Reverse RDP Attack

Pwning RDP clients (And more)

@EyalItkin

Who Am I

- **Eyal Itkin**
- Vulnerability Researcher
- cp<r> Check Point Research
- Focus network protocols & embedded devices
- @EyalItkin

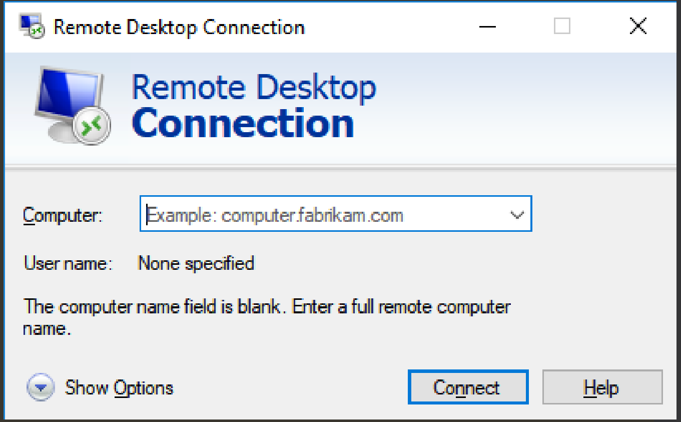
Motivation

- Lazy Lateral Movement
- “Ambush” privileged users
- IT Staff
  - Gain credentials
- Malware Researchers
  - Escape isolated virtual machines

Remote Desktop Protocol (RDP)

"Client" "Server"

- Connects to a remote Windows Machine
  - Remote corporate PC / Server
  - Remote Virtual Machine
- A.K.A. Mstsc
- Uses TCP:3389

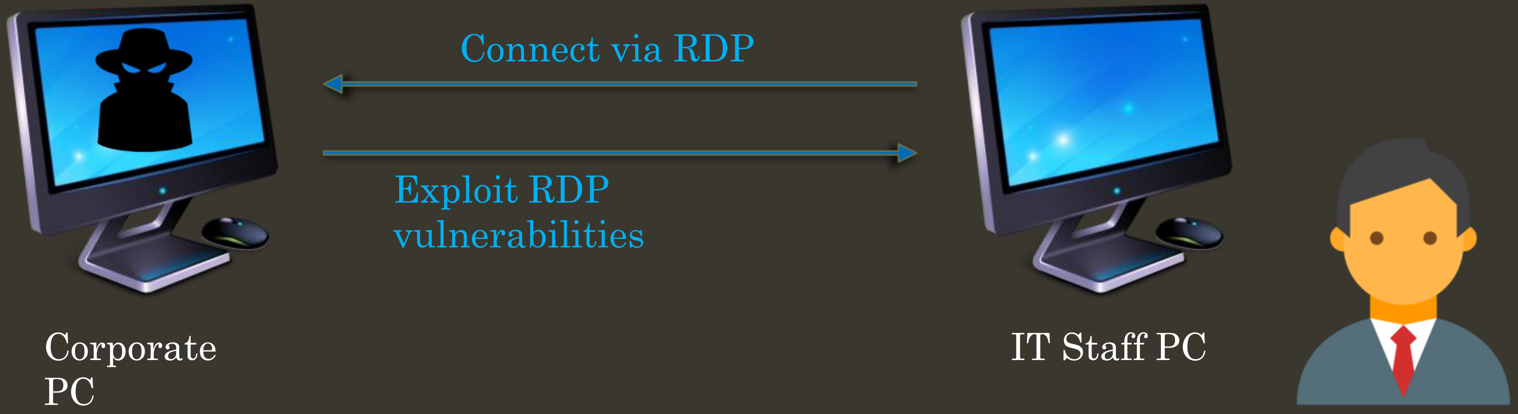


Computer: Example: computer.fabrikam.com

User name: None specified

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

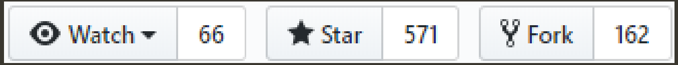
Reverse RDP ?

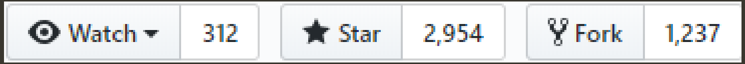


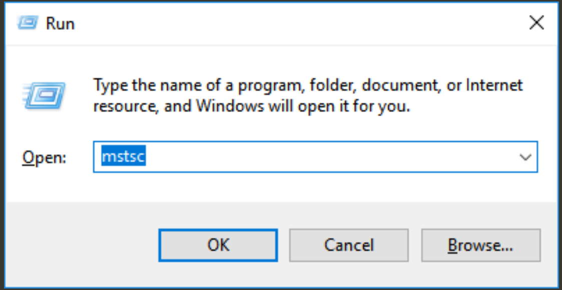
- Collect credentials from the victim
- Attack & Take over the victim’s computer

Our Targets

- Open Source RDP Clients
  - rdesktop
  - FreeRDP
- Microsoft's default client
  - Mstsc.exe







Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.

Open: mstsc

1. Start with the easiest target

- Pick the simplest open source - rdesktop
- Audit the code and learn how RDP works
- Gradually gain confidence
- Move on when scanned all of the code

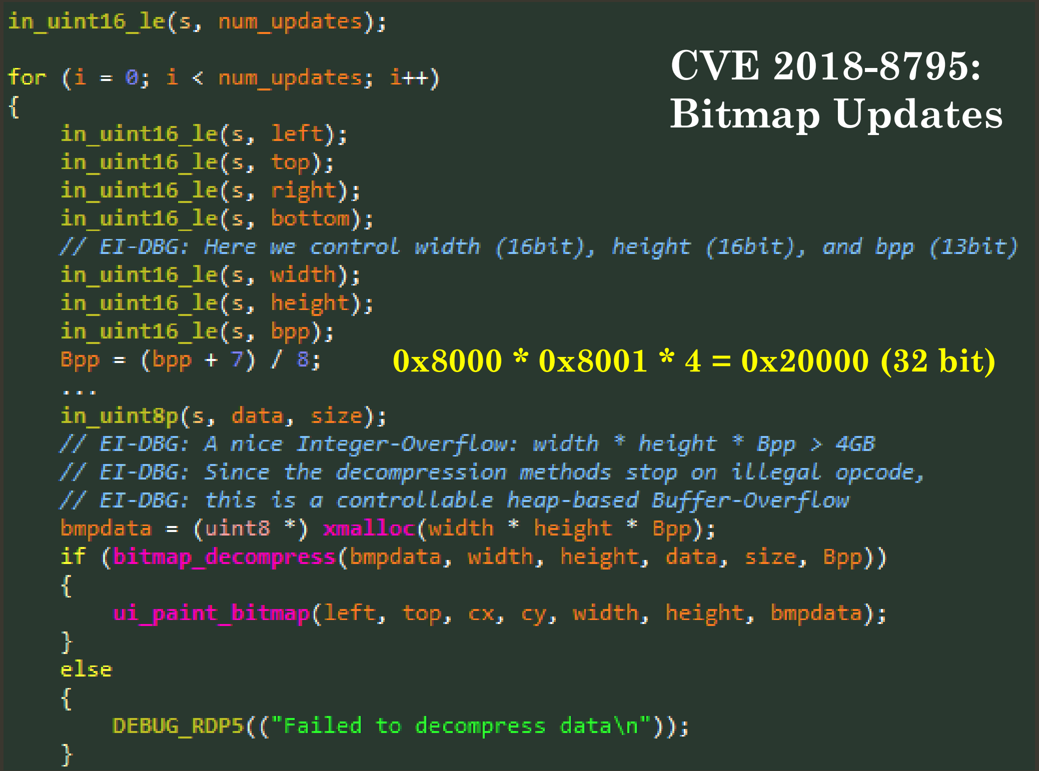
Lessons on RDP

- Protocol consists of logical channels
- Contains multiple authentication methods
- Screen updates are sent using Bitmaps
- Basic Clipboard types are shared

2. Break rdesktop

- Naïve C code with less than minimal checks
  - Almost no checks that minimal input was received
- Found 11 critical vulnerabilities (19 Overall)
- CVEs:
  - CVE 2018-8791 – CVE 2018-8800
  - CVE 2018-20174 – CVE 2018-20182

3. Find complicated features



CVE 2018-8795:
Bitmap Updates

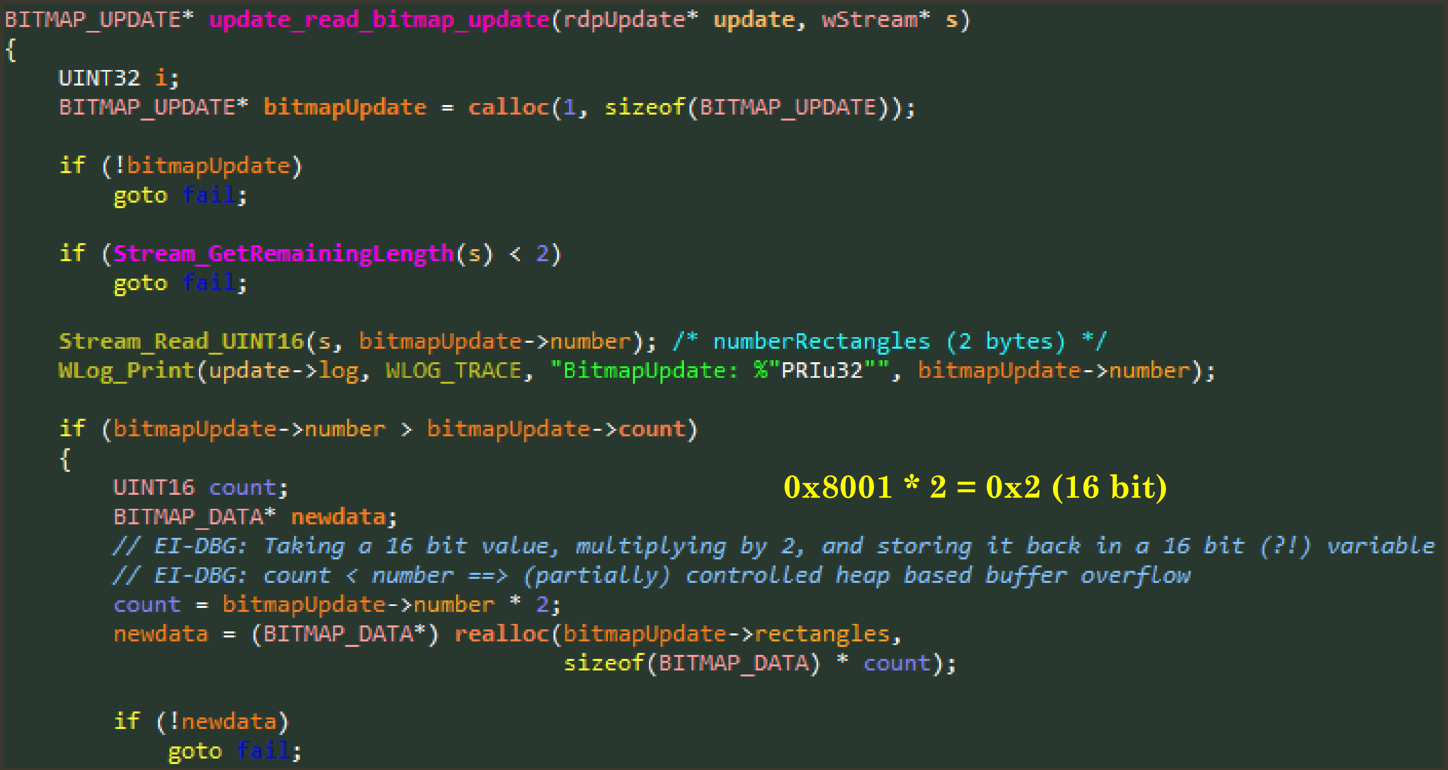
```
in_uint16_le(s, num_updates);

for (i = 0; i < num_updates; i++)
{
    in_uint16_le(s, left);
    in_uint16_le(s, top);
    in_uint16_le(s, right);
    in_uint16_le(s, bottom);
    // EI-DBG: Here we control width (16bit), height (16bit), and bpp (13bit)
    in_uint16_le(s, width);
    in_uint16_le(s, height);
    in_uint16_le(s, bpp);
    Bpp = (bpp + 7) / 8;
    ...
    in_uint8p(s, data, size);
    // EI-DBG: A nice Integer-Overflow: width * height * Bpp > 4GB
    // EI-DBG: Since the decompression methods stop on illegal opcode,
    // EI-DBG: this is a controllable heap-based Buffer-Overflow
    bmpdata = (uint8 *) xmalloc(width * height * Bpp);
    if (bitmap_decompress(bmpdata, width, height, data, size, Bpp))
    {
        ui_paint_bitmap(left, top, cx, cy, width, height, bmpdata);
    }
    else
    {
        DEBUG_RDP5(("Failed to decompress data\n"));
    }
```

0x8000 * 0x8001 * 4 = 0x20000 (32 bit)

4. Break FreeRDP

- The C code looks better
  - Still has a few cracks if we look deep enough
  - Again, vulnerable to Bitmap parsing
- Found 5 critical vulnerabilities (6 Overall)
- CVEs:
  - CVE 2018-8784 – CVE 2018-8789

RCE Test Case: CVE 2018-8786

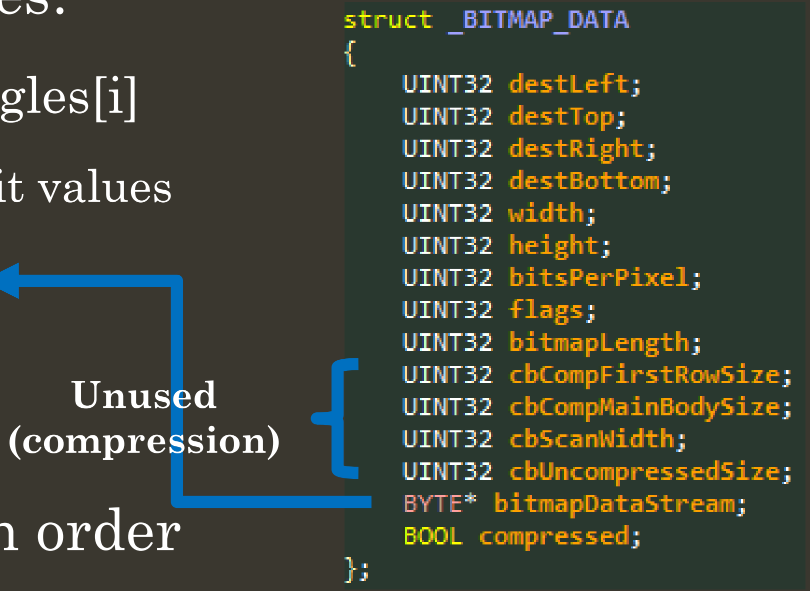


```
BITMAP_UPDATE* update_read_bitmap_update(rdpUpdate* update, wStream* s)
{
    UINT32 i;
    BITMAP_UPDATE* bitmapUpdate = calloc(1, sizeof(BITMAP_UPDATE));

    if (!bitmapUpdate)
        goto fail;

    if (Stream_GetRemainingLength(s) < 2)
        goto fail;

    Stream_Read_UINT16(s, bitmapUpdate->number); /* numberRectangles (2 bytes) */
    WLog_Print(update->log, WLOG_TRACE, "BitmapUpdate: %"PRIu32"", bitmapUpdate->number);

    if (bitmapUpdate->number > bitmapUpdate->count)
    {
        UINT16 count;
        BITMAP_DATA* newdata;
        // EI-DBG: Taking a 16 bit value, multiplying by 2, and storing it back in a 16 bit (?!) variable
        // EI-DBG: count < number ==> (partially) controlled heap based buffer overflow
        count = bitmapUpdate->number * 2;
        newdata = (BITMAP_DATA*) realloc(bitmapUpdate->rectangles,
                                         sizeof(BITMAP_DATA) * count);

        if (!newdata)
            goto fail;
```

0x8001 * 2 = 0x2 (16 bit)

CVE 2018-8786: Heap Shaping

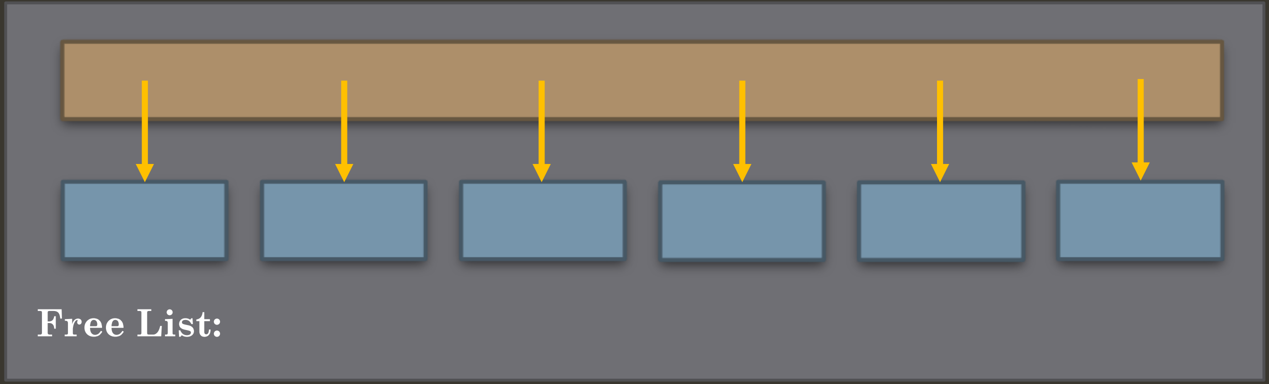
- For i in numberRectangles:
  - Parse rectangle into rectangles[i]
    - 16 bit values, stored as 32 bit values
    - Fully controlled allocation
- <processing>
- Rectangles are free()ed in order



```
struct _BITMAP_DATA
{
    UINT32 destLeft;
    UINT32 destTop;
    UINT32 destRight;
    UINT32 destBottom;
    UINT32 width;
    UINT32 height;
    UINT32 bitsPerPixel;
    UINT32 flags;
    UINT32 bitmapLength;
    UINT32 cbCompFirstRowSize;
    UINT32 cbCompMainBodySize;
    UINT32 cbScanWidth;
    UINT32 cbUncompressedSize;
    BYTE* bitmapDataStream;
    BOOL compressed;
};
```

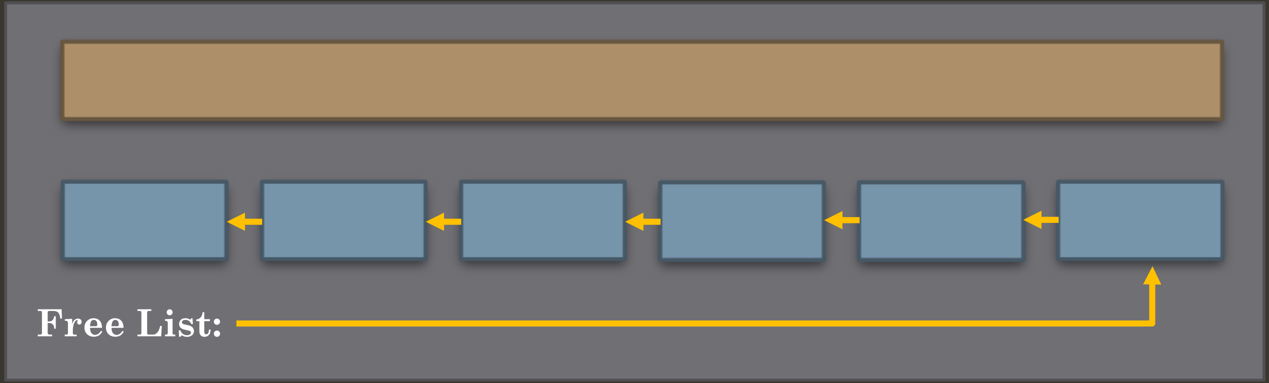
CVE 2018-8786: Heap Shaping

1. Allocate space for many rectangles with bitmap_length of the same size (0x60)



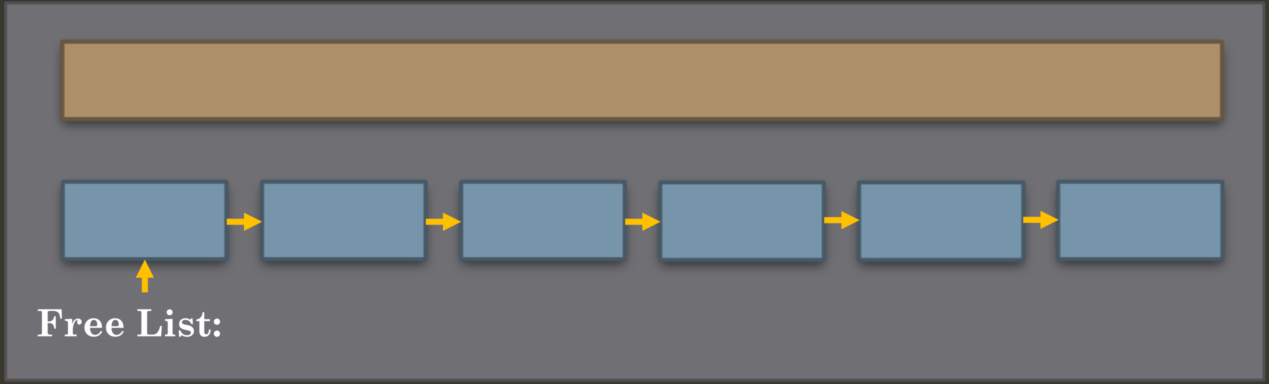
CVE 2018-8786: Heap Shaping

2. Upon free, the order will be flipped (LIFO)



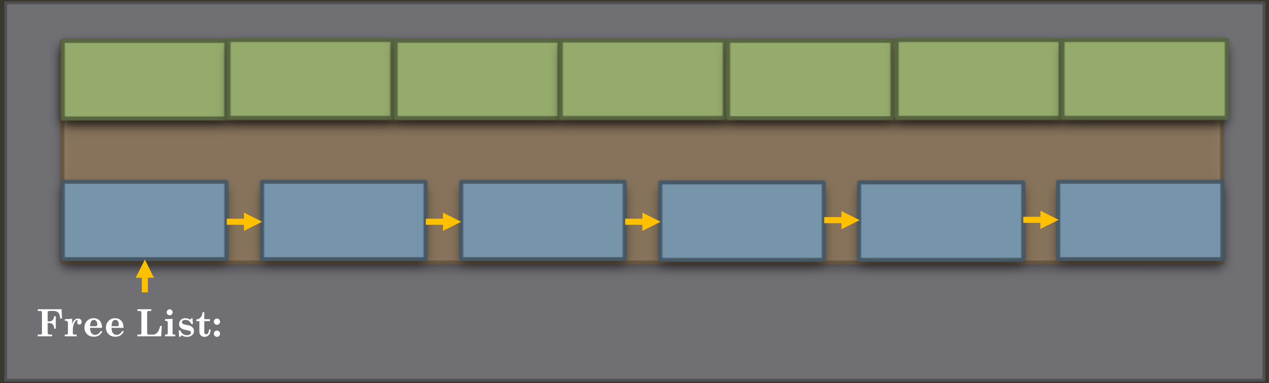
CVE 2018-8786: Heap Shaping

3. Allocate (and free) the space again to flip the memory back in the Free List

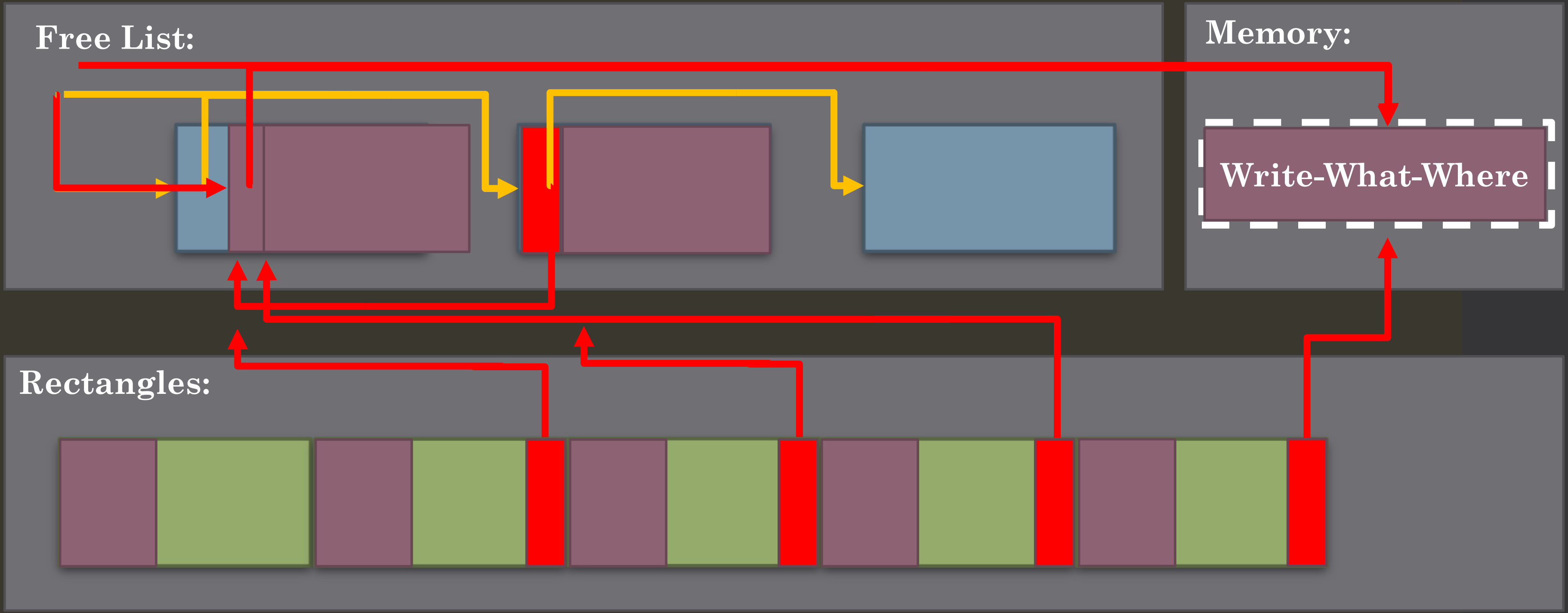


CVE 2018-8786: Heap Shaping

4. Trigger the vulnerability and write empty rects over the allocated region



Heap Shaping – Zooming in



5. Break Mstsc.exe ?

- PoCs from previous targets failed ☹
- The code is robust
  - Smart buffers check for parsing errors
- Includes many more features
- Where should we go now?

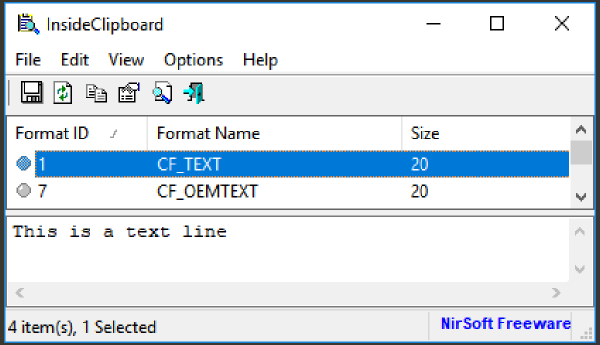
Clip

Back to the Dra[illegible]ng board

- Until now, the clipboard shared text:
  - CF_TEXT
  - CF_UNICODETEXT
- It seems like Microsoft supports many more formats now
- Let’s dig into the clipboard

Clipboard 101

- A kernel data structure that stores data
  - One clipboard per session
  - Shared between processes
- Stores data (blobs) by ID / Name
- **Caution:** Clipboard data is not trusted. Parse the data carefully before using it in your application.



Format ID	Format Name	Size
1	CF_TEXT	20
7	CF_OEMTEXT	20

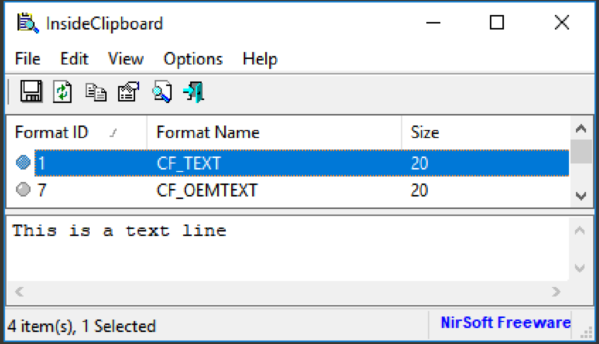
```
This is a text line
```

4 item(s), 1 Selected

NirSoft Freeware

Clipboard 101

- A kernel data structure that stores data
  - One clipboard per session
  - Shared between processes
- Stores data (blobs) by ID / Name
- "**Caution:** Clipboard data is not trusted. Parse the data carefully before using it in your application." (MSDN)



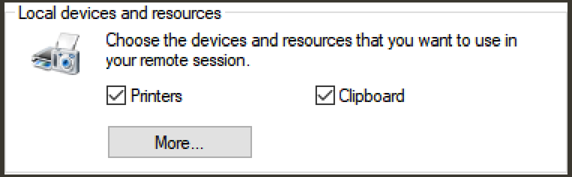
Format ID	Format Name	Size
1	CF_TEXT	20
7	CF_OEMTEXT	20

```
This is a text line
```

4 item(s), 1 Selected

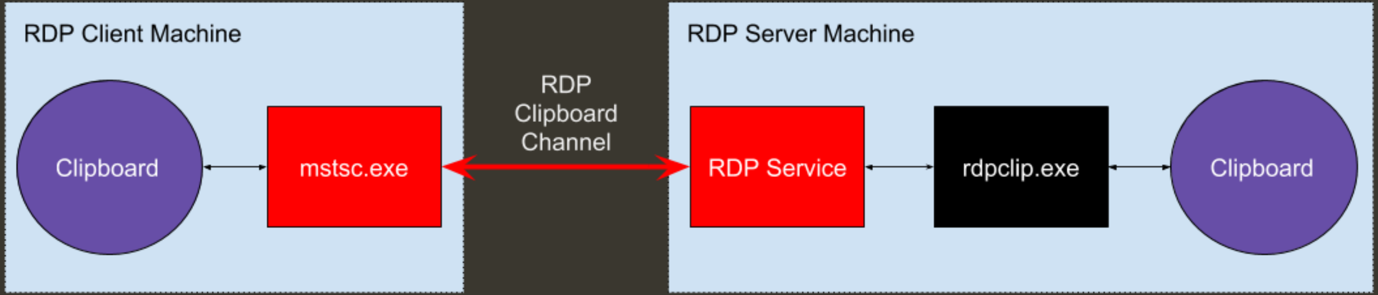
Clipboard Over RDP

- The Clipboard is a shared resource
  - Shared by default


Local devices and resources

Choose the devices and resources that you want to use in your remote session.

[✓] Printers [✓] Clipboard

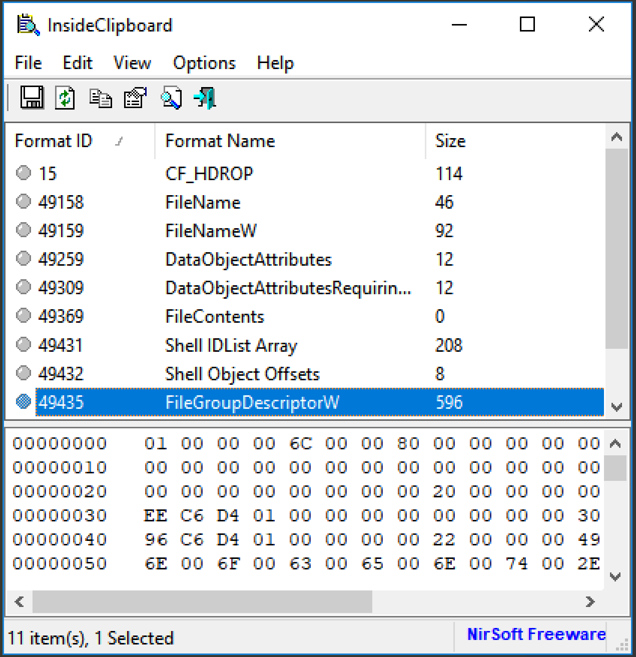


Clipboard Over RDP

- Black Lists instead of White Lists
  - Everything is synchronized automatically
  - Some formats are discarded (by ID or by Name)
  - Some formats have special handling
- Content is subject to “delay rendering”

Drag & Drop

- Transfer files using "Drag & Drop"
- Copying files uses multiple formats
  - CF_HDROP – lists the file names
  - FileGroupDescriptorW – full metadata
  - Many more…
- Let's see how it works in practice

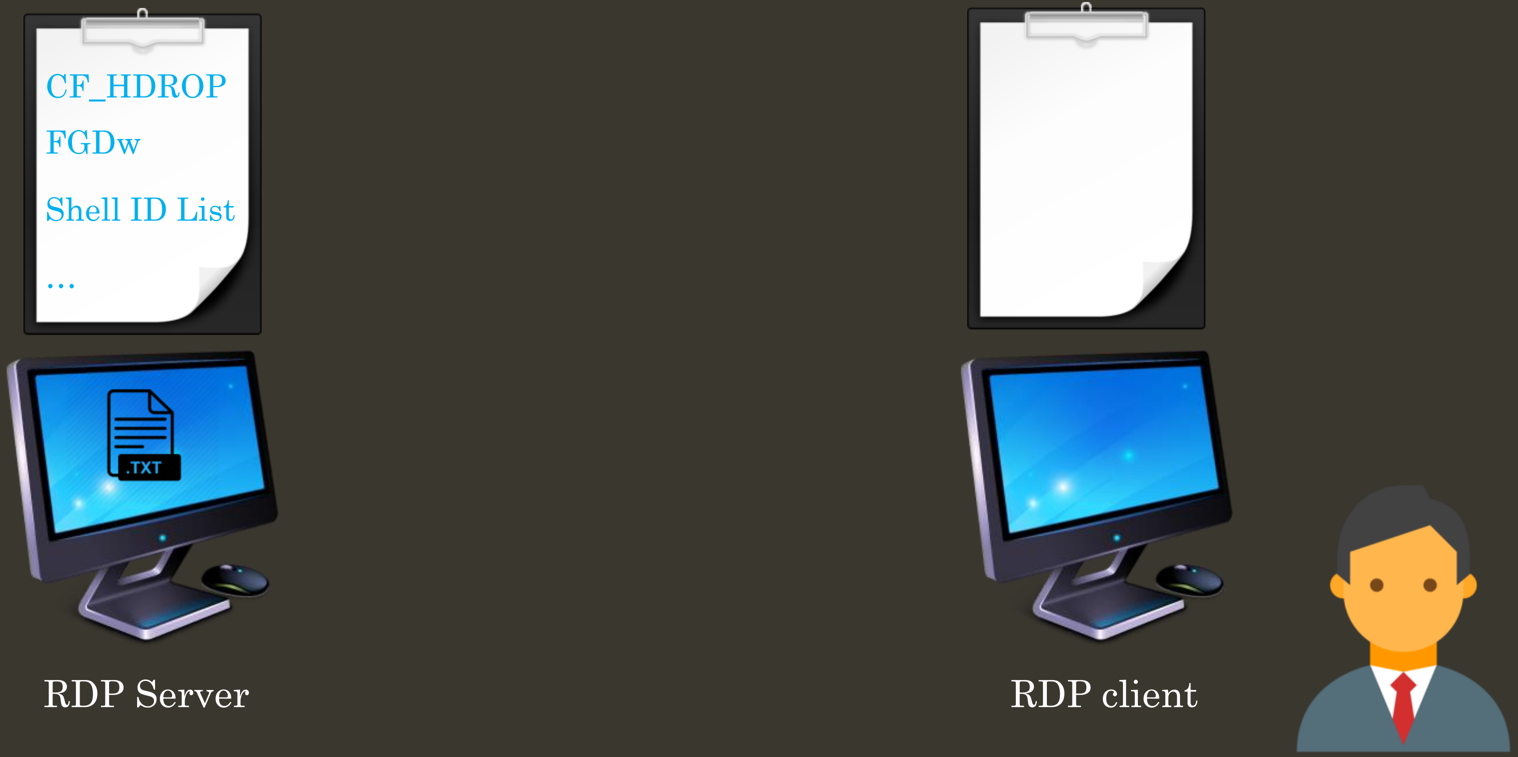


Format ID	Format Name	Size
15	CF_HDROP	114
49158	FileName	46
49159	FileNameW	92
49259	DataObjectAttributes	12
49309	DataObjectAttributesRequirin...	12
49369	FileContents	0
49431	Shell IDList Array	208
49432	Shell Object Offsets	8
49435	FileGroupDescriptorW	596

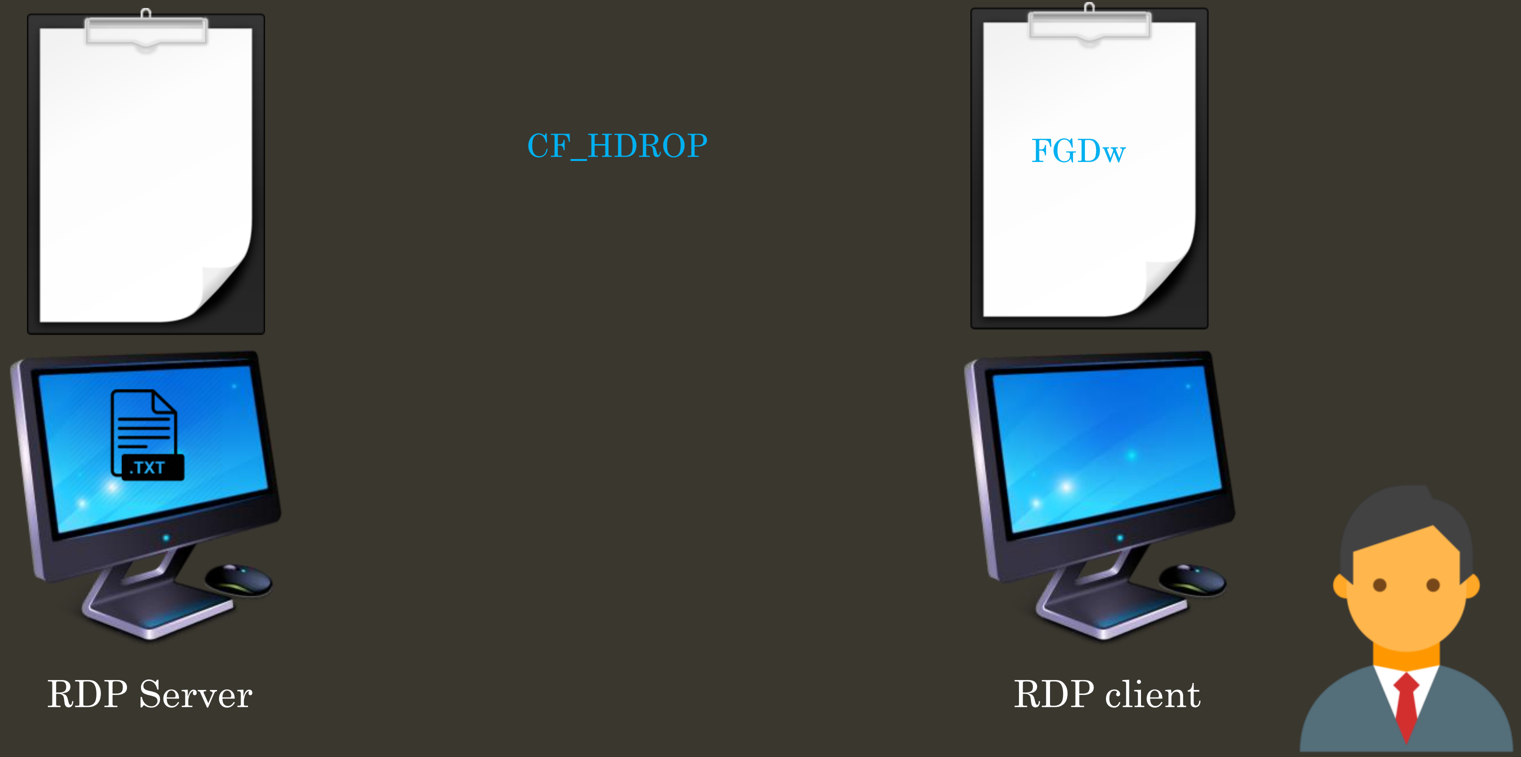
```
00000000   01 00 00 00 6C 00 00 80 00 00 00 00 00
00000010   00 00 00 00 00 00 00 00 00 00 00 00 00
00000020   00 00 00 00 00 00 00 00 20 00 00 00 00
00000030   EE C6 D4 01 00 00 00 00 00 00 00 00 30
00000040   96 C6 D4 01 00 00 00 00 22 00 00 00 49
00000050   6E 00 6F 00 63 00 65 00 6E 00 74 00 2E
```

11 item(s), 1 Selected   NirSoft Freeware

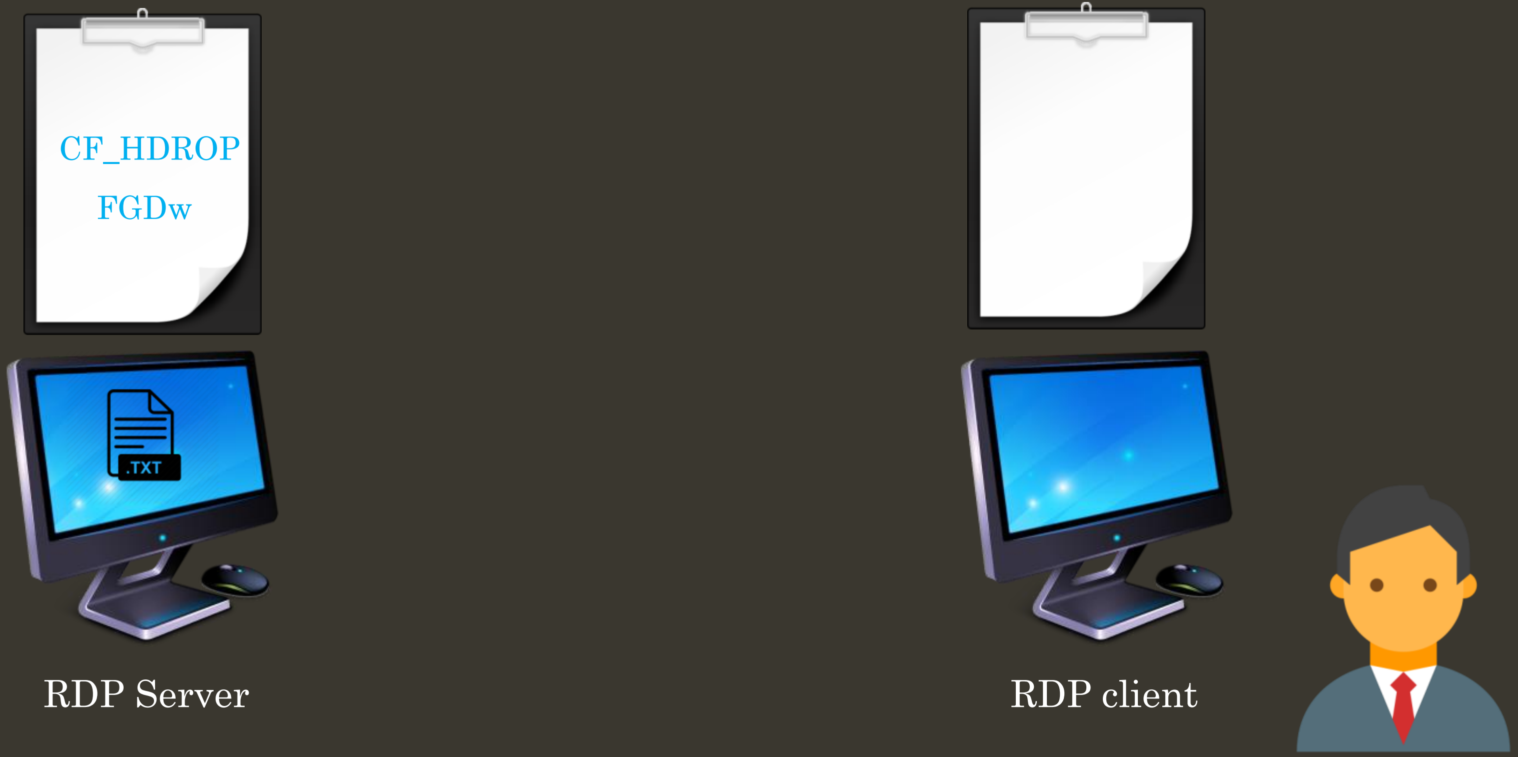
Drag & Drop In Action – Ctrl+C



Drag & Drop In Action – Ctrl+V



Drag & Drop In Action – Ctrl+V

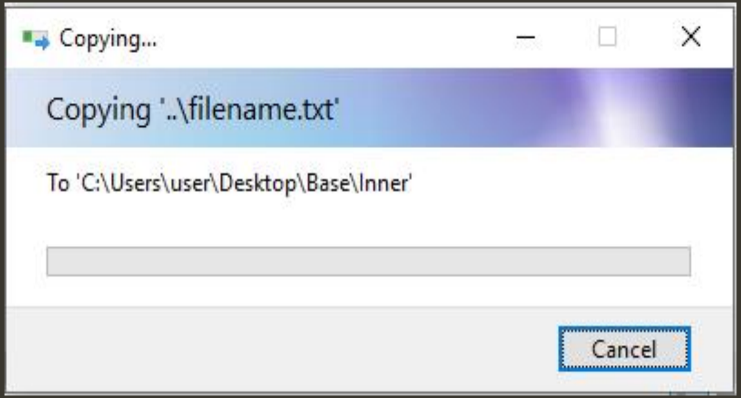


FileGroupDescriptorW

- Proprietary blob structure
- Contains a list of file records
  - Meta data (timestamps)
  - File path – filename / absolute path
- Client passes it directly to the clipboard

Path Canonicalization

@GullOmer: “try to find where they sanitize the path”


Copying...

Copying '..\filename.txt'

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

Path Traversal Over RDP

- When using “Copy & Paste” a malicious server can:
  - Drop arbitrary files to arbitrary locations
- Drop your script in the Startup and that’s it

Path Traversal Over RDP - Video

Recycle Bin
Notepad++
RW-Everyth...
Microsoft Edge
server.rdp
Windows 10 Enterprise N Insider Preview
Evaluation copy. Build 18252.rs_prerelease.180928-1410
Type here to search
ENG
02:53
29/10/18

Taking it one step further

- The clipboards are **fully** synchronized
  - Ctrl+C updates the clipboard
  - Each update sends a CLIPRDR_FROMAT_LIST
  - The receiver updates his clipboard accordingly
- What does it mean?

Scenario #1 - Eavesdropping

- When the client copies a password we get it too
- This is a **feature** of the synced clipboard
- We know in advance when the client is going to copy a file on **his** computer

Scenario #2 – Ctrl+V Only Attack

- Once again, ambush the client
- When he copies a file, start the attack
- Send an update message and switch his clipboard to a malicious FGDw
- His Ctrl+V will trigger the path traversal

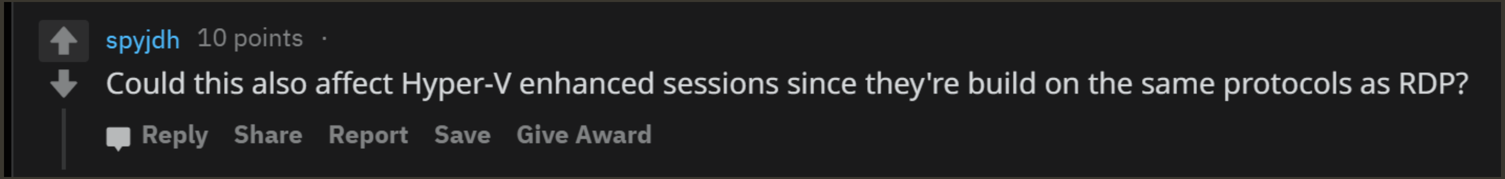
Responsible Disclosure

- rdesktop: patched everything – 19 CVEs
- FreeRDP: patched everything – 6 CVEs
- Microsoft:

The End ?

Reddit to the rescue

- Publication was posted to /r/netsec
- One comment asked:



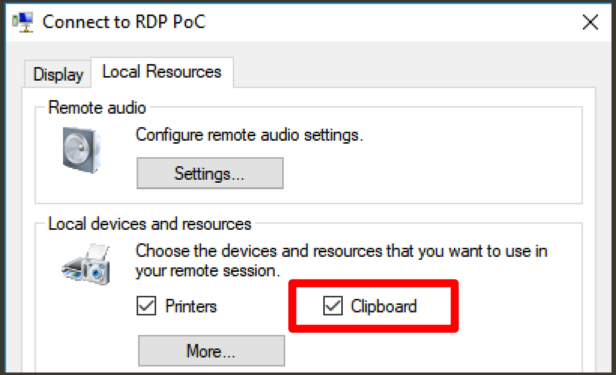
spyjdh 10 points ·

Could this also affect Hyper-V enhanced sessions since they're build on the same protocols as RDP?

- Excellent question

Hyper-V

- Never used it till now
- Installed a Hyper-V machine, and



Connect to RDP PoC

Remote audio

Configure remote audio settings.

Local devices and resources

Choose the devices and resources that you want to use in your remote session.

[✓] Printers
[✓] Clipboard

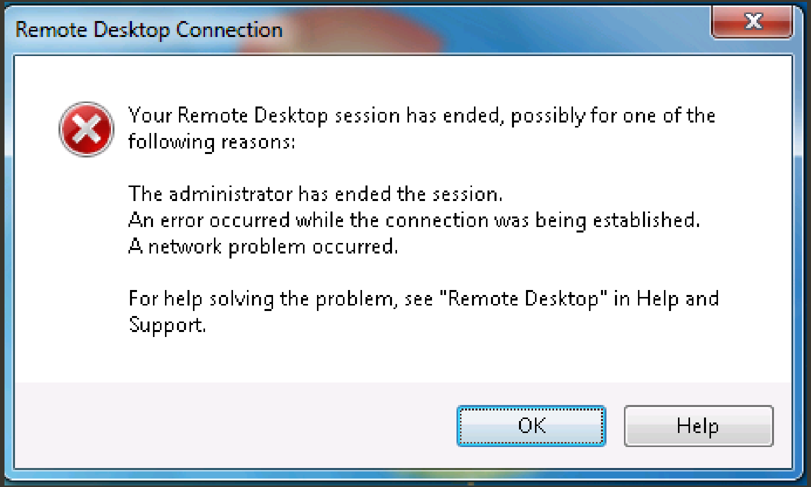
Hyper-V? RDP!

- Connection to the VM is transferred over RDP!
- The PoC we submitted to MS worked on the first attempt
- We found a Guest-to-Host VM Escape ☺

Hyper-V Demo

- **Live Demo**

That’s all folks



Your Remote Desktop session has ended, possibly for one of the following reasons:

The administrator has ended the session.
An error occurred while the connection was being established.
A network problem occurred.

For help solving the problem, see "Remote Desktop" in Help and Support.

@EyalItkin

eyalit@checkpoint.com

https://research.checkpoint.com/reverse-rdp-attack-code-execution-on-rdp-clients/