# WinregMITM

## Remotely Injecting Code into the Windows Registry

# About me

- Santiago Hernández
- Security Researcher at 11paths
- ToorCon SanDiego, Navaja Negra, Black Hat Europe…

@santiagohramos

# Windows Remote Registry Protocol

- Remote management of hierarchical data stores: Windows Registry
- Client/Server protocol
- Dependent on RPC and SMB for packet transport

| Parameter | Value | Reference |
|---|---|---|
| RPC Interface UUID | {338CD001-2244-31F1-AAAA-900038001003} | [C706] |
| Pipe name | \PIPE\winreg | [MS-SMB] |

# Relationship with Other Protocols

IP/TCP/NetBIOS/SMB2/DCE-RPC/WINREG

```
> Frame 140477: 358 bytes on wire (2864 bits), 358 bytes
> Ethernet II, Src: PcsCompu_81:02:b8 (08:00:27:81:02:b8
> Internet Protocol Version 4, Src: 192.168.1.114, Dst:
> Transmission Control Protocol, Src Port: 49884, Dst Po
> NetBIOS Session Service
> SMB2 (Server Message Block Protocol version 2)
> Distributed Computing Environment / Remote Procedure C
> Remote Registry Service, OpenKey
```

# How Does a Session Work?

1. The client initiates a connection request to the server.
2. The server authorizes the request.
3. The client requests the opening of a key on the server.
4. The server responds with an RPC context handle that references the key.

# How Does a Session Work?

5. The client uses the handle to operate on the key.
6. The client sends the desired operation along with the associated parameters.
7. The server responds to the request with the requested information.

# RPC Methods

| Operation | Opnum |
|---|---|
| **BaseRegCreateKey** | **6** |
| BaseRegDeleteKey | 7 |
| **BaseRegOpenKey** | **15** |
| BaseRegDeleteValue | 8 |
| **BaseRegSetValue** | **22** |
| BaseRegFlushKey | 11 |
| … | |

# BaseRegOpenKey

The server opens a key and sends the handle to the client.

```
WINREG     222 GetVersion request
WINREG     202 GetVersion response
WINREG     362 OpenKey request, S-1-5-21-2958448281-842647196-920211908-1000\AppEvents
WINREG     218 OpenKey response
```

```
error_status_t BaseRegOpenKey(
[in] RPC_HKEY hKey,
[in] PRRP_UNICODE_STRING lpSubKey,
[in] DWORD dwOptions,
[in] REGSAM samDesired,
[out] PRPC_HKEY phkResult
);
```

# BaseRegSetValue

The server sets the data sent for a specific value of a registry key.

```
WINREG      274 QueryValue request
WINREG      226 QueryValue response
WINREG      286 QueryValue request
WINREG      250 QueryValue response
WINREG      270 SetValue request,
WINREG      198 SetValue response
```

```
error_status_t BaseRegSetValue(
[in] RPC_HKEY hKey,
[in] PRRP_UNICODE_STRING lpValueName,
[in] DWORD dwType,
[in, size_is(cbData)] LPBYTE lpData,
[in] DWORD cbData
);
```

# BaseRegCreateKey

The server creates the key and returns a handle that references it.

```
WINREG      362 OpenKey request, S-1-5-21-2958448281-842647196-920211908-1000\AppEvents
WINREG      218 OpenKey response
WINREG      222 GetVersion request
WINREG      202 GetVersion response
WINREG      282 OpenKey request, Clave nueva #1
WINREG      218 OpenKey response, Error: WERR_BADFILE
WINREG      222 GetVersion request
WINREG      202 GetVersion response
WINREG      298 CreateKey request, Clave nueva #1
```

```
error_status_t BaseRegCreateKey(
 [in] RPC_HKEY hKey,
 [in] PRRP_UNICODE_STRING lpSubKey,
 [in] PRRP_UNICODE_STRING lpClass,
 …
```

# Protocol Integrity

## Opening of HKEY_CURRENT_USER

```
SMB2     210 Create Response File: winreg
DCERPC   334 Bind: call_id: 2, Fragment: Single, 2 context items: WINREG V1.0 (32bit NDR), WINREG V1.0 (6cb71c2c-9812-4540-0300-000000000000), NTLMSSP_NEGOTIATE
SMB2     138 Write Response
SMB2     171 Read Request Len:1024 Off:0 File: winreg
DCERPC   476 Bind_ack: call_id: 2, Fragment: Single, max_xmit: 4280 max_recv: 4280, 2 results: Acceptance, Negotiate ACK, NTLMSSP_CHALLENGE
DCERPC   718 AUTH3: call_id: 2, Fragment: Single, NTLMSSP_AUTH, User: DESKTOP-FS8NLP3\user
SMB2     138 Write Response
WINREG   290 OpenHKU request
DCERPC   202 Fault: call_id: 2, Fragment: Single, Ctx: 0, status: nca_s_fault_access_denied
SMB2     146 Close Request File: winreg
SMB2     182 Close Response
SMB2     190 Create Request File: winreg
SMB2     210 Create Response File: winreg
DCERPC   286 Bind: call_id: 2, Fragment: Single, 2 context items: WINREG V1.0 (32bit NDR), WINREG V1.0 (6cb71c2c-9812-4540-0300-000000000000)
SMB2     138 Write Response
SMB2     171 Read Request Len:1024 Off:0 File: winreg
DCERPC   230 Bind_ack: call_id: 2, Fragment: Single, max_xmit: 4280 max_recv: 4280, 2 results: Acceptance, Negotiate ACK
WINREG   214 OpenHKU request
WINREG   218 OpenHKU response
WINREG   222 GetVersion request
WINREG   202 GetVersion response
WINREG   342 OpenKey request, S-1-5-21-3453662643-653229605-3378087727-1001
```

# Security Context

https://msdn.microsoft.com/en-us/library/cc243713.aspx

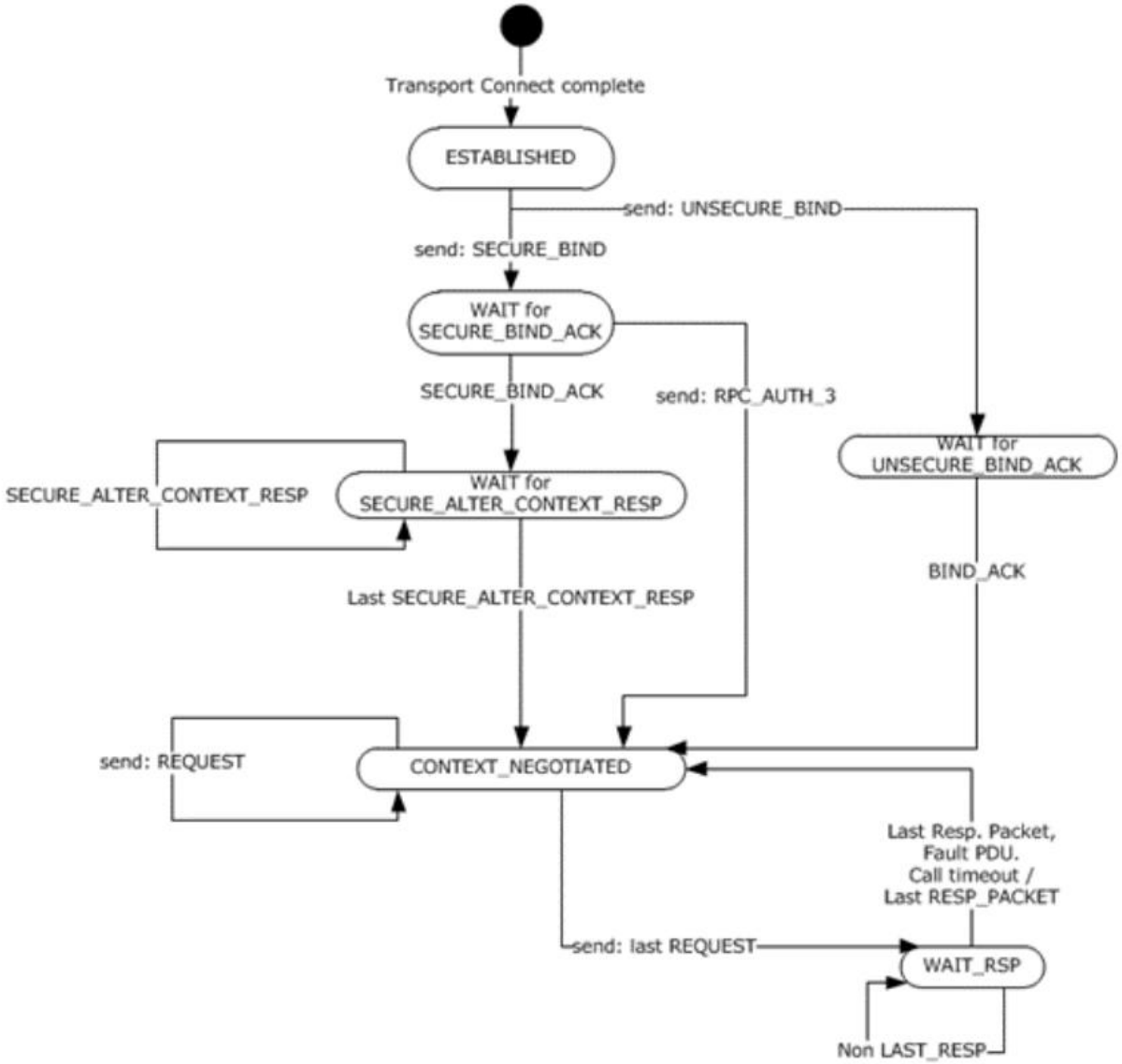
**security context**: An abstract data structure that contains authorization information for a particular security principal in the form of a Token/Authorization Context (see [MS-DTYP] section 2.5.2). A server uses the authorization information in a security context to check access to requested resources. A security context also contains a key identifier that associates mutually established cryptographic keys, along with other information needed to perform secure communication with another security principal.

- To make secure RPC calls, a security context must be established.
- The scope is the connection.
- Bind/Bind_ack message exchange with authorization info between the RPC client and server.
- If there is an error: Fault PDU and connection closure.
- Once the context is established, it can be used by RPC and other protocols to make authorization decisions.

https://msdn.microsoft.com/en-us/library/cc243630.aspx#gt_8a7f6700-8311-45bc-af10-82e10accd331

# Security Context

https://msdn.microsoft.com/en-us/library/cc243724.aspx



# Security Provider

**security provider**: A pluggable security module that is specified by the protocol layer above the remote procedure call (RPC) layer, and will cause the RPC layer to use this module to secure messages in a communication session with the server. The security provider is sometimes referred to as an authentication service. For more information, see [C706] and [MS-RPCE].

- The client begins by requesting an authorization token from the security provider (NTLM).
- This token represents the client's identity and can be used to make authorization decisions.
- **Additionally, RPC can use the security context to create a logical chain of information protected against tampering and disclosure. .**

https://msdn.microsoft.com/en-us/library/cc243630.aspx#gt_05fd3925-0672-4f24-9dd9-2b9d441eb333

# Authentication Levels

The amount of protection applied depends on the authentication level for the security context requested by the client when the security context is created. The authentication level is applied in two dimensions:

| Protection Level | Description |
|---|---|
| dce_c_auth_level_none (1) | No protection is established. |
| Dce_c_auth_level_connect (2) | Provides verification of the client and server identities, protects against replay attacks, but no measures are applied per PDU (Protocol Data Unit). |
| Dce_c_auth_level_call (3) | Provides integrity for the first fragment of each call. |
| Dce_c_auth_level_pkt (4) | Provides protection against replay and out-of-order attacks at the PDU level but does not protect against modification at the PDU level. |
| Dce_c_auth_level_pkt_integrity (5) | Provides integrity (signing) at the PDU level. |
| Dce_c_auth_level_pkt_privacy (6) | Provides integrity (signing) and confidentiality (encryption) at the PDU level. |

http://pubs.opengroup.org/onlinepubs/9629399/chap13.htm

# Protocol Integrity

## Opening of HKEY_CURRENT_USER

```
SMB2     210 Create Response File: winreg
DCERPC   334 Bind: call_id: 2, Fragment: Single, 2 context items: WINREG V1.0 (32bit NDR), WINREG V1.0 (6cb71c2c-9812-4540-0300-000000000000), NTLMSSP_NEGOTIATE
SMB2     138 Write Response
SMB2     171 Read Request Len:1024 Off:0 File: winreg
DCERPC   476 Bind_ack: call_id: 2, Fragment: Single, max_xmit: 4280 max_recv: 4280, 2 results: Acceptance, Negotiate ACK, NTLMSSP_CHALLENGE
DCERPC   718 AUTH3: call_id: 2, Fragment: Single, NTLMSSP_AUTH, User: DESKTOP-FS8NLP3\user
SMB2     138 Write Response
WINREG   290 OpenHKU request
DCERPC   202 Fault: call_id: 2, Fragment: Single, Ctx: 0, status: nca_s_fault_access_denied
SMB2     146 Close Request File: winreg
SMB2     182 Close Response
SMB2     190 Create Request File: winreg
SMB2     210 Create Response File: winreg
DCERPC   286 Bind: call_id: 2, Fragment: Single, 2 context items: WINREG V1.0 (32bit NDR), WINREG V1.0 (6cb71c2c-9812-4540-0300-000000000000)
SMB2     138 Write Response
SMB2     171 Read Request Len:1024 Off:0 File: winreg
DCERPC   230 Bind_ack: call_id: 2, Fragment: Single, max_xmit: 4280 max_recv: 4280, 2 results: Acceptance, Negotiate ACK
WINREG   214 OpenHKU request
WINREG   218 OpenHKU response
WINREG   222 GetVersion request
WINREG   202 GetVersion response
WINREG   342 OpenKey request, S-1-5-21-3453662643-653229605-3378087727-1001
```

# PDU Authentication Level

```
DCERPC    334 Bind: call_id: 2, Fragment: Single, 2 context items: WINREG V1.0 (32bit NDR), WINREG V1.0 (6cb71c2c-9812-4540-0300-000000000000)
SMB2      138 Write Response
SMB2      171 Read Request Len:1024 Off:0 File: winreg
DCERPC    476 Bind_ack: call_id: 2, Fragment: Single, max_xmit: 4280 max_recv: 4280, 2 results: Acceptance, Negotiate ACK, NTLMSSP_CHALLENGE
DCERPC    718 AUTH3: call_id: 2, Fragment: Single, NTLMSSP_AUTH, User: DESKTOP-FS8NLP3\user
SMB2      138 Write Response
WINREG    290 OpenHKU request
DCERPC    202 Fault: call_id: 2, Fragment: Single, Ctx: 0, status: nca_s_fault_access_denied
```

```
Auth Length: 40
Call ID: 2
Max Xmit Frag: 4280
Max Recv Frag: 4280
Assoc Group: 0x00000000
Num Ctx Items: 2
> Ctx Item[1]: Context ID:0, WINREG, 32bit NDR
> Ctx Item[2]: Context ID:1, WINREG, Bind Time Feature Negotiation
v Auth Info: NTLMSSP, Packet privacy, AuthContextId(0)
    Auth type: NTLMSSP (10)
    Auth level: Packet privacy (6)
    Auth pad len: 0
    Auth Rsrvd: 0
    Auth Context ID: 0
  v NTLM Secure Service Provider
      NTLMSSP identifier: NTLMSSP
      NTLM Message Type: NTLMSSP_NEGOTIATE (0x00000001)
```

# PDU Authentication Level

```
DCERPC    334 Bind: call_id: 2, Fragment: Single, 2 context items: WINREG V1.0 (32bit NDR), WINREG V1.0 (6cb71c2c-9812-4540-0300-000000000000)
SMB2      138 Write Response
SMB2      171 Read Request Len:1024 Off:0 File: winreg
DCERPC    476 Bind_ack: call_id: 2, Fragment: Single, max_xmit: 4280 max_recv: 4280, 2 results: Acceptance, Negotiate ACK, NTLMSSP_CHALLENGE
DCERPC    718 AUTH3: call_id: 2, Fragment: Single, NTLMSSP_AUTH, User: DESKTOP-FS8NLP3\user
SMB2      138 Write Response
WINREG    290 OpenHKU request
DCERPC    202 Fault: call_id: 2, Fragment: Single, Ctx: 0, status: nca_s_fault_access_denied
```

```
▾ Auth Info: NTLMSSP, Packet privacy, AuthContextId(0)
      Auth type: NTLMSSP (10)
      Auth level: Packet privacy (6)
      Auth pad len: 0
      Auth Rsrvd: 0
      Auth Context ID: 0
    ▾ NTLMSSP Verifier
          Version Number: 1
          Verifier Body: e8b48bbec787a3db00000000
    [Response in frame: 12]
▾ Remote Registry Service, OpenHKU
    Operation: OpenHKU (4)
    [Response in frame: 12]
    Encrypted stub data: 6dd7bc786dfe39265a3adb071f45d41c3419b13bf69498ca...
```

# PDU Authentication Level

```
DCERPC    334 Bind: call_id: 2, Fragment: Single, 2 context items: WINREG V1.0 (32bit NDR), WINREG V1.0 (6cb71c2c-9812-4540-0300-000000000000)
SMB2      138 Write Response
SMB2      171 Read Request Len:1024 Off:0 File: winreg
DCERPC    476 Bind_ack: call_id: 2, Fragment: Single, max_xmit: 4280 max_recv: 4280, 2 results: Acceptance, Negotiate ACK, NTLMSSP_CHALLENGE
DCERPC    718 AUTH3: call_id: 2, Fragment: Single, NTLMSSP_AUTH, User: DESKTOP-FS8NLP3\user
SMB2      138 Write Response
WINREG    290 OpenHKU request
DCERPC    202 Fault: call_id: 2, Fragment: Single, Ctx: 0, status: nca_s_fault_access_denied
```

In case of catastrophic errors (such as an out of memory condition or buffer overrun), a server MAY send a fault PDU or just close the connection. For information on client and server state machines, see sections 3.3.2 and 3.3.3.

- If the security provider indicates an error, the RPC runtime MUST take recovery action depending on whether this is the client or server.
  - If this is the client, the RPC runtime discards the security context and MUST NOT send any further PDUs on that connection. It SHOULD close the connection unless it is expecting responses on a multiplexed connection, as specified in section 3.3.1.5.8, in which case it SHOULD set the Activity's **Discard** flag to TRUE. If it does not wait for all responses on a multiplexe connection, it MUST provide indication in an implementation-specific way to upper layers that the outstanding calls have failed.

https://msdn.microsoft.com/en-us/library/cc243713.aspx

# PDU Authentication Level

```
DCERPC    286 Bind: call_id: 2, Fragment: Single, 2 context items: WINREG V1.0 (32bit NDR), WINREG V1.0 (6cb71c2c-9812-4540-0300-000000000000)
SMB2      138 Write Response
SMB2      171 Read Request Len:1024 Off:0 File: winreg
DCERPC    230 Bind_ack: call_id: 2, Fragment: Single, max_xmit: 4280 max_recv: 4280, 2 results: Acceptance, Negotiate ACK
WINREG    214 OpenHKU request
WINREG    218 OpenHKU response
WINREG    222 GetVersion request
WINREG    202 GetVersion response
WINREG    342 OpenKey request, S-1-5-21-3453662643-653229605-3378087727-1001
```

```
▾ Distributed Computing Environment / Remote Procedure Call (DCE/RPC) Bind, Fragmer
    Version: 5
    Version (minor): 0
    Packet type: Bind (11)
  ▸ Packet Flags: 0x03
  ▸ Data Representation: 10000000 (Order: Little-endian, Char: ASCII, Float: IEEE)
    Frag Length: 116
    Auth Length: 0
    Call ID: 2
    Max Xmit Frag: 4280
    Max Recv Frag: 4280
    Assoc Group: 0x00000000
    Num Ctx Items: 2
  ▸ Ctx Item[1]: Context ID:0, WINREG, 32bit NDR
  ▸ Ctx Item[2]: Context ID:1, WINREG, Bind Time Feature Negotiation
```

# PDU Authentication Level

```
DCERPC    286 Bind: call_id: 2, Fragment: Single, 2 context items: WINREG V1.0 (32bit NDR), WINREG V1.0 (6cb71c2c-9812-4540-0300-000000000000)
SMB2      138 Write Response
SMB2      171 Read Request Len:1024 Off:0 File: winreg
DCERPC    230 Bind_ack: call_id: 2, Fragment: Single, max_xmit: 4280 max_recv: 4280, 2 results: Acceptance, Negotiate ACK
WINREG    214 OpenHKU request
WINREG    218 OpenHKU response
WINREG    222 GetVersion request
WINREG    202 GetVersion response
WINREG    342 OpenKey request, S-1-5-21-3453662643-653229605-3378087727-1001
```

```
    Auth Length: 0
    Call ID: 2
    Alloc hint: 12
    Context ID: 0
    Opnum: 4
    [Response in frame: 22]
  ▾ Complete stub data (12 bytes)
      Payload stub data (12 bytes)
▾ Remote Registry Service, OpenHKU
    Operation: OpenHKU (4)
    [Response in frame: 22]
  ▾ Pointer to System Name (uint16)
      Referent ID: 0x00020000
      System Name: 1024
  ▸ Access Mask: 0x02000000
```

**The dce_c_authn_level_none Protection Level**

The **auth_value** is null; the entire authentication verifier may be omitted.

http://pubs.opengroup.org/onlinepubs/9629399/chap13.htm

# Recapping...

1. When the client initiates a connection with the server using the Remote Registry Protocol, a security context with an authentication level of 6 (Dce_c_auth_level_pkt_privacy) is associated.

2. During the first interaction with the remote machine's registry, an error occurs that forces the security context to be discarded.

3. The client initiates a new connection, this time requesting an authentication level of 1 (dce_c_auth_level_none) from the security provider, which results in no integrity or confidentiality protection for the PDUs.

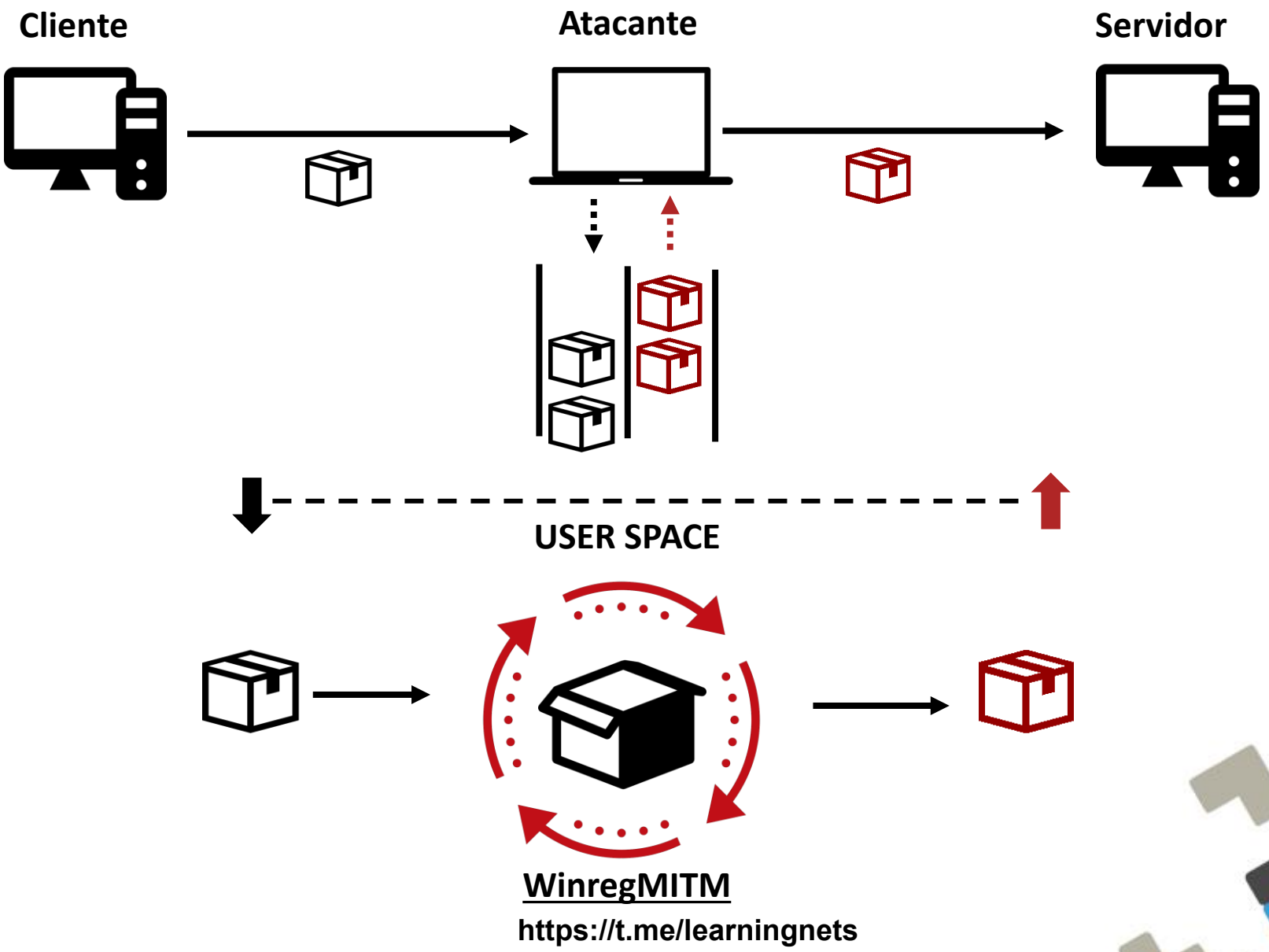
# WinregMITM

**https://github.com/shramos/winregmitm**

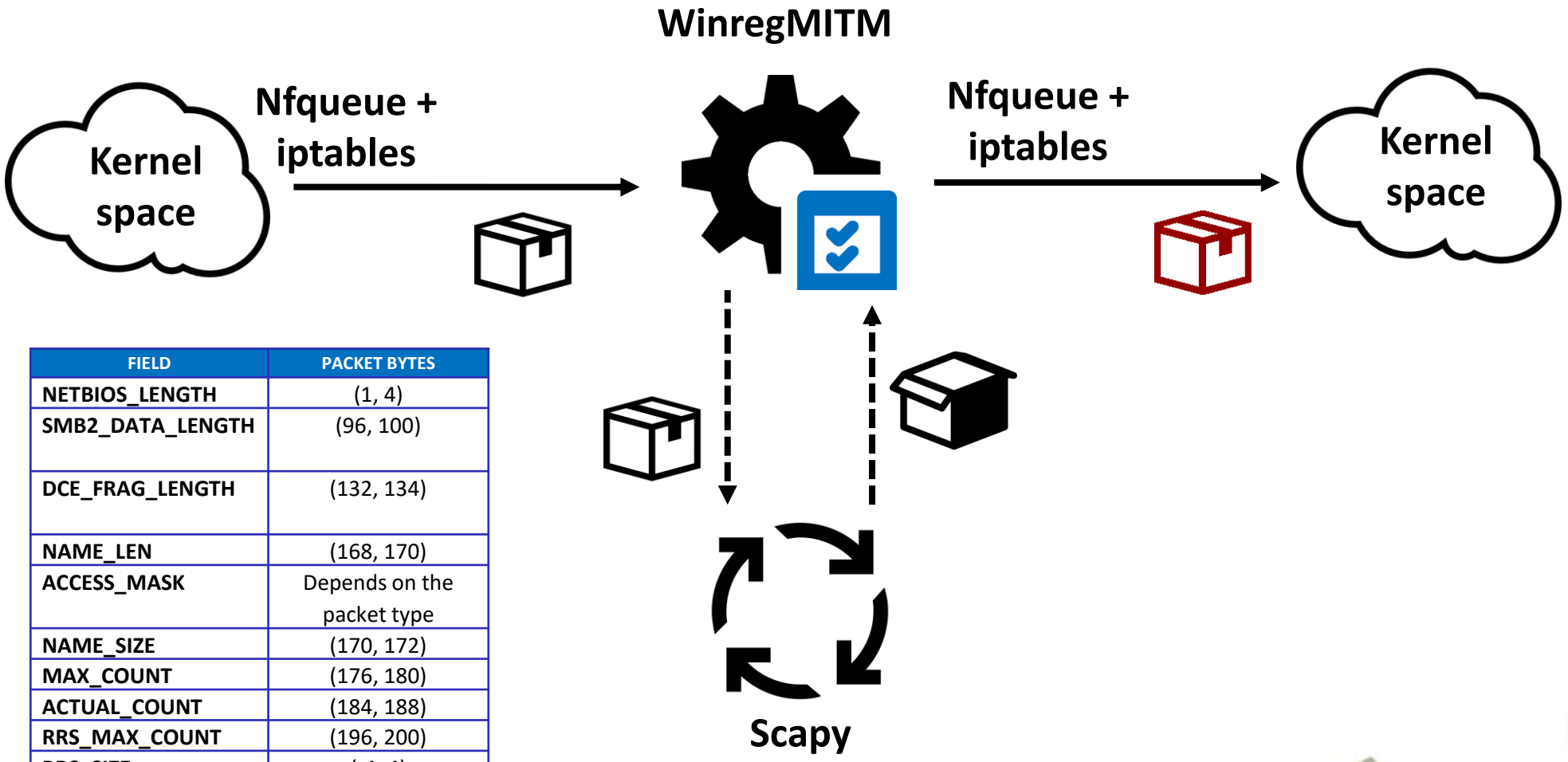
# Attack

1. Intercept the communication between the client and the server
2. Intercept the packets and forward them to user space so they can be filtered and modified by our tool.
3. Filter specific packets and interpret their bytes.
4. Modify the fields of interest.
5. Recalculate all control fields of the WINREG layer and the preceding layers.
6. Reconstruct the packet and resend it.

# WinregMITM workflow



# WinregMITM workflow



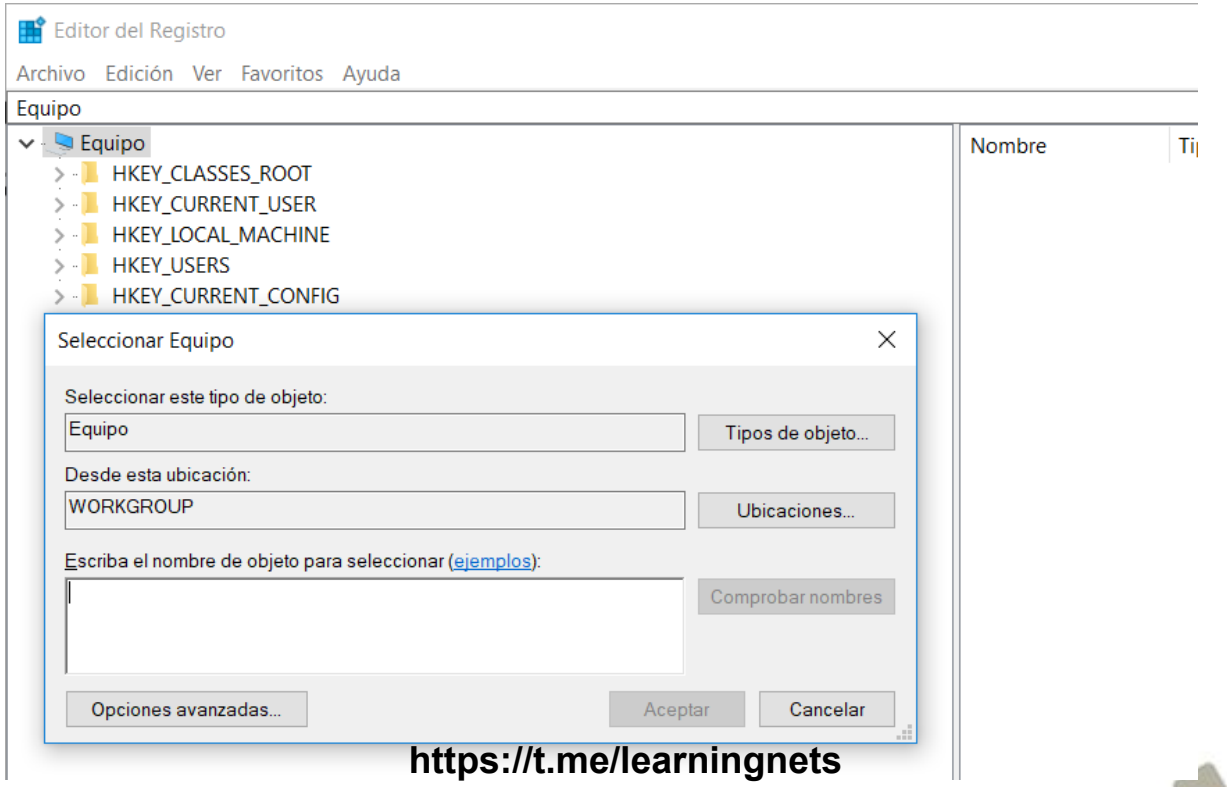

| FIELD | PACKET BYTES |
|---|---|
| NETBIOS_LENGTH | (1, 4) |
| SMB2_DATA_LENGTH | (96, 100) |
| DCE_FRAG_LENGTH | (132, 134) |
| NAME_LEN | (168, 170) |
| ACCESS_MASK | Depends on the packet type |
| NAME_SIZE | (170, 172) |
| MAX_COUNT | (176, 180) |
| ACTUAL_COUNT | (184, 188) |
| RRS_MAX_COUNT | (196, 200) |
| RRS_SIZE | (-4, 4) |

https://github.com/shramos/winregmitm
https://github.com/secdev/scapy/tree/master/scapy
http://www.netfilter.org/projects/libnetfilter_queue/index.html
https://pypi.python.org/pypi/NetfilterQueue

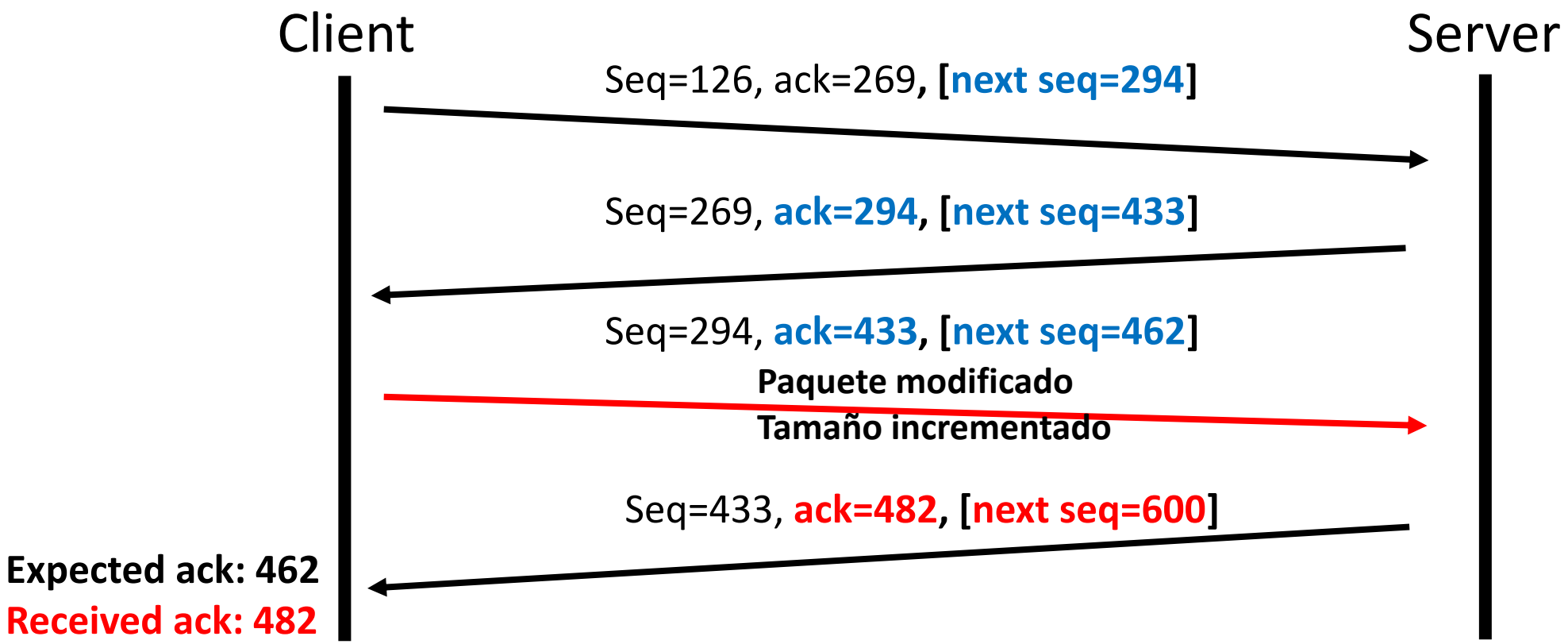
# Am I missing something?

# Regedit

- Most popular tool to access the log of a remote machine
- I modify a packet, recalculate all fields, resend the packet, and the session breaks. Why?



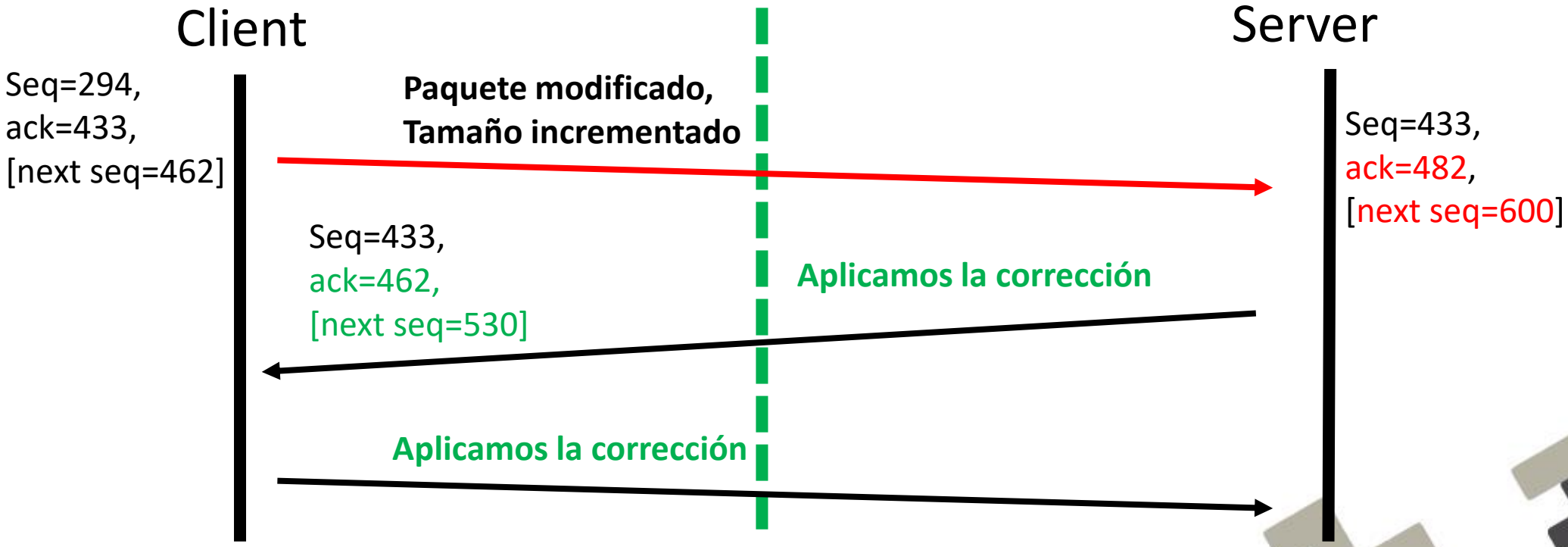
# Keeping the TCP/IP session alive



# Keeping the TCP/IP session alive

**Next sequence number = Actual sequence number + TCP payload length (*)**

(*) TCP payload length = length(IP) – length(IP.header) – length(TCP.header)



# DEMO!!!

1. ARP spoofing between the Client and the Server
2. We run WinreMITM and start monitoring user activity on the remote machine.
3. We intercept and modify an OpenKey packet and a SetValue packet to insert a payload into a specific registry branch of the remote machine, in this case: HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run
4. When the user restarts the machine, our payload will execute, performing a Fileless UAC bypass and returning a remote connection with maximum privileges to the attacking machine.

# This is not all the true…

- **The Windows Remote Registry Protocol is not always unencrypted and unsigned.**
- There are two use cases related to encryption and integrity control of the protocol:

1. Authentication is performed with a user whose name or password differs from that of the remote machine:
    - User and password must be provided.
    - The authentication level is 1 (no encryption or signing).
2. Authentication is performed with a user who has the same name and password as the remote machine:
    - User and password are not required.
    - The authentication level is 6 (encrypted and signed).

# Authentication Level 1 vs 6

- Failure during the first access to the remote registry: In the next connection, a security context is established with authentication level 1.

```
SMB2      190 Create Request File: winreg
SMB2      210 Create Response File: winreg
DCERPC    290 Bind: call_id: 4, Fragment: Single, 1 context items: WINREG V1.0 (32bit NDR), NTLMSSP_NEGOTIATE
SMB2      138 Write Response
SMB2      171 Read Request Len:1024 Off:0 File: winreg
DCERPC    372 Bind_ack: call_id: 4, Fragment: Single, max_xmit: 4280 max_recv: 4280, 1 results: Acceptance, NTLMSSP_CHALLENGE
DCERPC    628 AUTH3: call_id: 4, Fragment: Single, NTLMSSP_AUTH, User: santi-PC\santi
SMB2      138 Write Response
WINREG    242 OpenHKU request
DCERPC    202 Fault: call_id: 4, Fragment: Single, Ctx: 0, status: nca_s_fault_access_denied
SMB2      146 Close Request File: winreg
```

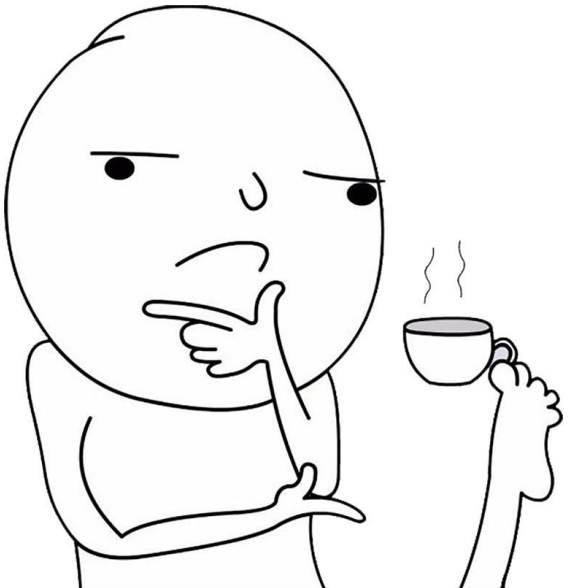
- No failure occurs: The security context with authentication level 6 is maintained.

```
SMB2      190 Create Request File: winreg
SMB2      210 Create Response File: winreg
SMB2      162 GetInfo Request FILE_INFO/SMB2_FILE_STANDARD_INFO File: winreg
SMB2      154 GetInfo Response
DCERPC    334 Bind: call_id: 2, Fragment: Single, 2 context items: WINREG V1.0 (32bit NDR), WINREG V1.0 (6cb71c2c-9812-4540-030
SMB2      138 Write Response
SMB2      171 Read Request Len:1024 Off:0 File: winreg
DCERPC    396 Bind_ack: call_id: 2, Fragment: Single, max_xmit: 4280 max_recv: 4280, 2 results: Acceptance, Negotiate ACK, NTLM
DCERPC    622 AUTH3: call_id: 2, Fragment: Single, NTLMSSP_AUTH, User: user-PC\user
SMB2      138 Write Response
WINREG    290 OpenHKLM request
WINREG    250 OpenHKLM response
```

# Forcing the security context

In case of catastrophic errors (such as an out of memory condition or buffer overrun), a server MAY send a fault PDU or just close the connection. For information on client and server state machines, see sections 3.3.2 and 3.3.3.

https://msdn.microsoft.com/en-us/library/cc243713.aspx

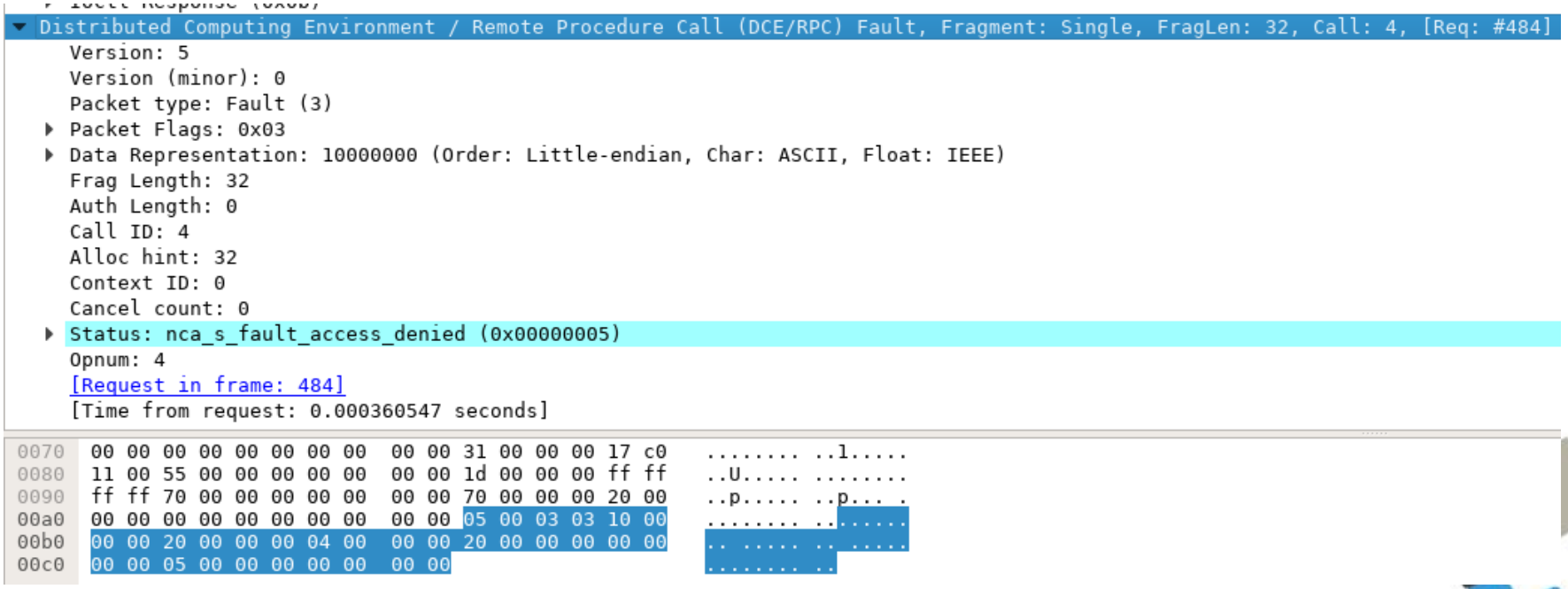




# What happens if…

1. We extract the bytes of the **DCE/RPC layer** from the Fault packet sent by the server to the client in sessions without encryption or signing.

2. We replace the **DCE/RPC layer "on-the-fly"** in the packet where the server accepts authentication level 6 in an encrypted and signed session.



# DEMO 2!!!

- Forcing the security context of a session that should sign and encrypt packets so that they are sent without any security mechanisms.

# Thank you very much!

## Questions?

shramos@protonmail.com

@santiagohramos

https://github.com/shramos/winregmitm