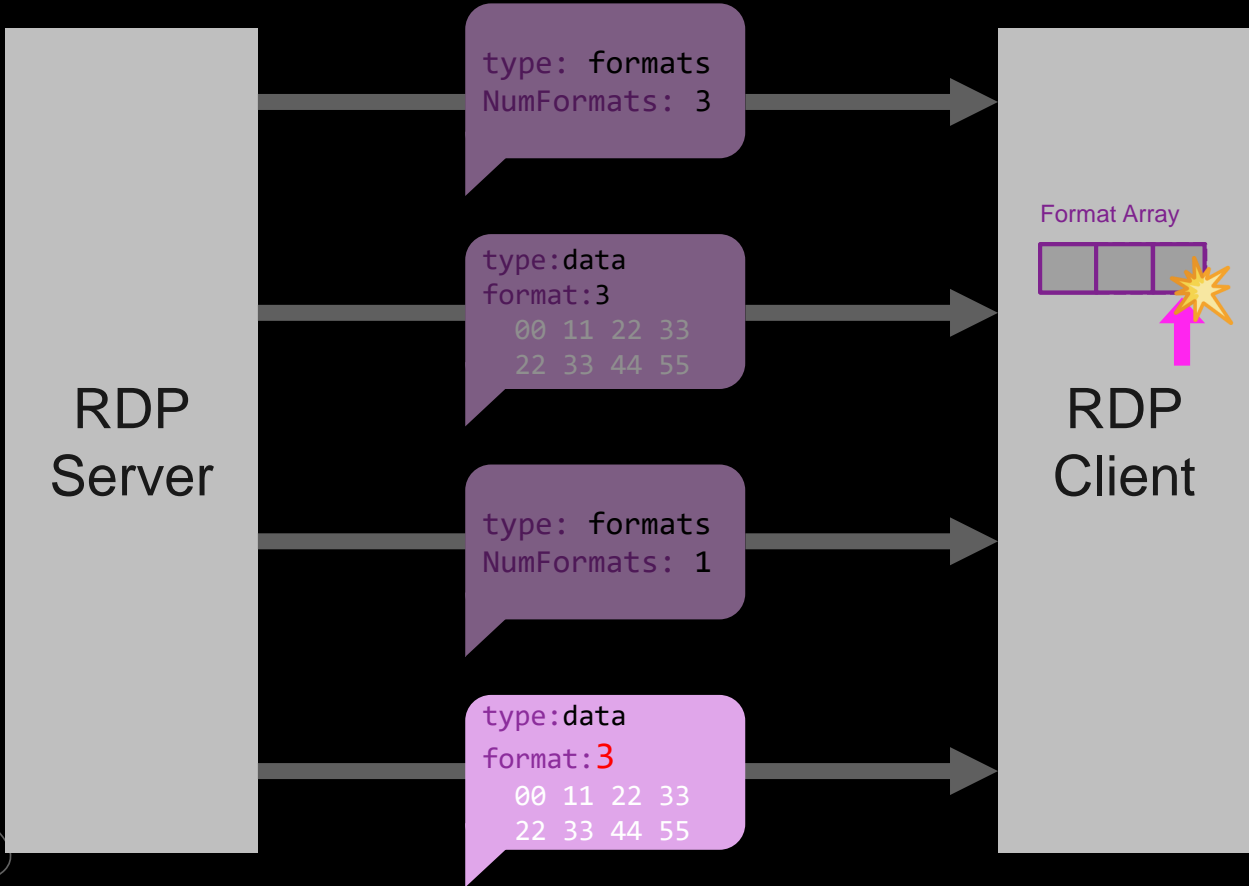
type: formats  
NumFormats: 3


Format Array

RDP  
Client










```
// mstscax!CRdpAudioController::OnWaveData
last_format = pThis->last_format;
format_from_msg = *((unsigned __int16*)msg + 3);

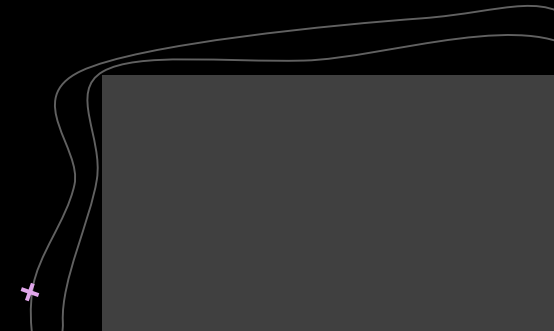
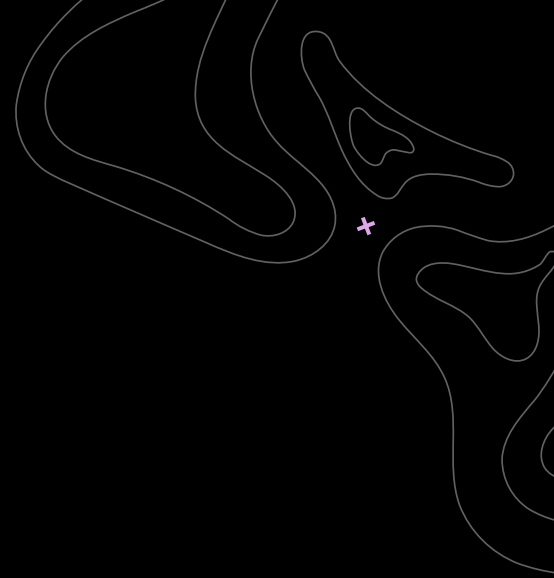
if (last_format != format_from_msg)
{
    /* ... */
    // Treat format change
    v5 = CRdpAudioController::OnNewFormat(pThis, (__int64*)format_from_msg);
    /* ... */
    pThis->last_format = last_format = format_from_msg;
}
formats_array = (AUDIO_FORMAT**)pThis->formatArray;
/* ... */
current_wFormatTag = formats_array[last_format]->wFormatTag;
```





# 04

## Summary





# Future Work

**RDP**

**RPC**

**R\*\*\***

<https://github.com/cyberark/RDPFuzz>

# THANKS



@OrBenPorath  
@ShakReiner



**CYBERARK**<sup>®</sup>

CREDITS: This presentation template was created by **Slidesgo**, including icons by **Flaticon**, infographics & images by **Freepik**

