Call & Register

Relay attack on WinReg RPC client

Agenda

- MS-RPC introduction and overview
- RPC Authentication and NTLM (relay)
- Windows Remote Registry
- Downgrade attack -> relay

whoami

Stiv Kupchik

Security Research Team Lead @ Akamai

@kupsul  

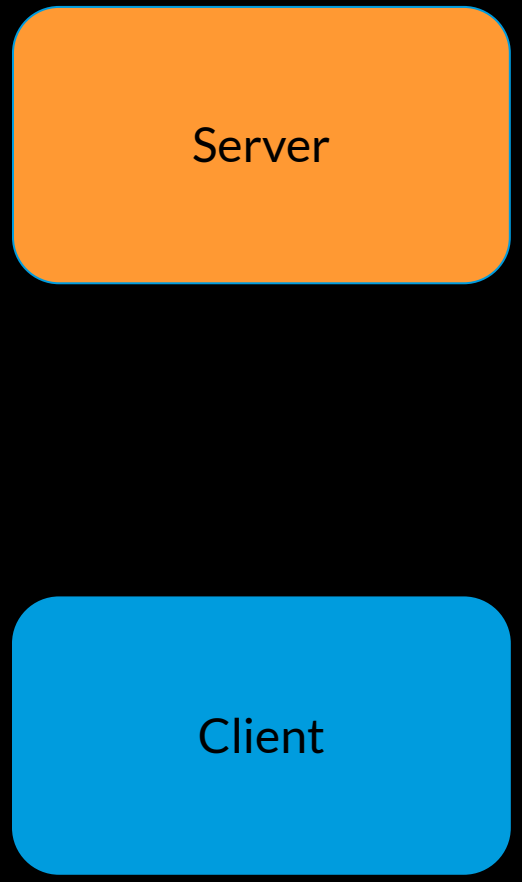
Background in DFIR and Windows internals

MS-RPC Overview

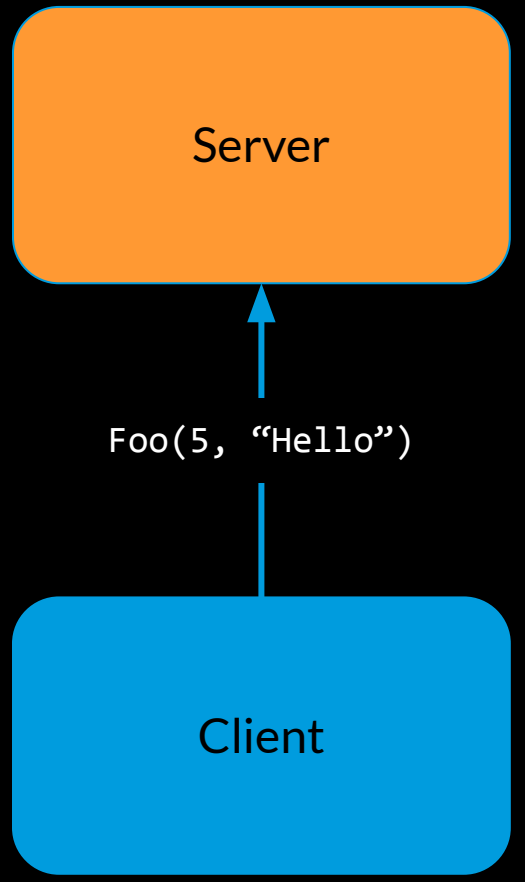
Terminology

- Interface
- {M}IDL
- Transport
- Endpoint
- Binding

The RPC Client-Server Model

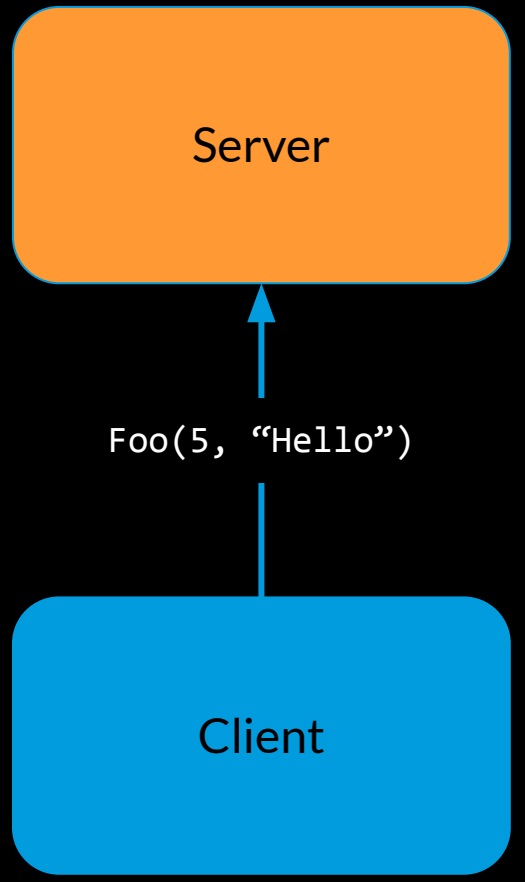


The RPC Client-Server Model

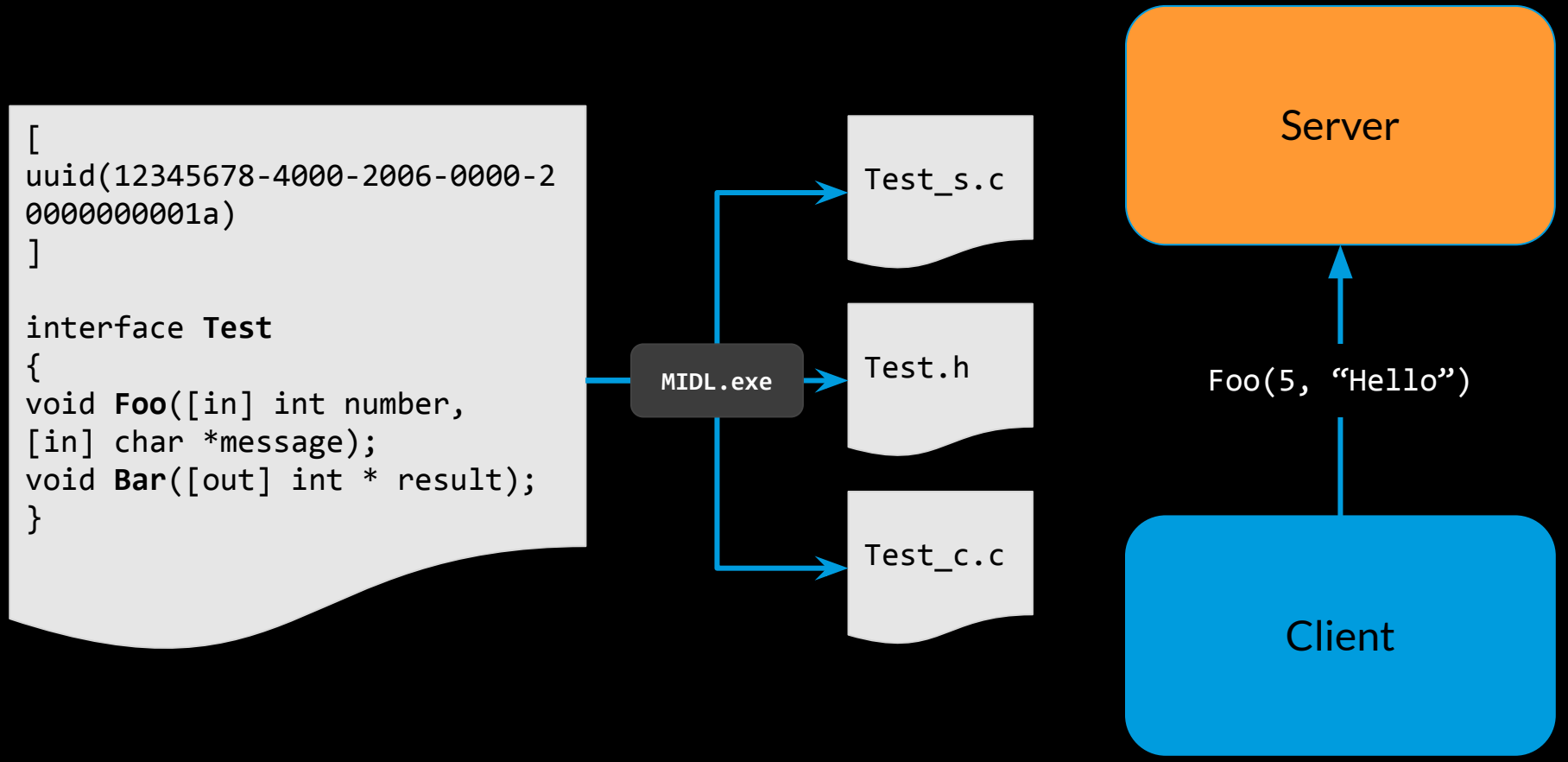


The RPC Client-Server Model

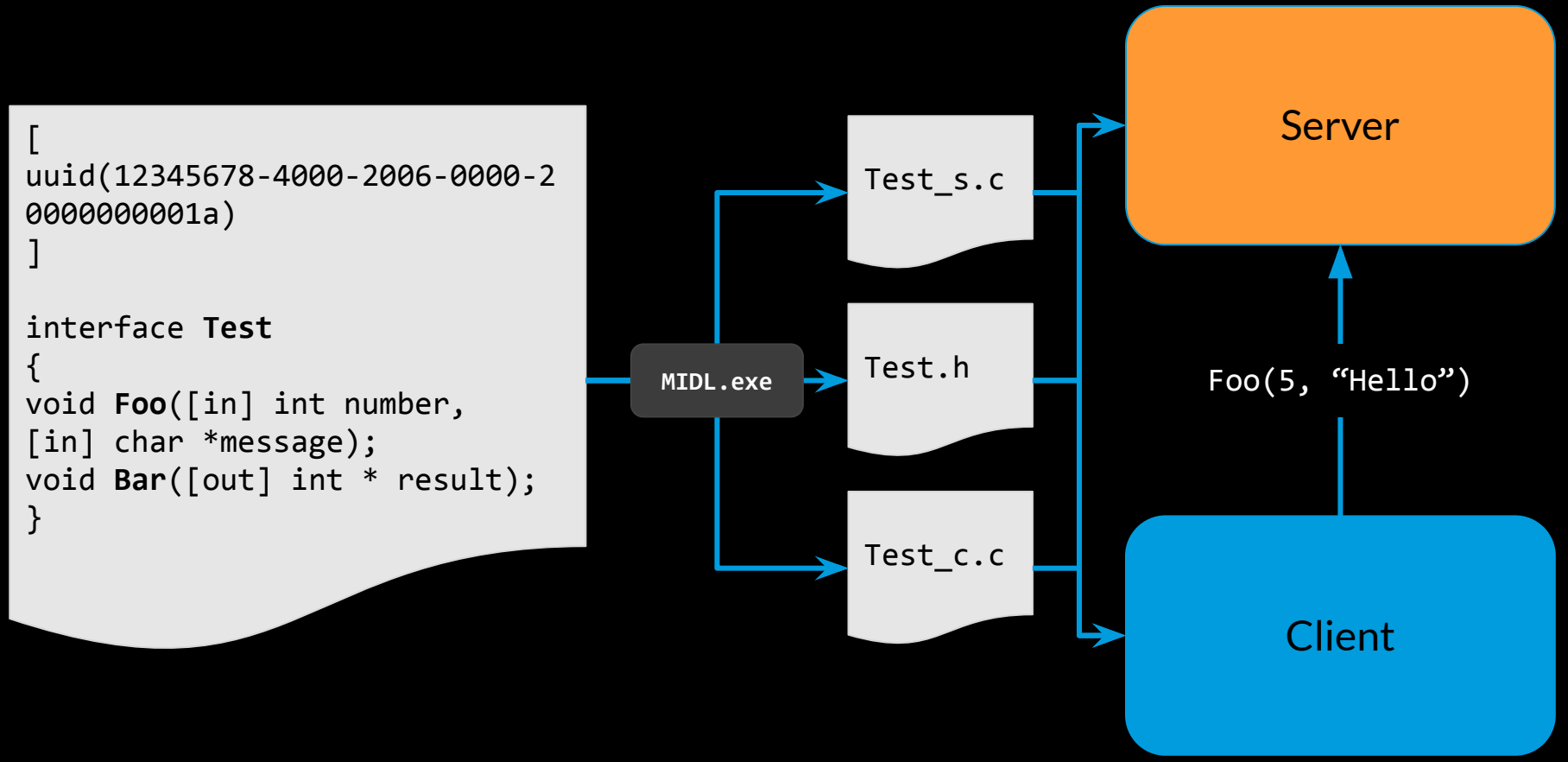
```
[
uuid(12345678-4000-2006-0000-2
0000000001a)
]

interface Test
{
void Foo([in] int number,
[in] char *message);
void Bar([out] int * result);
}
```



The RPC Client-Server Model



The RPC Client-Server Model



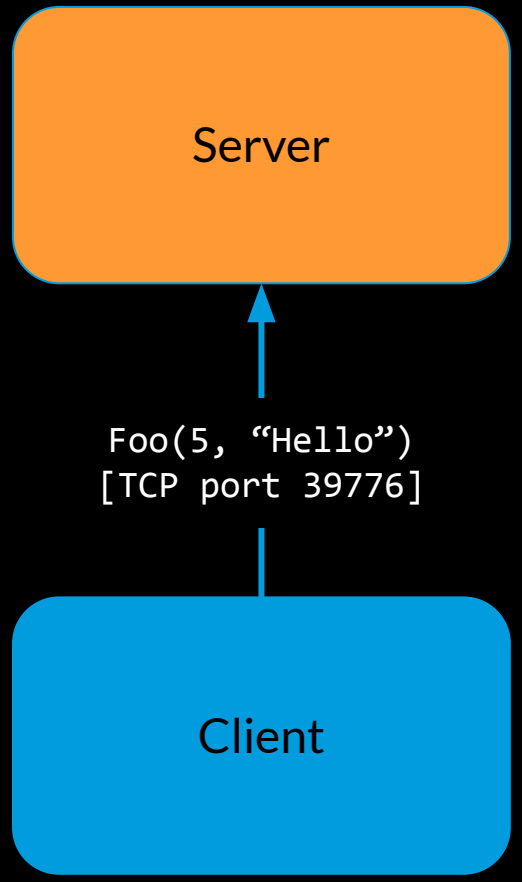
Endpoints

The server registers an *endpoint* using a certain *transport*

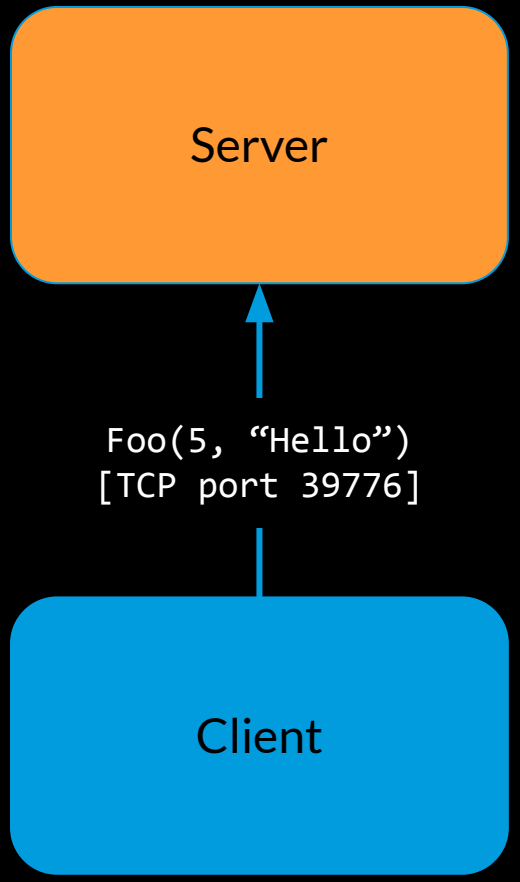
Transports	Protocol Sequence	Endpoints
TCP	`ncacn_ip_tcp`	<port number>
Named pipe	`ncacn_np`	<pipe name>
UDP	`ncadg_ip_udp`	<port number>
ALPC	`ncalrpc`	<ALPC port>
HTTP	`ncacn_http`	<hostname>
Hyper-V socket	`ncacn_hvsocket`	<UUID>

Well-Known Endpoints

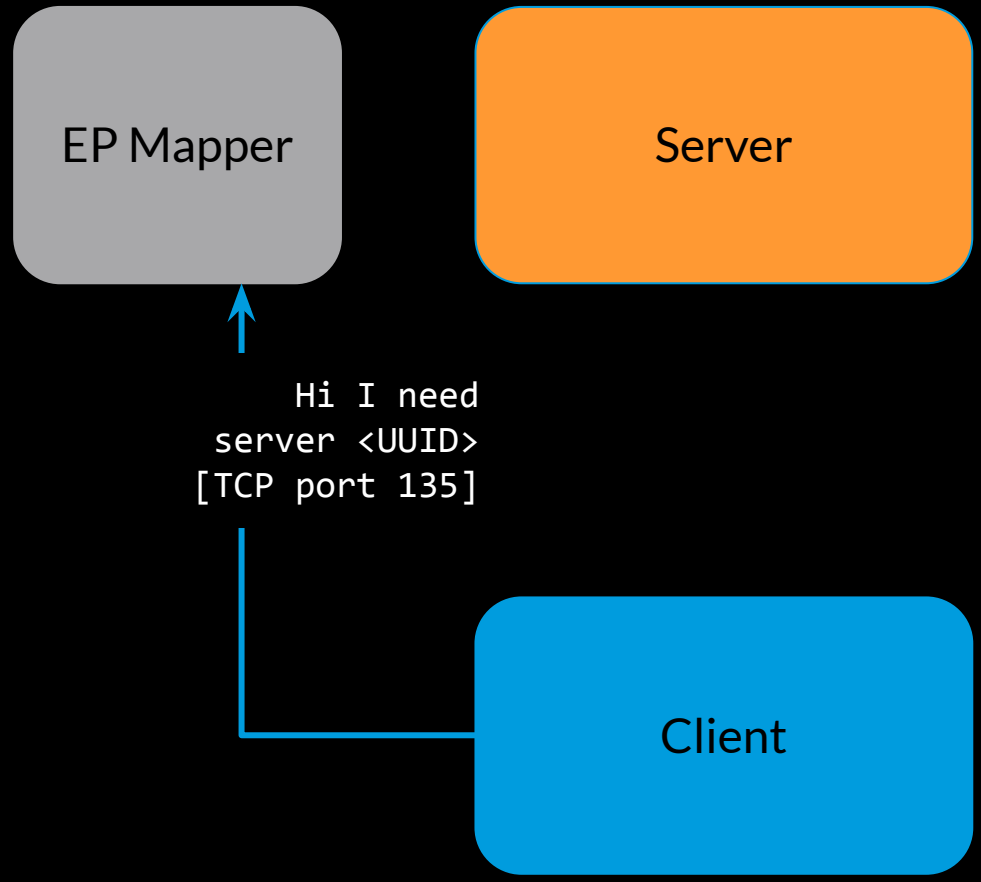
Dynamic Endpoints



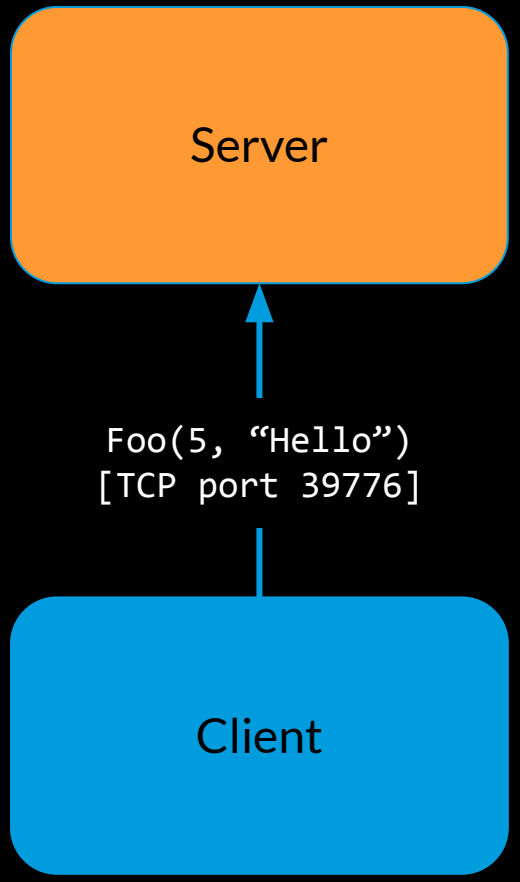
Well-Known Endpoints



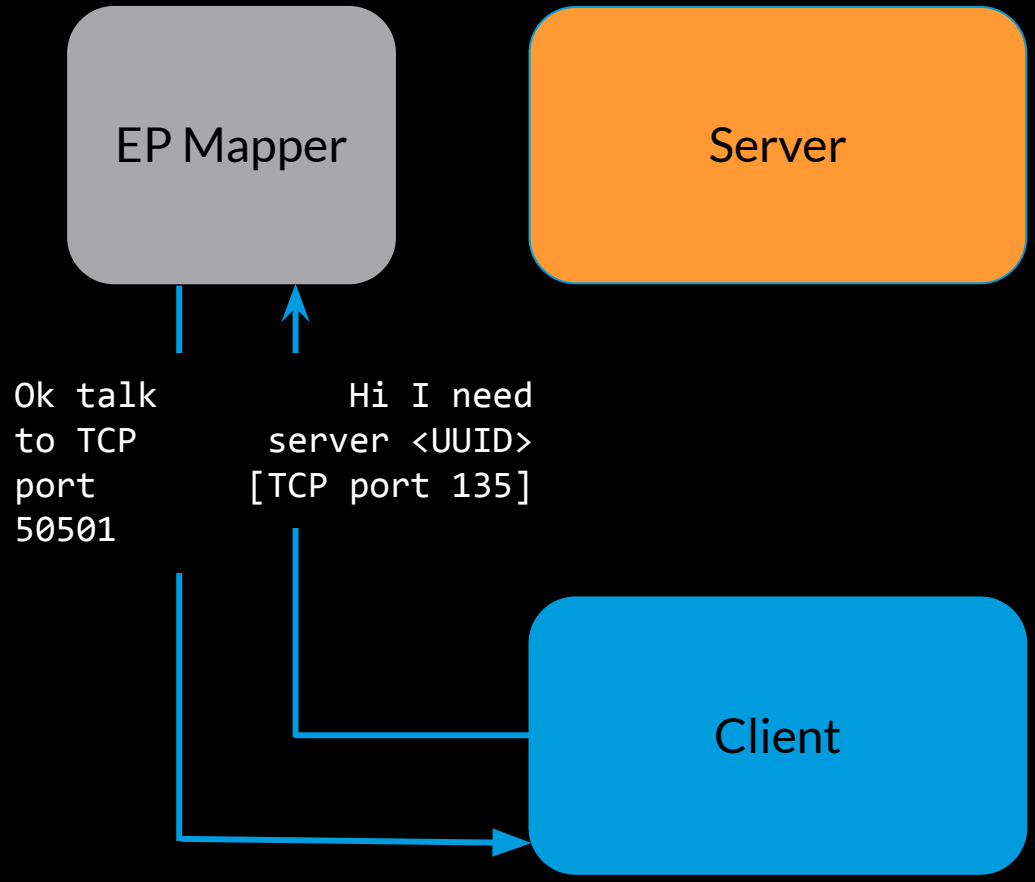
Dynamic Endpoints



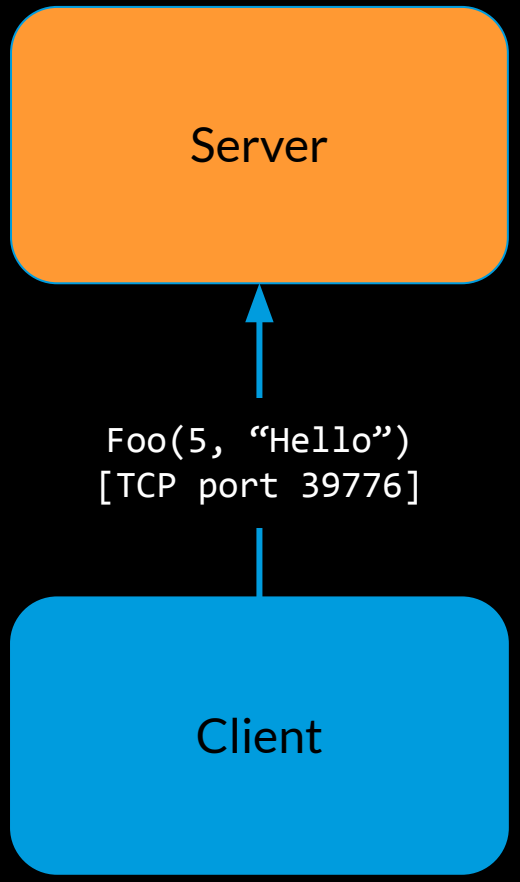
Well-Known Endpoints



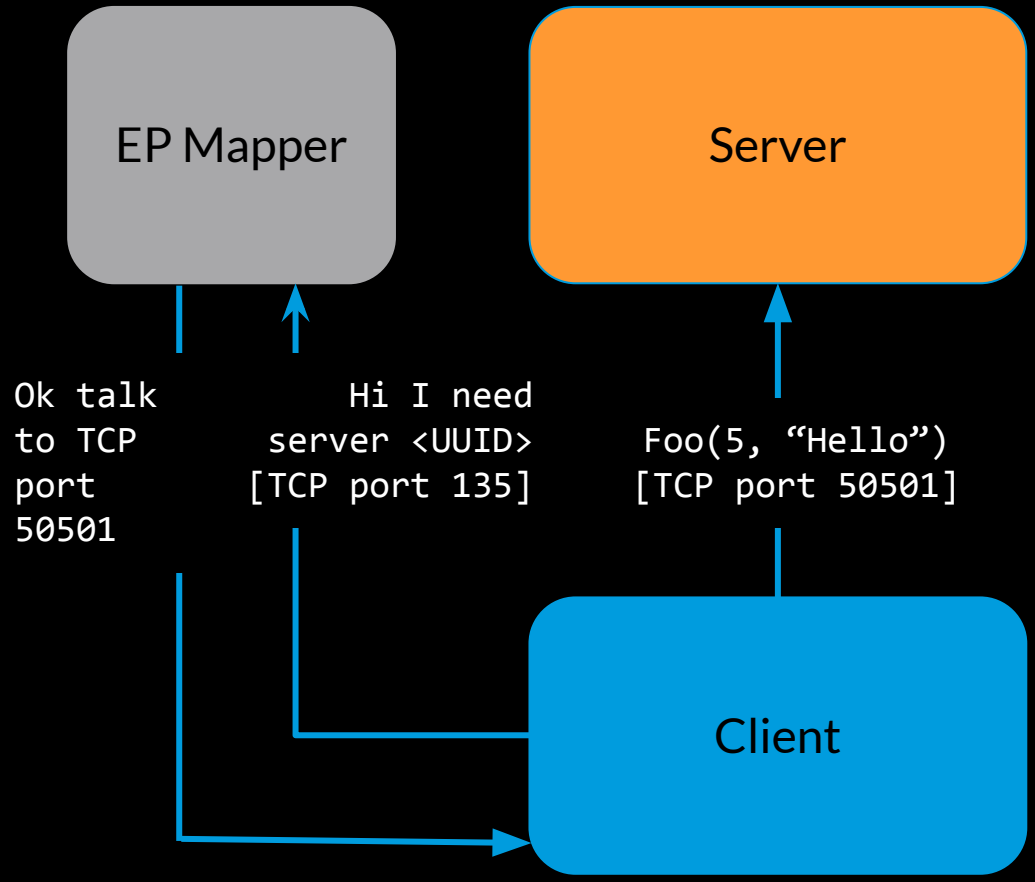
Dynamic Endpoints



Well-Known Endpoints



Dynamic Endpoints



Name	Value	Purpose
GUID_ATSvc	1FF70682-0A51-30E8-076D-740BE8CEE98B	ATSvc UUID version 1.0
GUID_SASec	378E52B0-C0A9-11CF-822D-00AA0051E40F	SASec UUID version 1.0
GUID_ITaskSchedulerService	86D35949-83C9-4044-B424-DB363231FD0C	ITaskSchedulerService UUID version 1.0

Task Scheduler Service Remoting Protocol

Parameter	Value
RPC interface UUID	{367ABB81-9844-35F1-AD32-98F038001003}
Named pipe	\PIPE\svcctl

Service control manager remote protocol

Parameter	Value
RPC Well-Known Endpoint	\pipe\lsarpc<3>
RPC Interface UUID	{c681d488-d850-11d0-8c52-00c04fd90f7e}
RPC Well-Known Endpoint	\pipe\efsrpc
RPC Interface UUID	{df1941c5-fe89-4e79-bf10-463657acf44d}

Encrypting File System Remote (EFSRPC) Protocol

Endpoint Resolution

```
192.168.0.4 192.168.0.5 TCP      66 52803 → 135 [SYN, ECE, CWR] Seq=0 Win=64240 Len=0 MSS=1
192.168.0.5 192.168.0.4 TCP      66 135 → 52803 [SYN, ACK, ECE] Seq=0 Ack=1 Win=65535 Len=0
192.168.0.4 192.168.0.5 TCP      54 52803 → 135 [ACK] Seq=1 Ack=1 Win=2102272 Len=0
192.168.0.4 192.168.0.5 DCERPC  214 Bind: call_id: 2, Fragment: Single, 3 context items: EP
192.168.0.5 192.168.0.4 DCERPC  162 Bind_ack: call_id: 2, Fragment: Single, max_xmit: 5840
192.168.0.4 192.168.0.5 EPM     222 Map request, SVCCTL, 32bit NDR
192.168.0.5 192.168.0.4 EPM     226 Map response, SVCCTL, 32bit NDR
192.168.0.4 192.168.0.5 TCP      66 52804 → 49704 [SYN, ECE, CWR] Seq=0 Win=64240 Len=0 MSS
192.168.0.5 192.168.0.4 TCP      66 49704 → 52804 [SYN, ACK, ECE] Seq=0 Ack=1 Win=65535 Len
192.168.0.4 192.168.0.5 TCP      54 52804 → 49704 [ACK] Seq=1 Ack=1 Win=2102272 Len=0
192.168.0.4 192.168.0.5 DCERPC  218 Bind: call_id: 2, Fragment: Single, 2 context items: SV
192.168.0.5 192.168.0.4 DCERPC  416 Bind_ack: call_id: 2, Fragment: Single, max_xmit: 5840
192.168.0.4 192.168.0.5 DCERPC  644 AUTH3: call_id: 2, Fragment: Single, NTLMSSP_AUTH, User
192.168.0.4 192.168.0.5 SVCCTL  214 OpenSCManagerW request
```

Endpoint Resolution

```
192.168.0.4 192.168.0.5 TCP      66 52803 → 135 [SYN, ECE, CWR] Seq=0 Win=64240 Len=0 MSS=1
192.168.0.5 192.168.0.4 TCP      66 135 → 52803 [SYN, ACK, ECE] Seq=0 Ack=1 Win=65535 Len=0
192.168.0.4 192.168.0.5 TCP      54 52803 → 135 [ACK] Seq=1 Ack=1 Win=2102272 Len=0
192.168.0.4 192.168.0.5 DCERPC  214 Bind: call_id: 2, Fragment: Single, 3 context items: EP
192.168.0.5 192.168.0.4 DCERPC  162 Bind_ack: call_id: 2, Fragment: Single, max_xmit: 5840
192.168.0.4 192.168.0.5 EPM     222 Map request, SVCCTL, 32bit NDR
192.168.0.5 192.168.0.4 EPM     226 Map response, SVCCTL, 32bit NDR
192.168.0.4 192.168.0.5 TCP      66 52804 → 49704 [SYN, ECE, CWR] Seq=0 Win=64240 Len=0 MSS
192.168.0.5 192.168.0.4 TCP      66 49704 → 52804 [SYN, ACK, ECE] Seq=0 Ack=1 Win=65535 Len
192.168.0.4 192.168.0.5 TCP      54 52804 → 49704 [ACK] Seq=1 Ack=1 Win=2102272 Len=0
192.168.0.4 192.168.0.5 DCERPC  218 Bind: call_id: 2, Fragment: Single, 2 context items: SV
192.168.0.5 192.168.0.4 DCERPC  416 Bind_ack: call_id: 2, Fragment: Single, max_xmit: 5840
192.168.0.4 192.168.0.5 DCERPC  644 AUTH3: call_id: 2, Fragment: Single, NTLMSSP_AUTH, User
192.168.0.4 192.168.0.5 SVCCTL  214 OpenSCManagerW request
```

Endpoint Resolution

```
192.168.0.4 192.168.0.5 TCP      66 52803 → 135 [SYN, ECE, CWR] Seq=0 Win=64240 Len=0 MSS=1
192.168.0.5 192.168.0.4 TCP      66 135 → 52803 [SYN, ACK, ECE] Seq=0 Ack=1 Win=65535 Len=0
192.168.0.4 192.168.0.5 TCP      54 52803 → 135 [ACK] Seq=1 Ack=1 Win=2102272 Len=0
192.168.0.4 192.168.0.5 DCERPC  214 Bind: call_id: 2, Fragment: Single, 3 context items: EP
192.168.0.5 192.168.0.4 DCERPC  162 Bind_ack: call_id: 2, Fragment: Single, max_xmit: 5840
192.168.0.4 192.168.0.5 EPM     222 Map request, SVCCTL, 32bit NDR
192.168.0.5 192.168.0.4 EPM     226 Map response, SVCCTL, 32bit NDR
192.168.0.4 192.168.0.5 TCP      66 52804 → 49704 [SYN, ECE, CWR] Seq=0 Win=64240 Len=0 MSS
                                    52804 [SYN, ACK, ECE] Seq=0 Ack=1 Win=65535 Len
                                    49704 [ACK] Seq=1 Ack=1 Win=2102272 Len=0
                                   all_id: 2, Fragment: Single, 2 context items: SV
                                   k: call_id: 2, Fragment: Single, max_xmit: 5840
                                   call_id: 2, Fragment: Single, NTLMSSP_AUTH, User
                                   anagerW request
```

```
▾ Tower pointer:
    Referent ID: 0x0000000000000003
    Length: 75
    Length: 75
    Number of floors: 5
  ▸ Floor 1 UUID: SVCCTL
  ▸ Floor 2 UUID: 32bit NDR
  ▸ Floor 3 RPC connection-oriented protocol
  ▸ Floor 4 TCP Port:49704
  ▸ Floor 5 IP:192.168.0.5
```

Endpoint Resolution

```
192.168.0.4 192.168.0.5 TCP      66 52803 → 135 [SYN, ECE, CWR] Seq=0 Win=64240 Len=0 MSS=1
192.168.0.5 192.168.0.4 TCP      66 135 → 52803 [SYN, ACK, ECE] Seq=0 Ack=1 Win=65535 Len=0
192.168.0.4 192.168.0.5 TCP      54 52803 → 135 [ACK] Seq=1 Ack=1 Win=2102272 Len=0
192.168.0.4 192.168.0.5 DCERPC  214 Bind: call_id: 2, Fragment: Single, 3 context items: EP
192.168.0.5 192.168.0.4 DCERPC  162 Bind_ack: call_id: 2, Fragment: Single, max_xmit: 5840
192.168.0.4 192.168.0.5 EPM     222 Map request, SVCCTL, 32bit NDR
192.168.0.5 192.168.0.4 EPM     226 Map response, SVCCTL, 32bit NDR
192.168.0.4 192.168.0.5 TCP      66 52804 → 49704 [SYN, ECE, CWR] Seq=0 Win=64240 Len=0 MSS
                                    52804 [SYN, ACK, ECE] Seq=0 Ack=1 Win=65535 Len
                                    49704 [ACK] Seq=1 Ack=1 Win=2102272 Len=0
                                   all_id: 2, Fragment: Single, 2 context items: SV
                                   k: call_id: 2, Fragment: Single, max_xmit: 5840
                                   call_id: 2, Fragment: Single, NTLMSSP_AUTH, User
                                   anagerW request
```

```
▾ Tower pointer:
    Referent ID: 0x0000000000000003
    Length: 75
    Length: 75
    Number of floors: 5
  ▸ Floor 1 UUID: SVCCTL
  ▸ Floor 2 UUID: 32bit NDR
  ▸ Floor 3 RPC connection-oriented protocol
  ▸ Floor 4 TCP Port:49704
  ▸ Floor 5 IP:192.168.0.5
```

Endpoint Resolution

```
192.168.0.4 192.168.0.5 TCP      66 52803 → 135 [SYN, ECE, CWR] Seq=0 Win=64240 Len=0 MSS=1
192.168.0.5 192.168.0.4 TCP      66 135 → 52803 [SYN, ACK, ECE] Seq=0 Ack=1 Win=65535 Len=0
192.168.0.4 192.168.0.5 TCP      54 52803 → 135 [ACK] Seq=1 Ack=1 Win=2102272 Len=0
192.168.0.4 192.168.0.5 DCERPC  214 Bind: call_id: 2, Fragment: Single, 3 context items: EP
192.168.0.5 192.168.0.4 DCERPC  162 Bind_ack: call_id: 2, Fragment: Single, max_xmit: 5840
192.168.0.4 192.168.0.5 EPM     222 Map request, SVCCTL, 32bit NDR
192.168.0.5 192.168.0.4 EPM     226 Map response, SVCCTL, 32bit NDR
192.168.0.4 192.168.0.5 TCP      66 52804 → 49704 [SYN, ECE, CWR] Seq=0 Win=64240 Len=0 MSS
192.168.0.5 192.168.0.4 TCP      66 49704 → 52804 [SYN, ACK, ECE] Seq=0 Ack=1 Win=65535 Len
192.168.0.4 192.168.0.5 TCP      54 52804 → 49704 [ACK] Seq=1 Ack=1 Win=2102272 Len=0
192.168.0.4 192.168.0.5 DCERPC  218 Bind: call_id: 2, Fragment: Single, 2 context items: SV
192.168.0.5 192.168.0.4 DCERPC  416 Bind_ack: call_id: 2, Fragment: Single, max_xmit: 5840
192.168.0.4 192.168.0.5 DCERPC  644 AUTH3: call_id: 2, Fragment: Single, NTLMSSP_AUTH, User
192.168.0.4 192.168.0.5 SVCCTL  214 OpenSCManagerW request
```

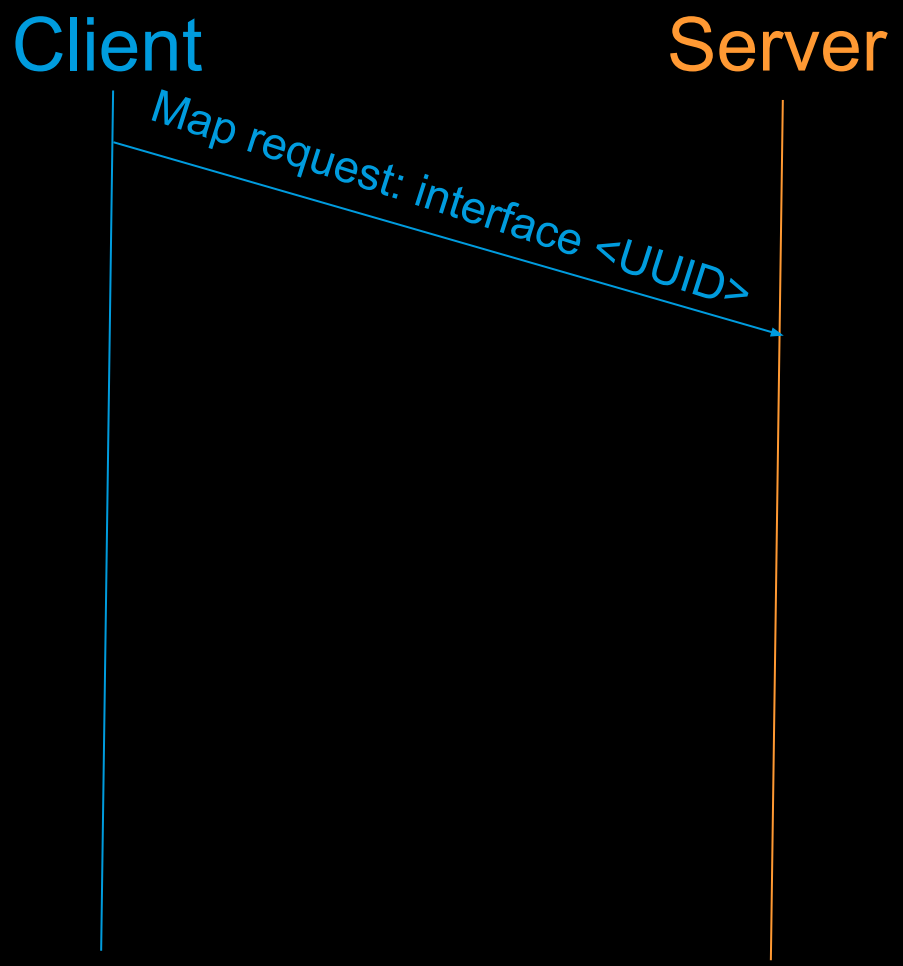
Binding

```
192.168.0.4 192.168.0.5 TCP      66 52803 → 135 [SYN, ECE, CWR] Seq=0 Win=64240 Len=0 MSS=1
192.168.0.5 192.168.0.4 TCP      66 135 → 52803 [SYN, ACK, ECE] Seq=0 Ack=1 Win=65535 Len=0
192.168.0.4 192.168.0.5 TCP      54 52803 → 135 [ACK] Seq=1 Ack=1 Win=2102272 Len=0
192.168.0.4 192.168.0.5 DCERPC  214 Bind: call_id: 2, Fragment: Single, 3 context items: EP
192.168.0.5 192.168.0.4 DCERPC  162 Bind_ack: call_id: 2, Fragment: Single, max_xmit: 5840
192.168.0.4 192.168.0.5 EPM     222 Map request, SVCCTL, 32bit NDR
192.168.0.5 192.168.0.4 EPM     226 Map response, SVCCTL, 32bit NDR
192.168.0.4 192.168.0.5 TCP      66 52804 → 49704 [SYN, ECE, CWR] Seq=0 Win=64240 Len=0 MSS
192.168.0.5 192.168.0.4 TCP      66 49704 → 52804 [SYN, ACK, ECE] Seq=0 Ack=1 Win=65535 Len
192.168.0.4 192.168.0.5 TCP      54 52804 → 49704 [ACK] Seq=1 Ack=1 Win=2102272 Len=0
192.168.0.4 192.168.0.5 DCERPC  218 Bind: call_id: 2, Fragment: Single, 2 context items: SV
192.168.0.5 192.168.0.4 DCERPC  416 Bind_ack: call_id: 2, Fragment: Single, max_xmit: 5840
192.168.0.4 192.168.0.5 DCERPC  644 AUTH3: call_id: 2, Fragment: Single, NTLMSSP_AUTH, User
192.168.0.4 192.168.0.5 SVCCTL  214 OpenSCManagerW request
```

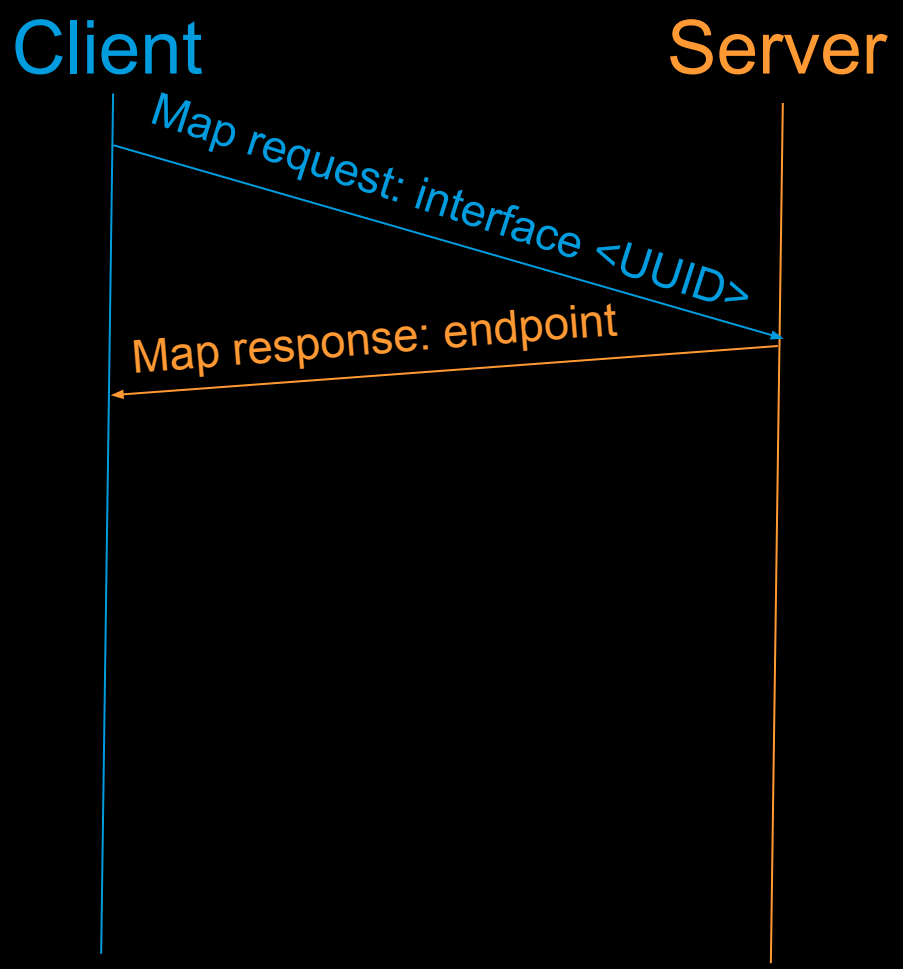
Binding

- Establishing a context session between a client and a server
- Carries details about the connection
  - Context
  - Security
  - Authentication

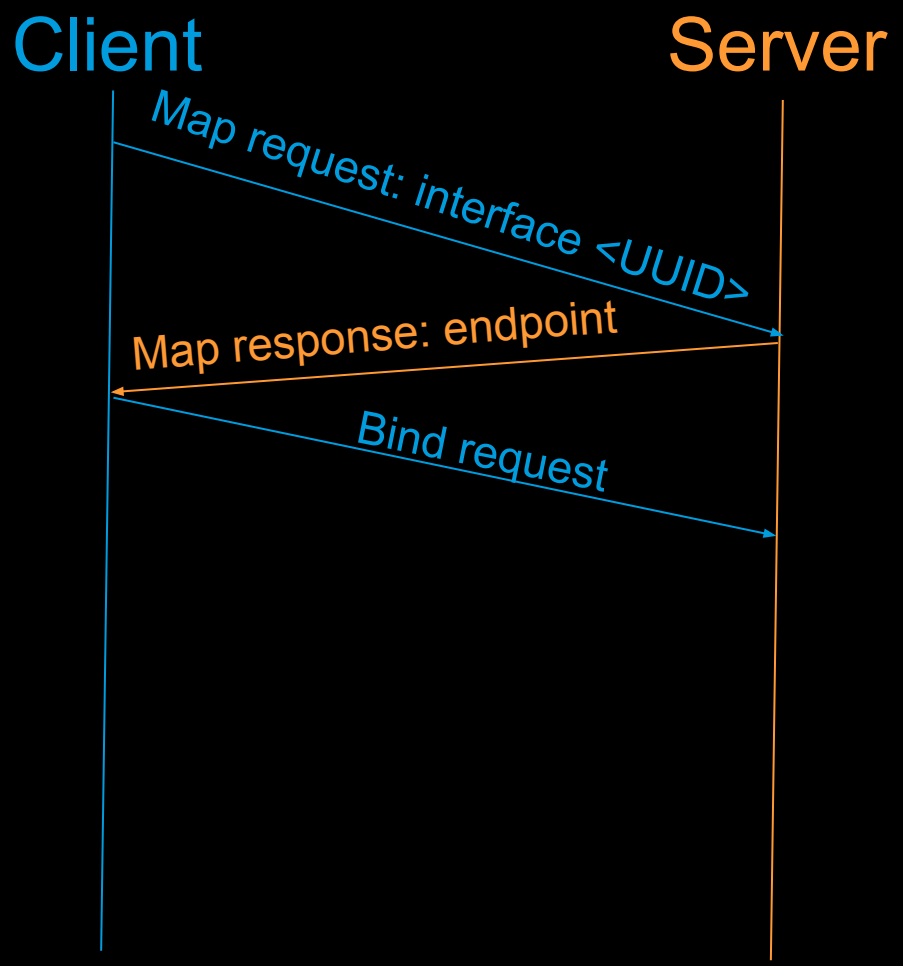
Message Flow



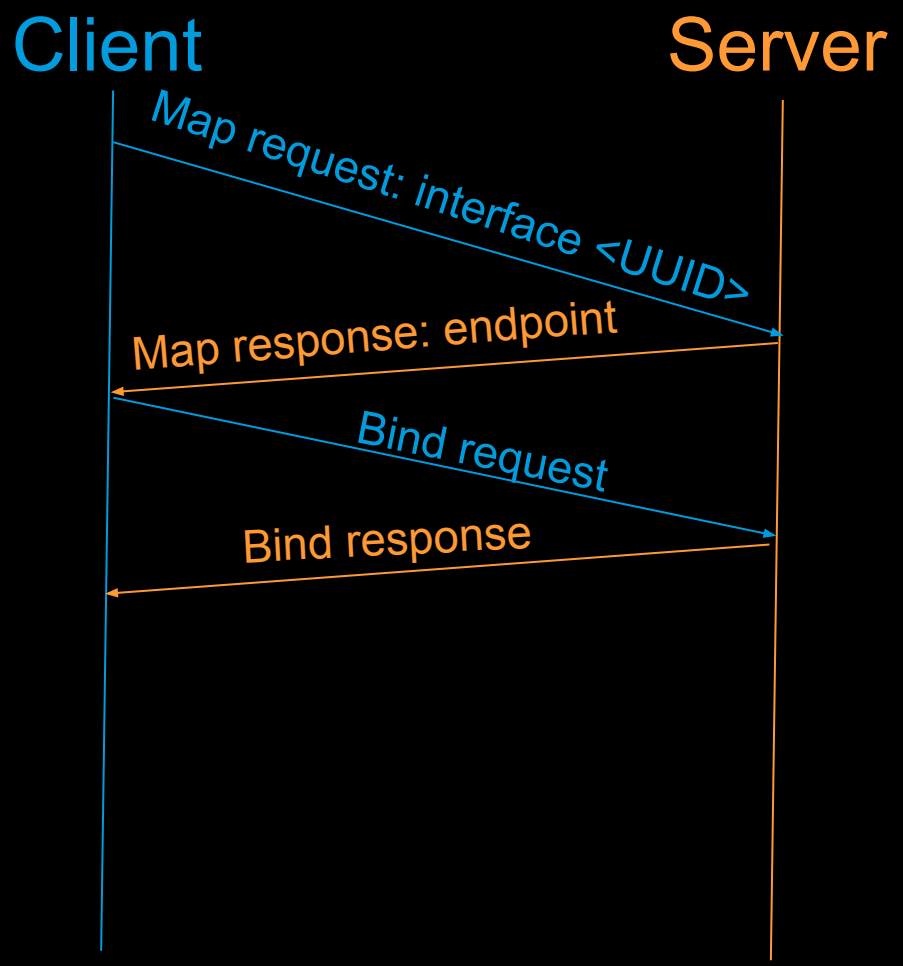
Message Flow



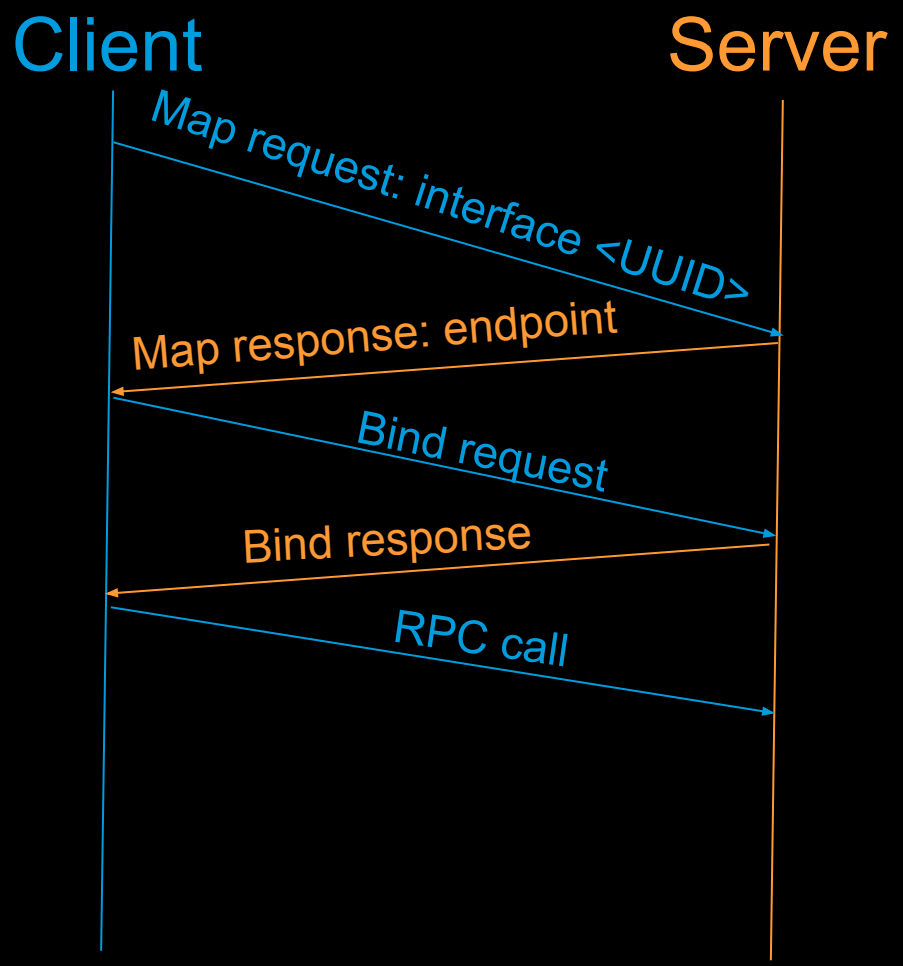
Message Flow



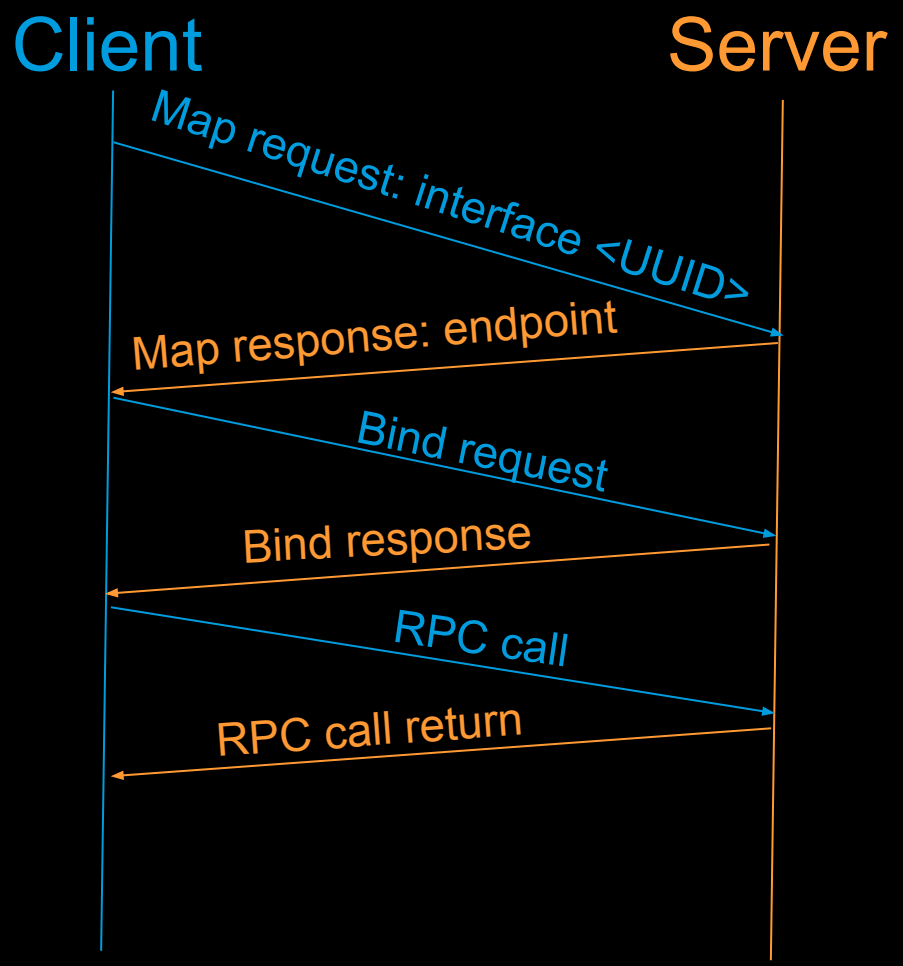
Message Flow



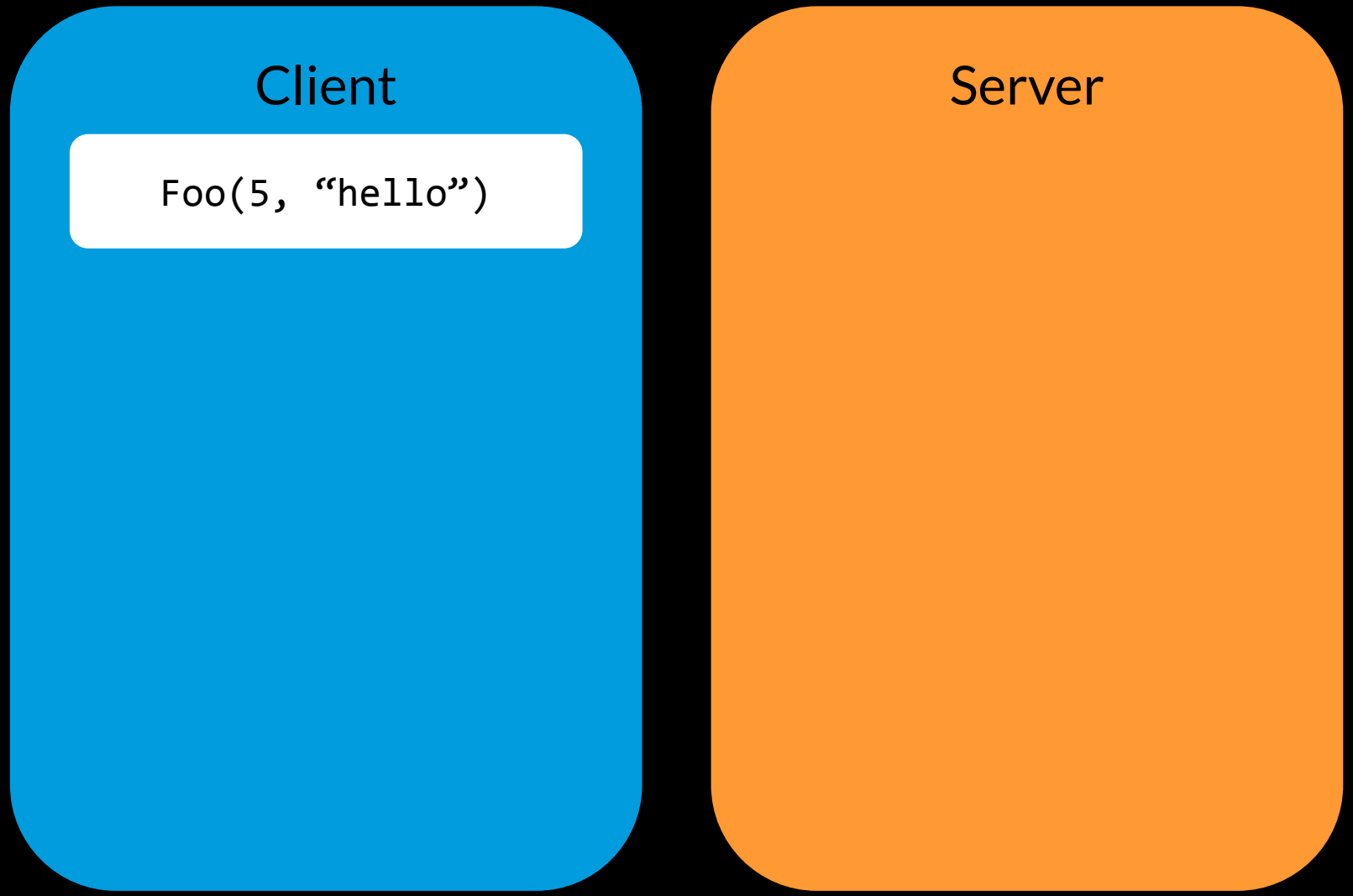
Message Flow



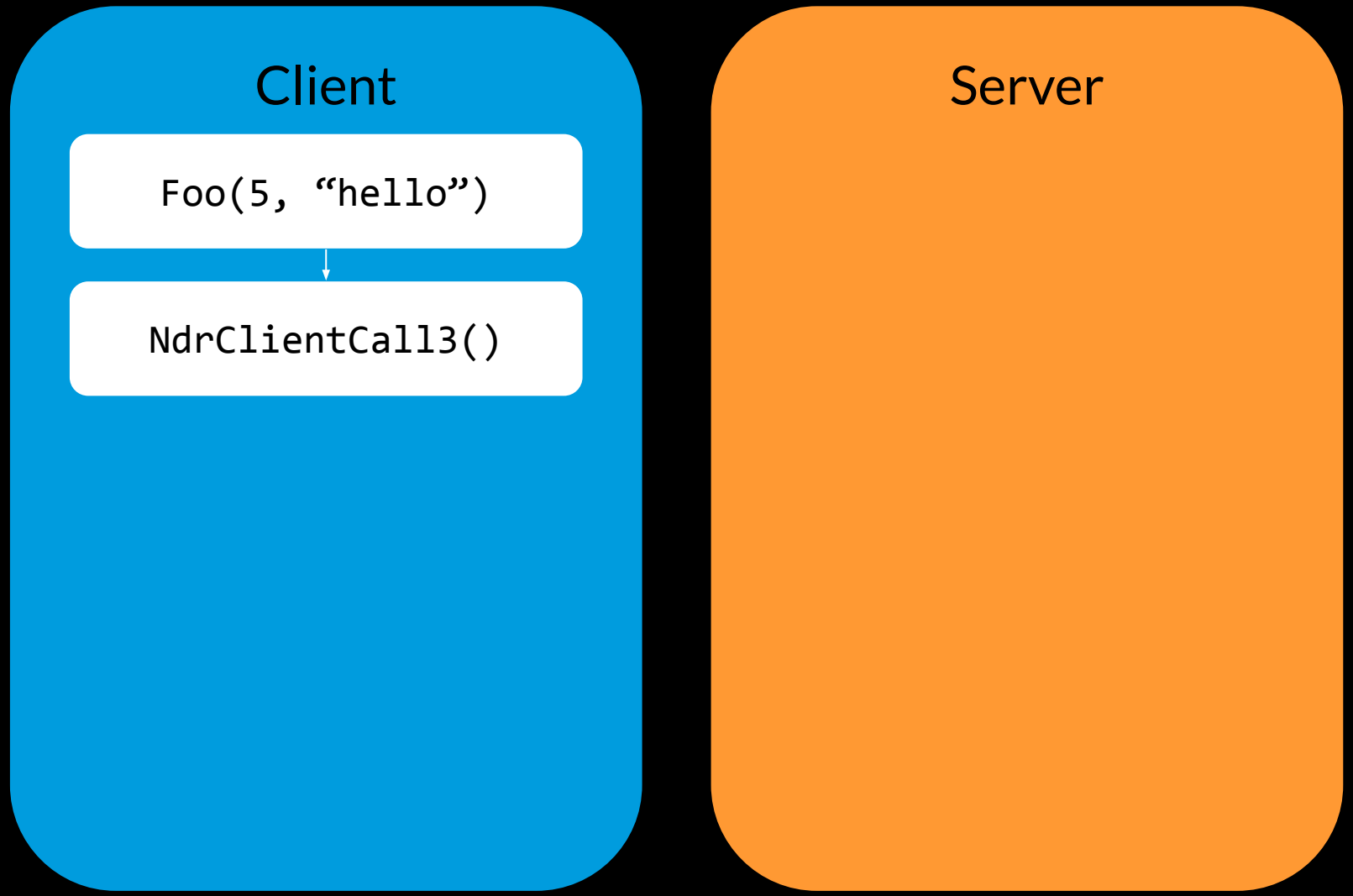
Message Flow



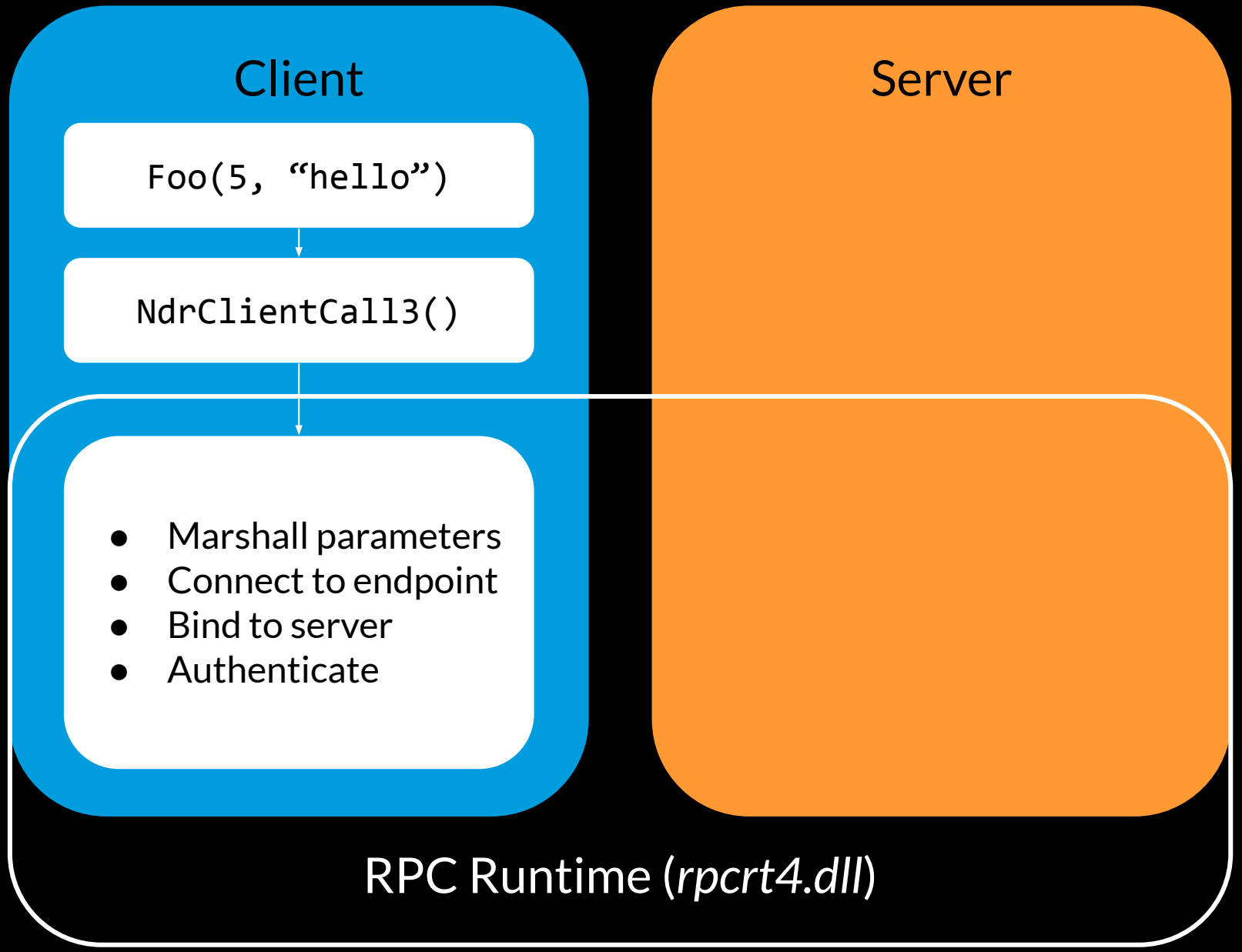
An RPC Call's Flow



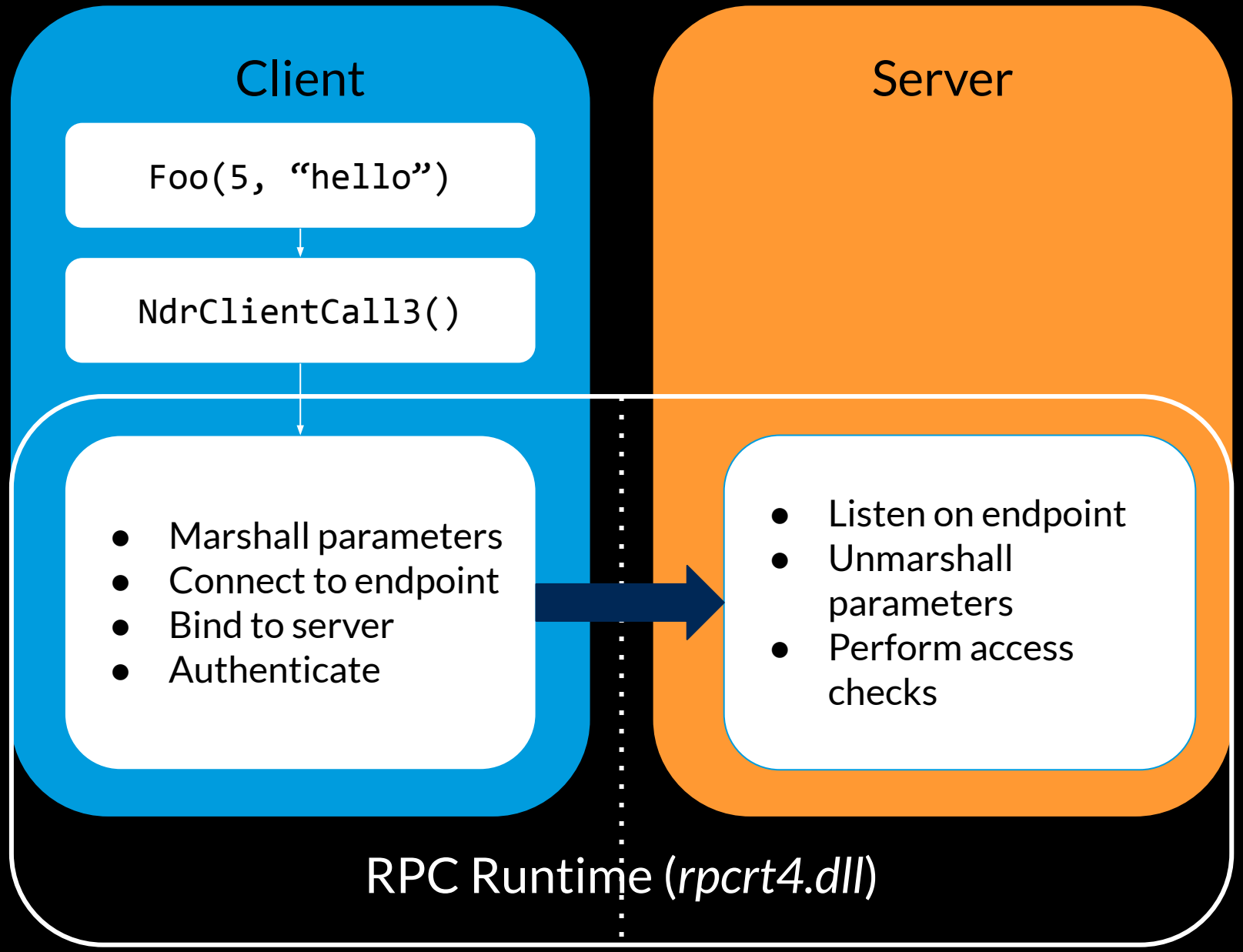
An RPC Call's Flow



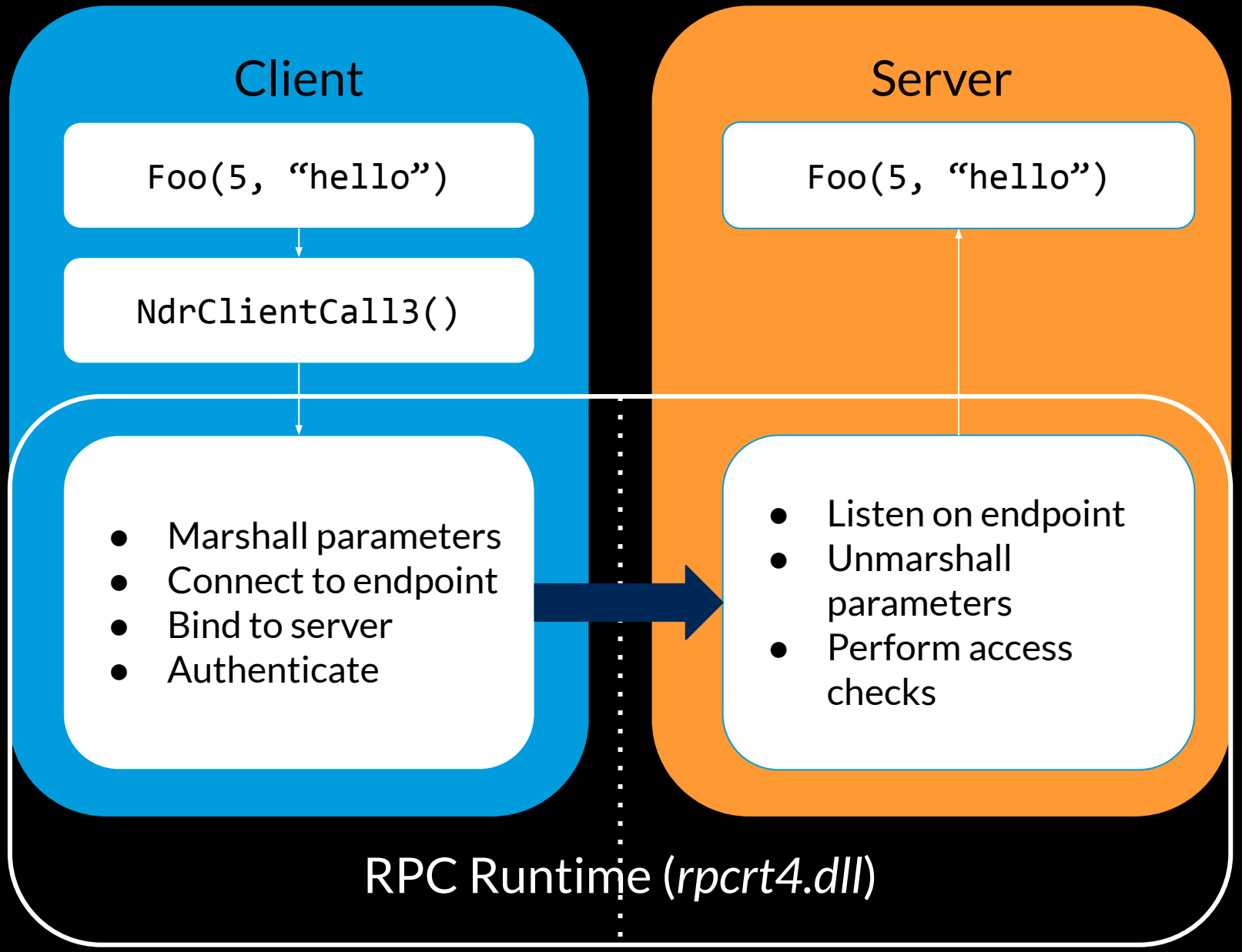
An RPC Call's Flow



An RPC Call's Flow



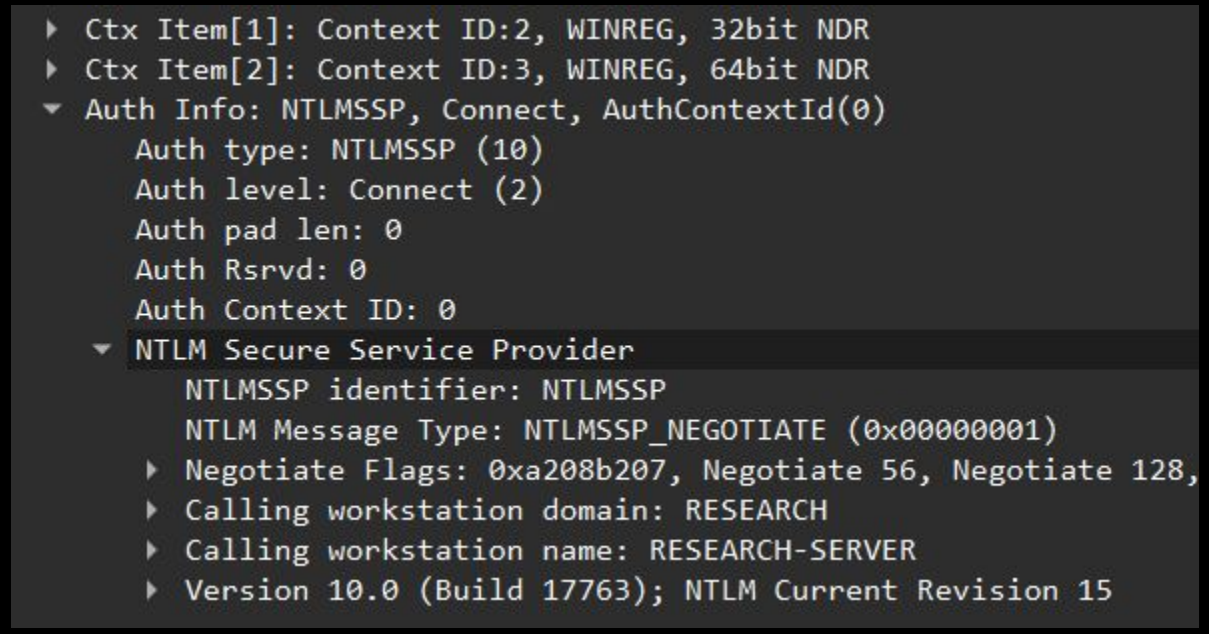
An RPC Call's Flow



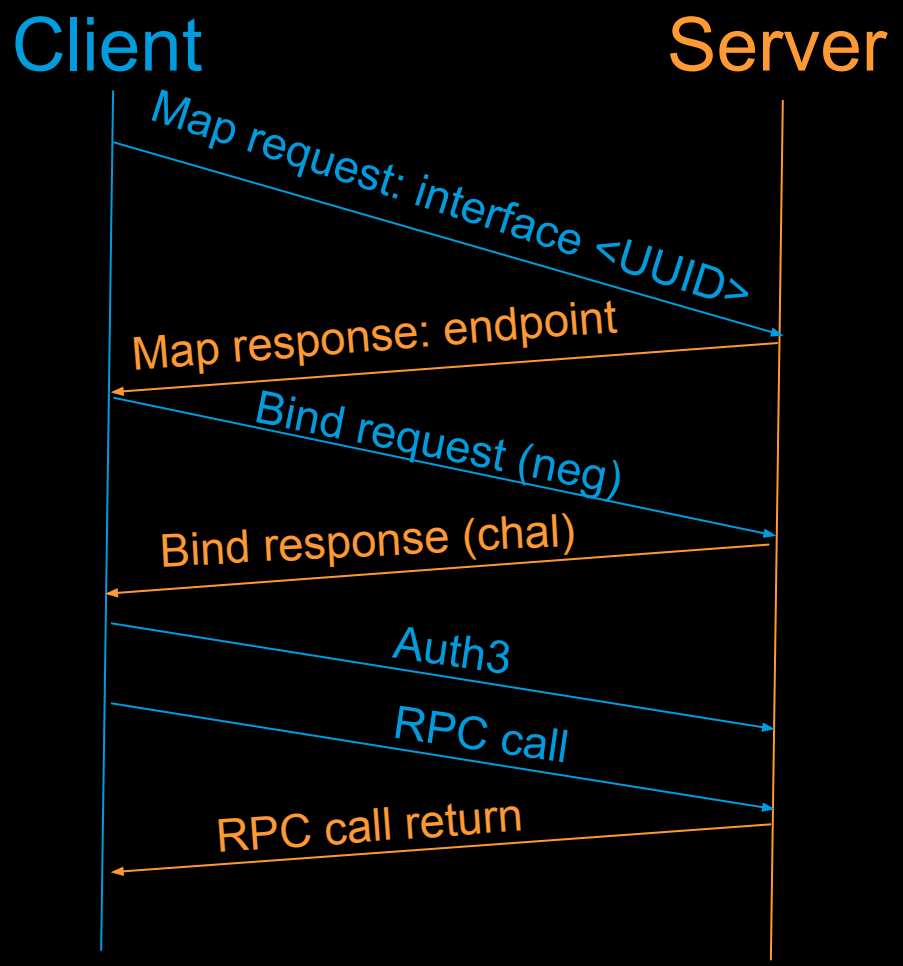
RPC → Auth → NTLM

Binding & Authentication

```
Bind: call_id: 66, Fragment: Single, 2 context items: WINREG V1.0 (32bit NDR), WINREG V1.0 (64bit NDR), NTLMSSP_NEGOTIATE
135 → 53743 [ACK] Seq=1 Ack=188 Win=64128 Len=0
Bind_ack: call_id: 66, Fragment: Single, max_xmit: 5840 max_recv: 5840, 2 results: Acceptance, Provider rejection, NTLMSSP_CHALLENGE
AUTH3: call_id: 66, Fragment: Single, NTLMSSP_AUTH, User: RESEARCH\Administrator
```



Message Flow (NTLM)



RPC Authentication

- RPC calls are unauthenticated by default

RPC Authentication

- RPC calls are unauthenticated by default
- RPC servers must tell the runtime they want authentication support
  - RpcServerRegisterAuthInfo

RPC Authentication

- RPC calls are unauthenticated by default
- RPC servers must tell the runtime they want authentication support
  - RpcServerRegisterAuthInfo
- Clients aren’t forced to authenticate
  - Unless server specify RPC_IF_ALLOW_SECURE_ONLY during registration

Server (side of) Authentication

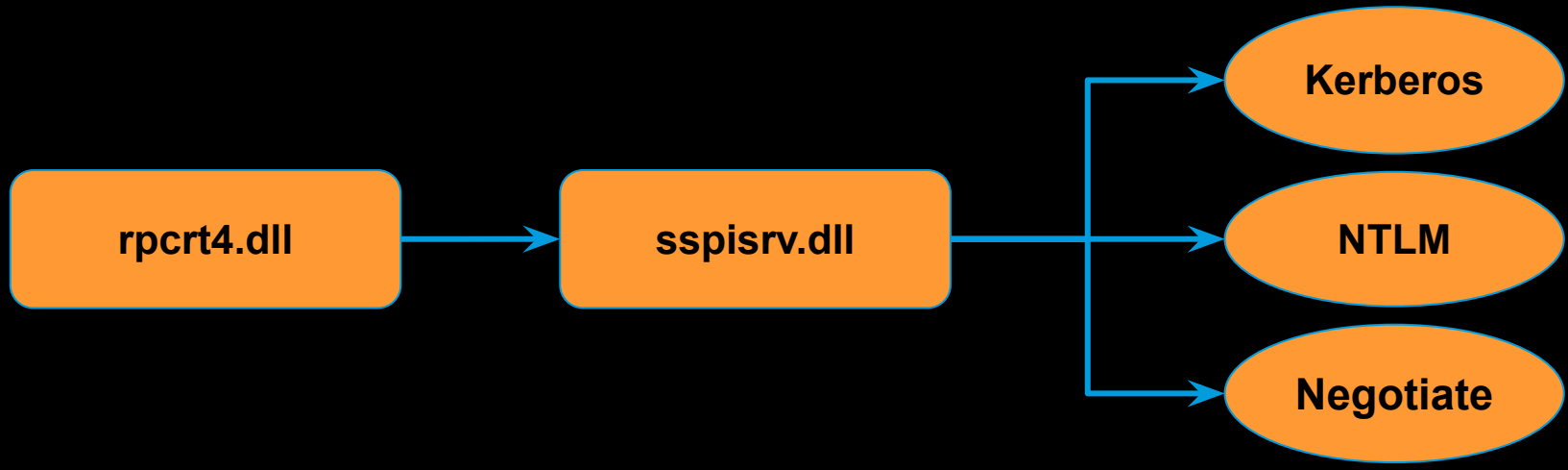
- Authentication in RPC is implemented with the Security Support Provider Interface (SSPI)

Server (side of) Authentication

- Authentication in RPC is implemented with the Security Support Provider Interface (SSPI)
- RPC servers wishing to use authentication must instruct the RPC runtime to load the corresponding SSPI

Server (side of) Authentication

- Authentication in RPC is implemented with the Security Support Provider Interface (SSPI)
- RPC servers wishing to use authentication must instruct the RPC runtime to load the corresponding SSPI



Client (side of) Authentication

```
RPC_STATUS RpcBindingSetAuthInfo(
  RPC_BINDING_HANDLE        Binding,
  RPC_CSTR                  ServerPrincName,
  unsigned long             AuthnLevel,
  unsigned long             AuthnSvc,
  RPC_AUTH_IDENTITY_HANDLE  AuthIdentity,
  unsigned long             AuthzSvc
);
```

Client (side of) Authentication

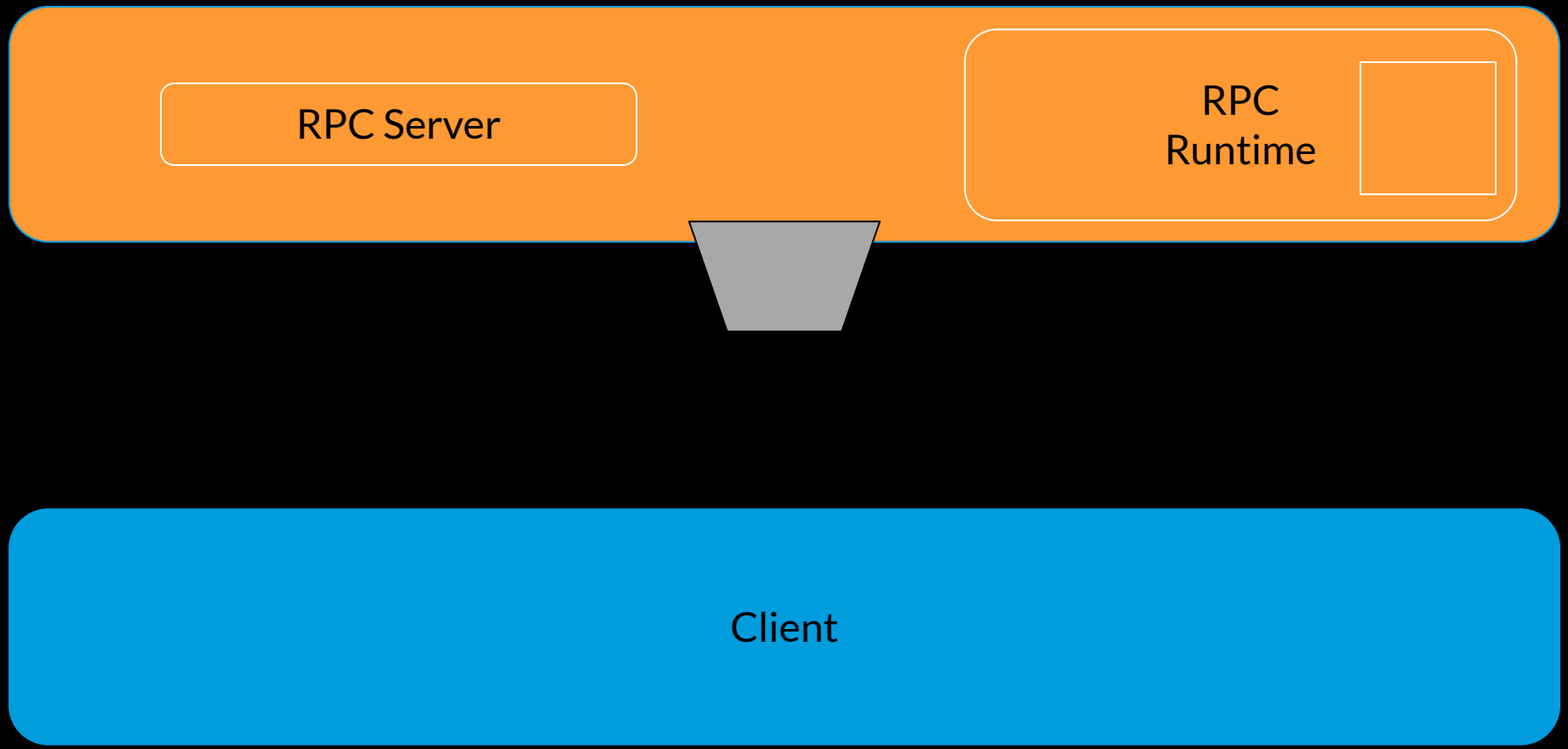
RPC session

Protocol

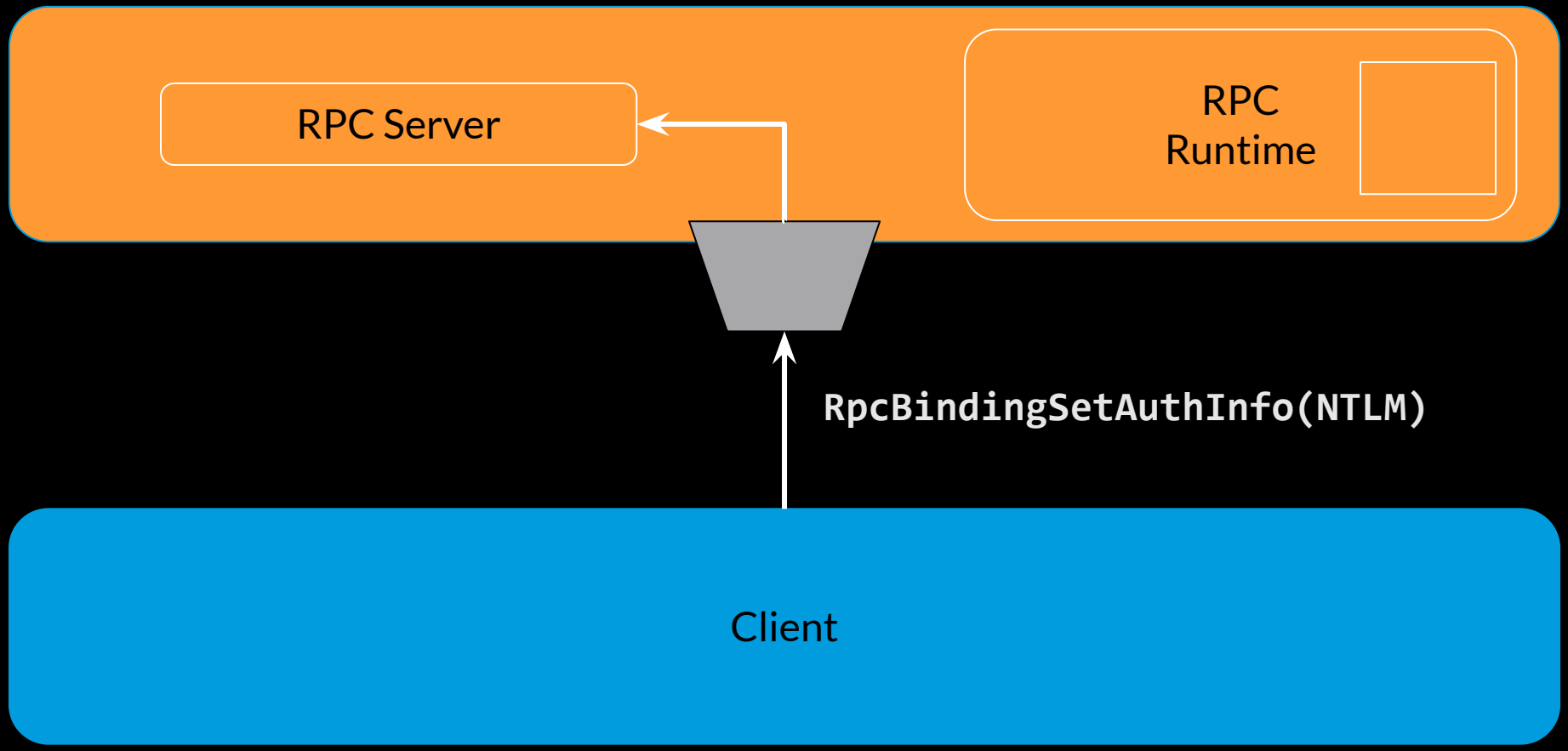
Credentials

```
RPC_STATUS RpcBindingSetAuthInfo(
  RPC_BINDING_HANDLE        Binding,
  RPC_CSTR                  ServerPrincName,
  unsigned long             AuthnLevel,
  unsigned long             AuthnSvc,
  RPC_AUTH_IDENTITY_HANDLE  AuthIdentity,
  unsigned long             AuthzSvc
);
```

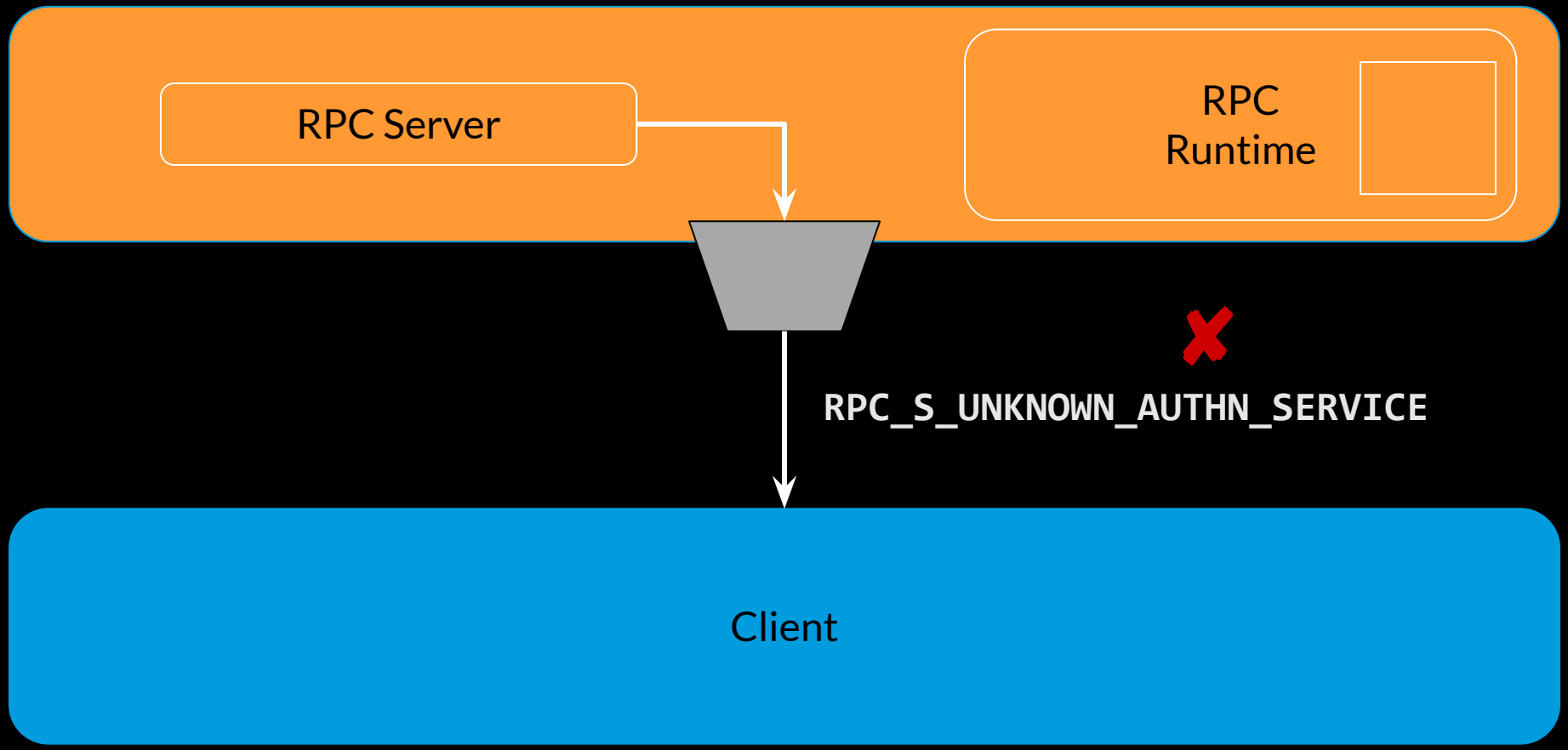
Server Authentication



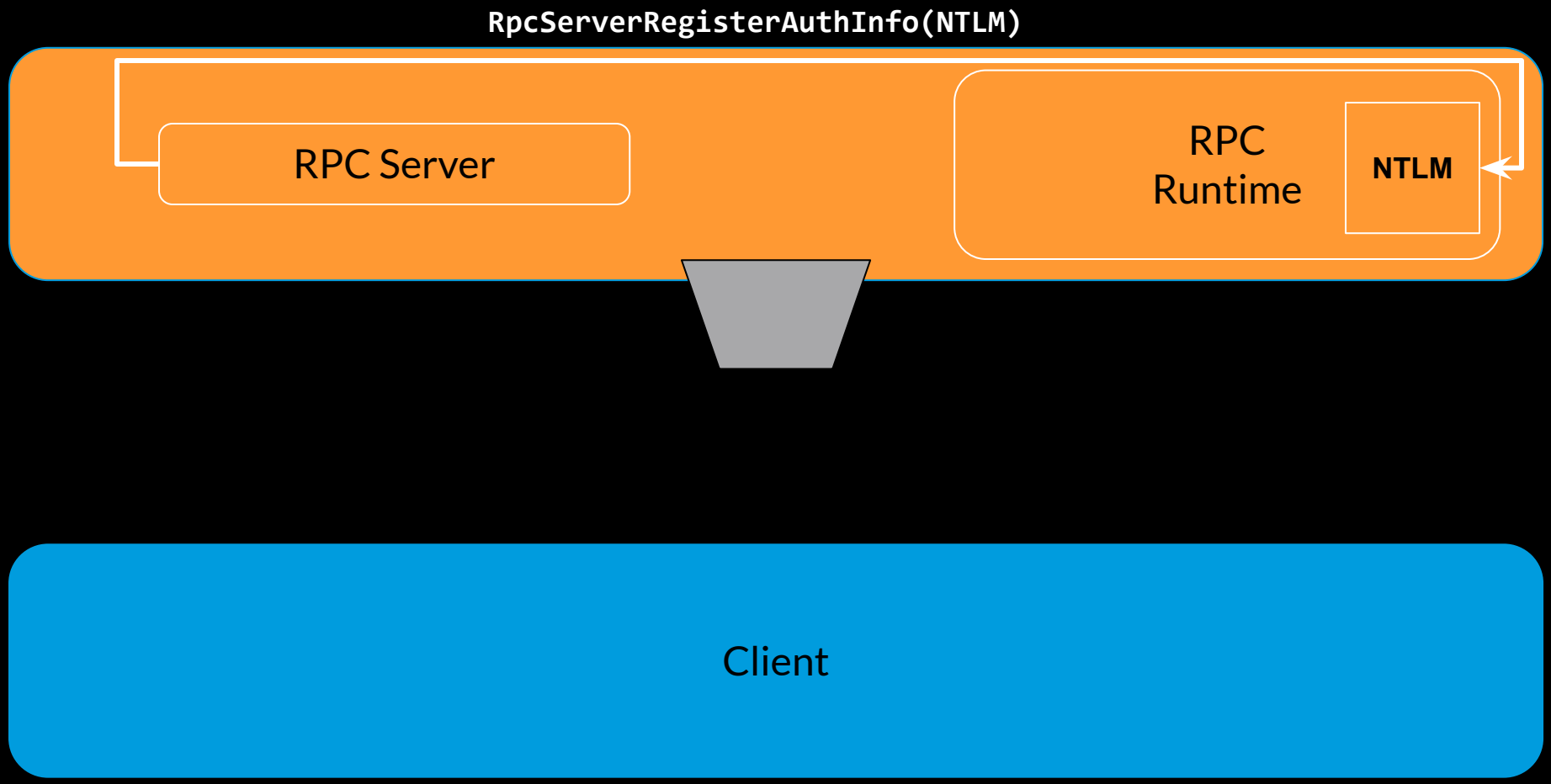
Server Authentication



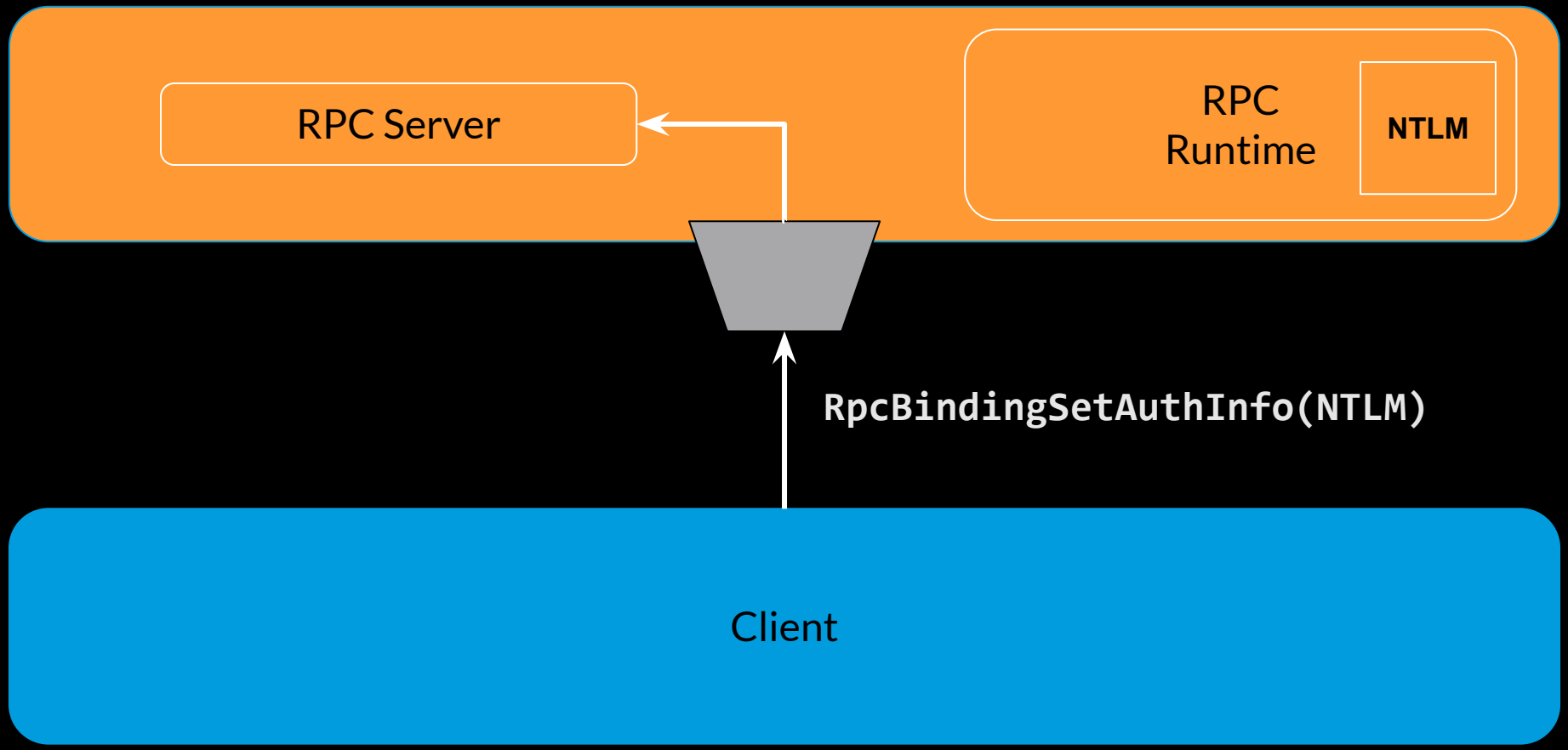
Server Authentication



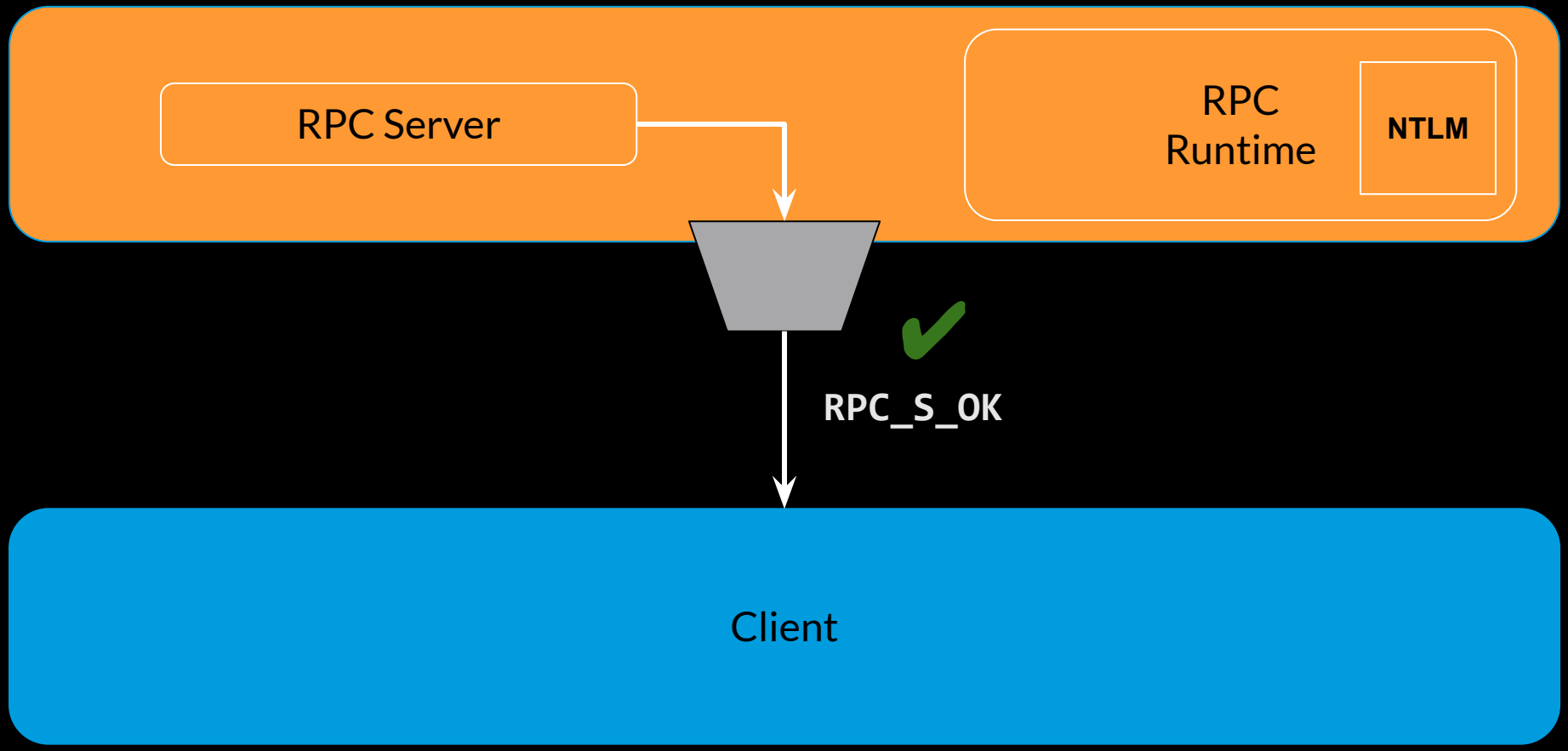
Server Authentication



Server Authentication



Server Authentication



Security

```
RPC_STATUS RpcBindingSetAuthInfo(
  RPC_BINDING_HANDLE        Binding,
  RPC_CSTR                  ServerPrincName,
  unsigned long             AuthnLevel,
  unsigned long             AuthnSvc,
  RPC_AUTH_IDENTITY_HANDLE  AuthIdentity,
  unsigned long             AuthzSvc
);
```

Authentication Levels

RPC_C_AUTHN_LEVEL_NONE	No authentication
RPC_C_AUTHN_LEVEL_CONNECT	Authenticate when establishing a binding
RPC_C_AUTHN_LEVEL_CALL	Authenticate on each call
RPC_C_AUTHN_LEVEL_PKT	Authenticate on each packet and verify that all the data was received
RPC_C_AUTHN_LEVEL_PKT_INTEGRITY	Authenticate & verify data integrity
RPC_C_AUTHN_LEVEL_PKT_PRIVACY	Authenticate & encrypt all data

Authentication Levels

RPC_C_AUTHN_LEVEL_NONE	No authentication
RPC_C_AUTHN_LEVEL_CONNECT	Authenticate when establishing a binding
RPC_C_AUTHN_LEVEL_CALL	Authenticate on each call
RPC_C_AUTHN_LEVEL_PKT	Authenticate on each packet and verify that all the data was received
RPC_C_AUTHN_LEVEL_PKT_INTEGRITY	Authenticate & verify data integrity
RPC_C_AUTHN_LEVEL_PKT_PRIVACY	Authenticate & encrypt all data

Authentication Levels

RPC_C_AUTHN_LEVEL_NONE	No authentication
RPC_C_AUTHN_LEVEL_CONNECT	Authenticate when establishing a binding
RPC_C_AUTHN_LEVEL_CALL	Authenticate on each call
RPC_C_AUTHN_LEVEL_PKT	Authenticate on each packet and verify that all the data was received
RPC_C_AUTHN_LEVEL_PKT_INTEGRITY	Authenticate & verify data integrity
RPC_C_AUTHN_LEVEL_PKT_PRIVACY	Authenticate & encrypt all data

Let’s Find Some Victims

Find RPC PEs

Imports rpcrt4.dll

RpcServerRegisterAuthInfoW /
RpcBindingSetAuthInfoA

Let’s Find Some Victims

Find RPC PEs

Parse authentication info

Imports rpcrt4.dll

RpcServerRegisterAuthInfoW / RpcBindingSetAuthInfoA

AuthnSvc -> NTLM/Negotiate

AuthnLevel -> Connect

Let’s Find Some Victims

Find RPC PEs

Imports rpcrt4.dll

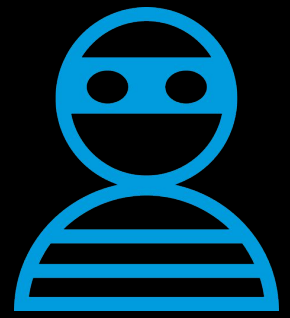
RpcServerRegisterAuthInfoW / RpcBindingSetAuthInfoA

Parse authentication info

AuthnSvc -> NTLM/Negotiate

AuthnLevel -> Connect

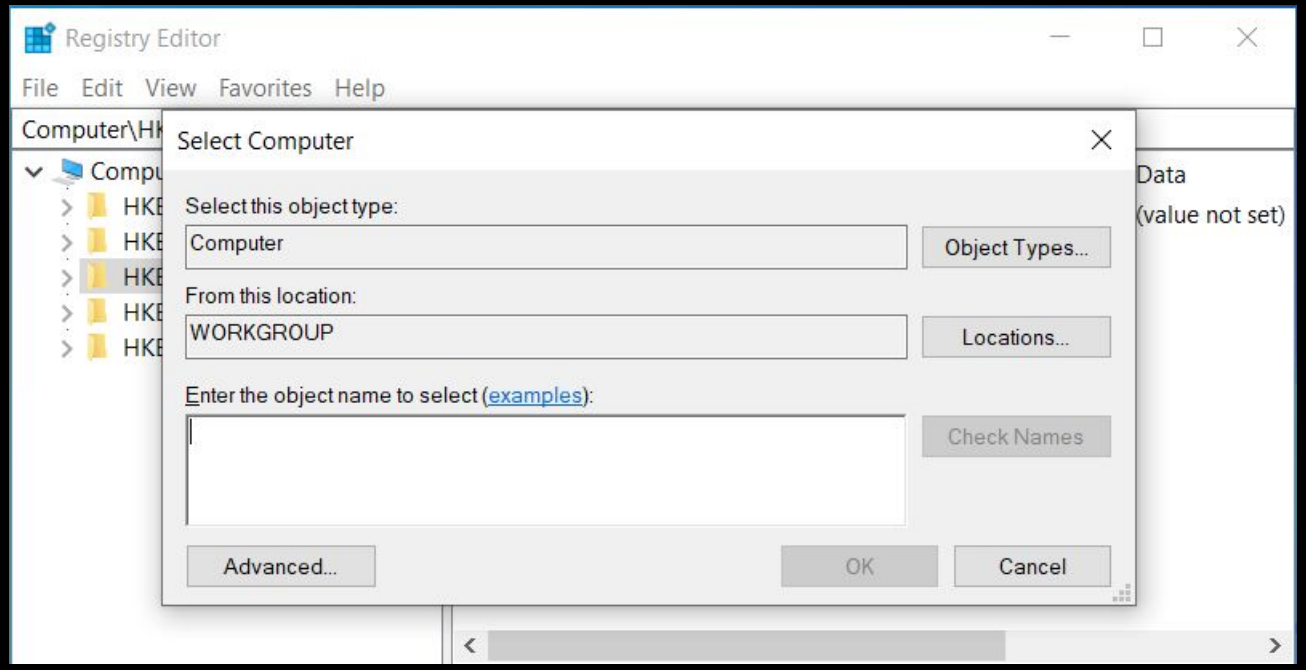
Profit?



Windows Remote Registry

RPC Interface: MS-RRP

- Interface UUID: {338CD001-2244-31F1-AAAA-900038001003}
- Well known endpoint: \PIPE\winreg
- RPC server implemented in regsvc.dll
- RPC client implemented in advapi32.dll



https://learn.microsoft.com/en-us/openspecs/windows_protocols/ms-rrp/0fa3191d-bb79-490a-81bd-54c2601b7a78

Why Is It Special?

```
v12 = RpcBindingSetAuthInfoA(Binding, (RPC_CSTR)&ServerPrincName, 2u, 0xAu, 0i64, 0);
```

```
v12 = RpcBindingSetAuthInfoW(Binding, v19, 6u, 9u, AuthzSvc, (unsigned int)AuthzSvc);
```

Why Is It Special?

```
v12 = RpcBindingSetAuthInfoA(Binding, (RPC_CSTR)&ServerPrincName, 2u, 0xAu, 0i64, 0);
```

```
v12 = RpcBindingSetAuthInfoW(Binding, v19, 6u, 9u, AuthzSvc, (unsigned int)AuthzSvc);
```

Why Is It Special?

```
v12 = RpcBindingSetAuthInfoA(Binding, (RPC_CSTR)&ServerPrincName, 2u, 0xAu, 0i64, 0);
```

RPC_C_AUTHN_LEVEL_CONNECT

```
v12 = RpcBindingSetAuthInfoW(Binding, v19, 6u, 9u, AuthzSvc, (unsigned int)AuthzSvc);
```

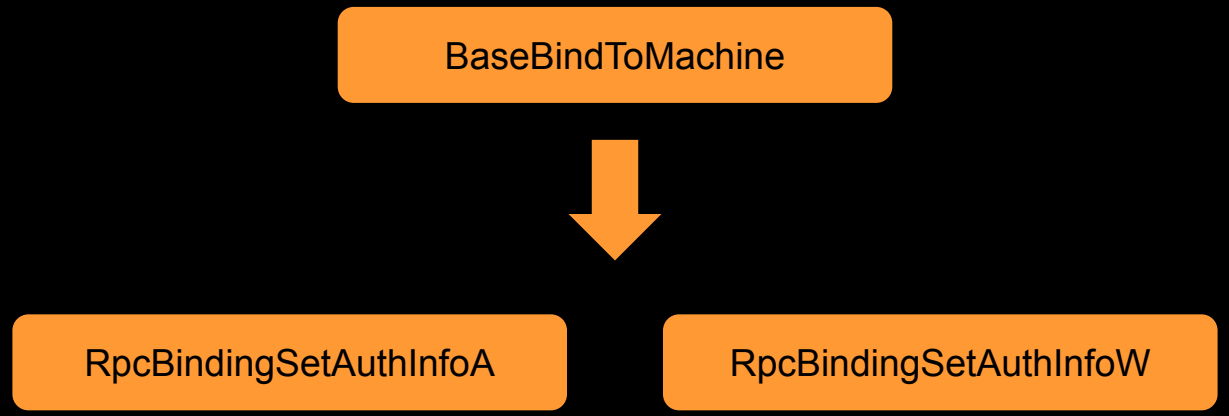
RPC_C_AUTHN_LEVEL_PKT_PRIVACY

Callstack

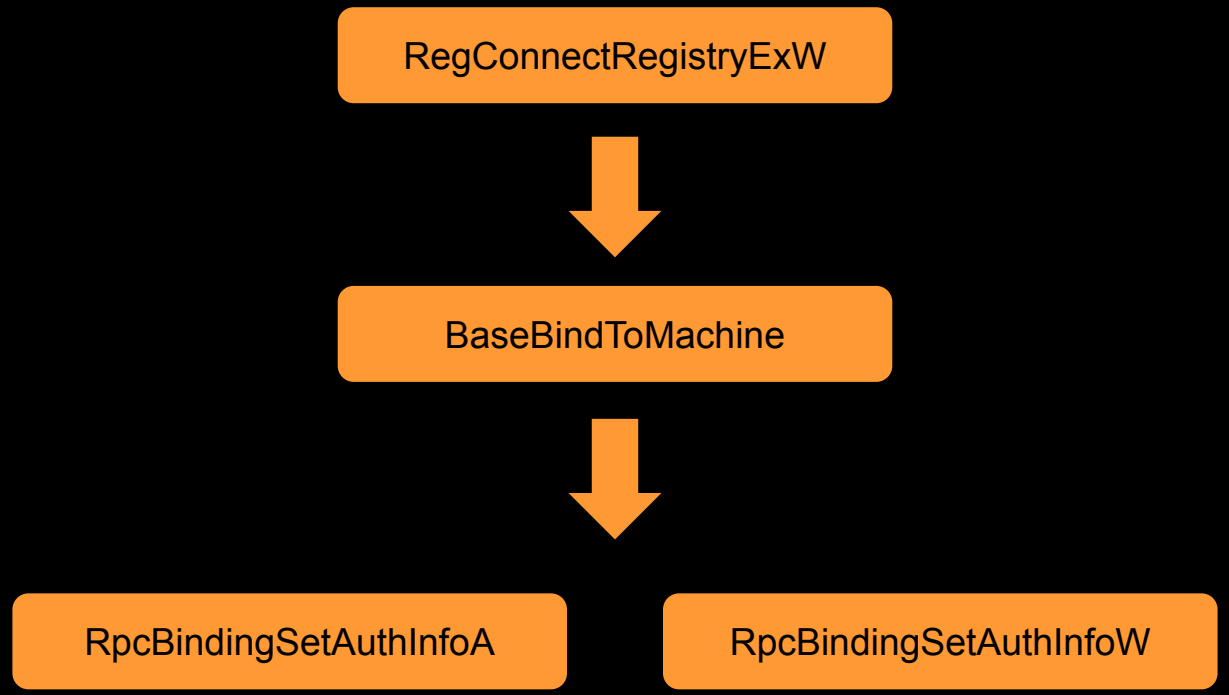
RpcBindingSetAuthInfoA

RpcBindingSetAuthInfoW

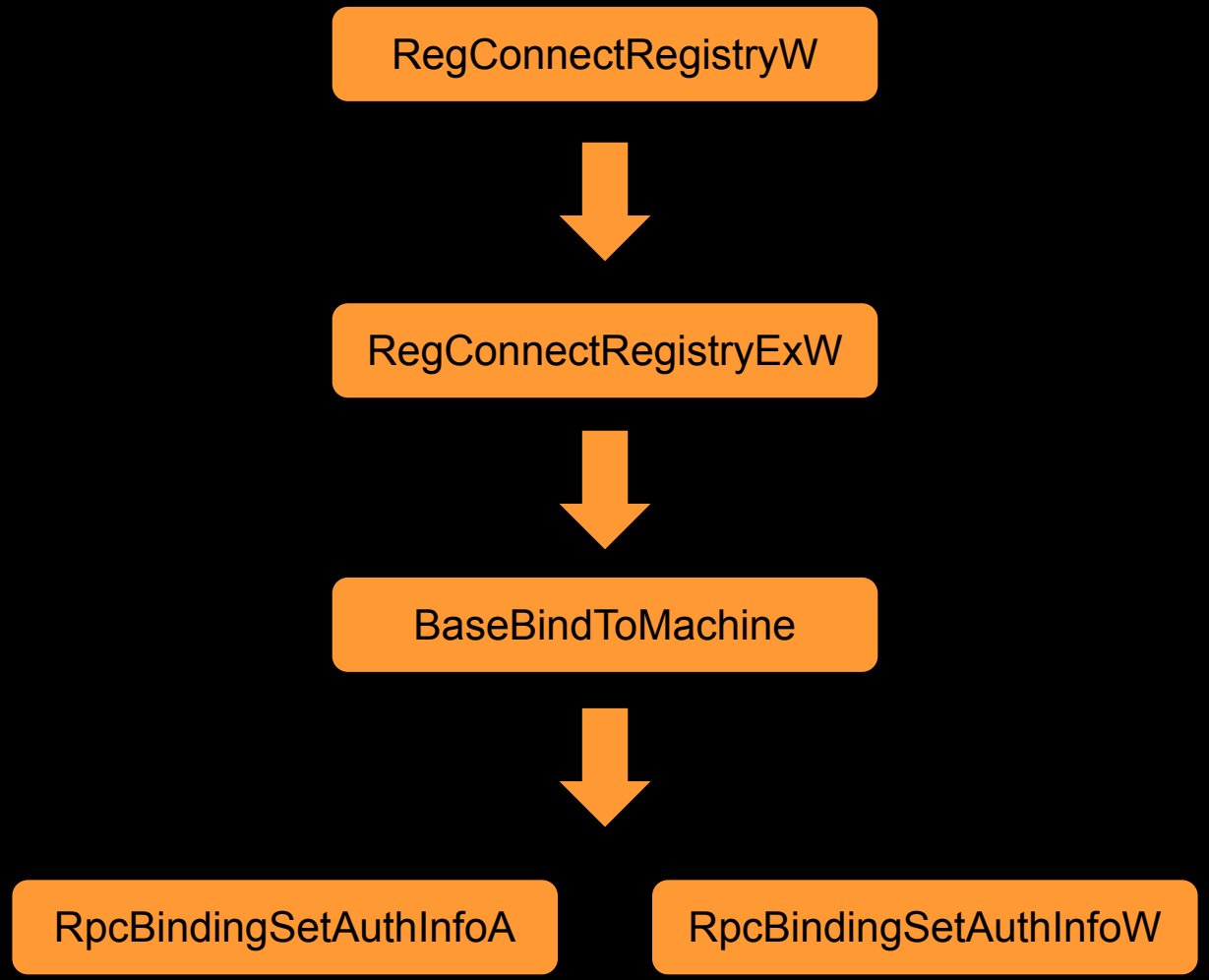
Callstack


BaseBindToMachine
RpcBindingSetAuthInfoA
RpcBindingSetAuthInfoW

Callstack


RegConnectRegistryExW
BaseBindToMachine
RpcBindingSetAuthInfoA
RpcBindingSetAuthInfoW

Callstack


RegConnectRegistryW
RegConnectRegistryExW
BaseBindToMachine
RpcBindingSetAuthInfoA
RpcBindingSetAuthInfoW

Transport Fallback

```
BindingBuilder = RegConn_np;
v5 = function_array;
v6 = 1;
v7 = a4;
v8 = (a3 & 1) == 0;
do
{
  while ( 1 )
  {
    while ( 1 )
    {
      if ( ((unsigned int (__fastcall *)(STRSAFE_PCNZWCH, RPC_BINDING_HANDLE *))BindingBuilder)(pszSrc, &Binding) )
        goto LABEL_49;
      if ( BindingBuilder == RegConn_np )
        break;
      if ( RpcEpResolveBinding(Binding, &unk_180070560) )
        goto LABEL_44;
      v12 = RpcBindingSetAuthInfoA(Binding, (RPC_CSTR)&ServerPrincName, 2u, 0xAu, 0i64, 0);
```

Transport Fallback

- Tries to connect via RPC over SMB (ncacn_np) — \PIPE\winreg
- On failure, tries to use other transport protocols:
  - SPX — ncacn_spx
  - TCP — ncacn_ip_tcp
  - NetBEUI — ncacn_nb_nb
  - NetBIOS — ncacn_nb_tcp
  - IPX — ncacn_nb_ipx

Transport Fallback

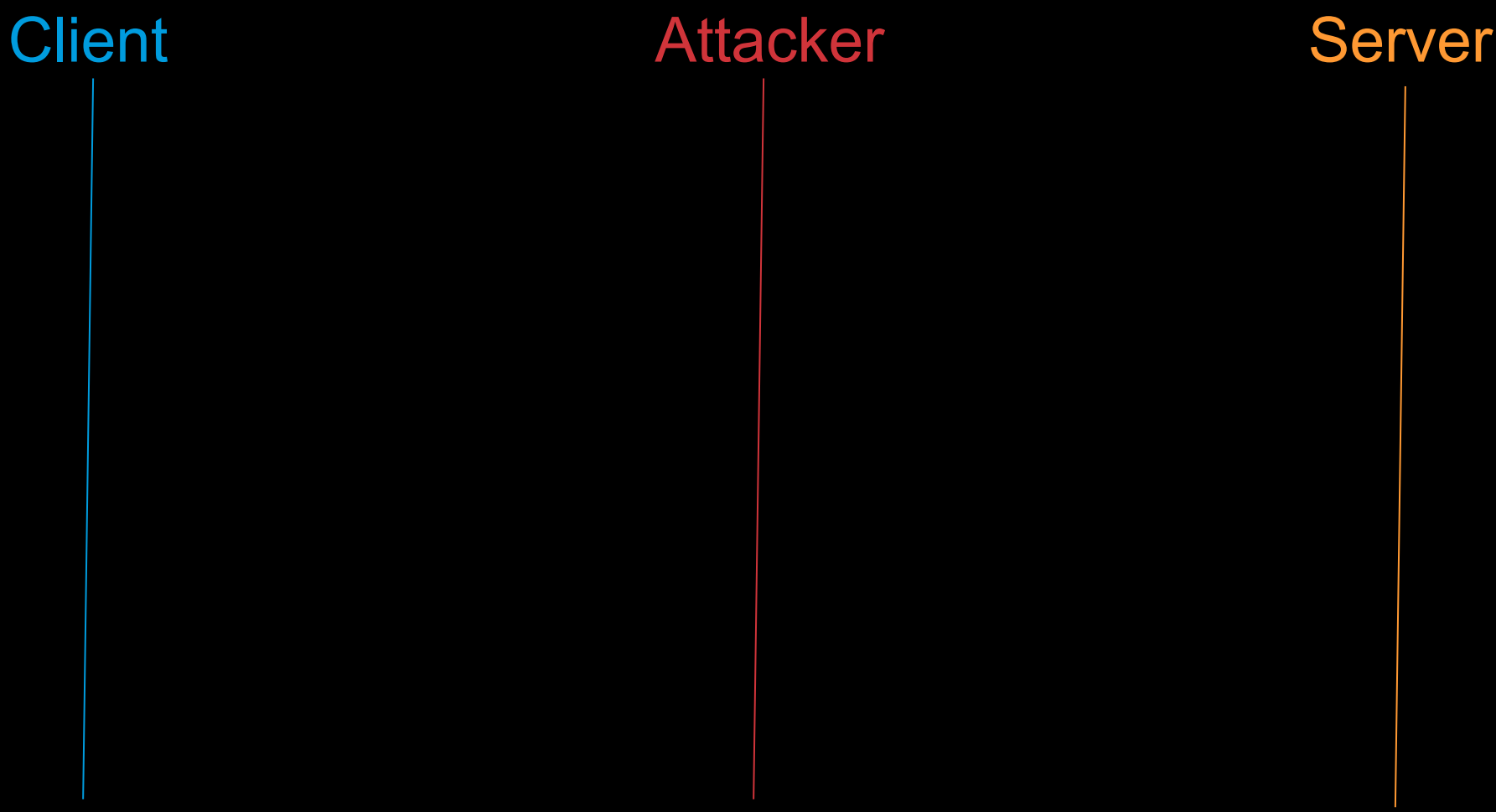
- Tries to connect via RPC over SMB (ncacn_np) — \PIPE\winreg
- On failure, tries to use other transport protocols:
  - SPX — ncacn_spx OBSOLETE
  - TCP — ncacn_ip_tcp
  - NetBEUI — ncacn_nb_nb OBSOLETE
  - NetBIOS — ncacn_nb_tcp OBSOLETE
  - IPX — ncacn_nb_ipx OBSOLETE

Transport Fallback

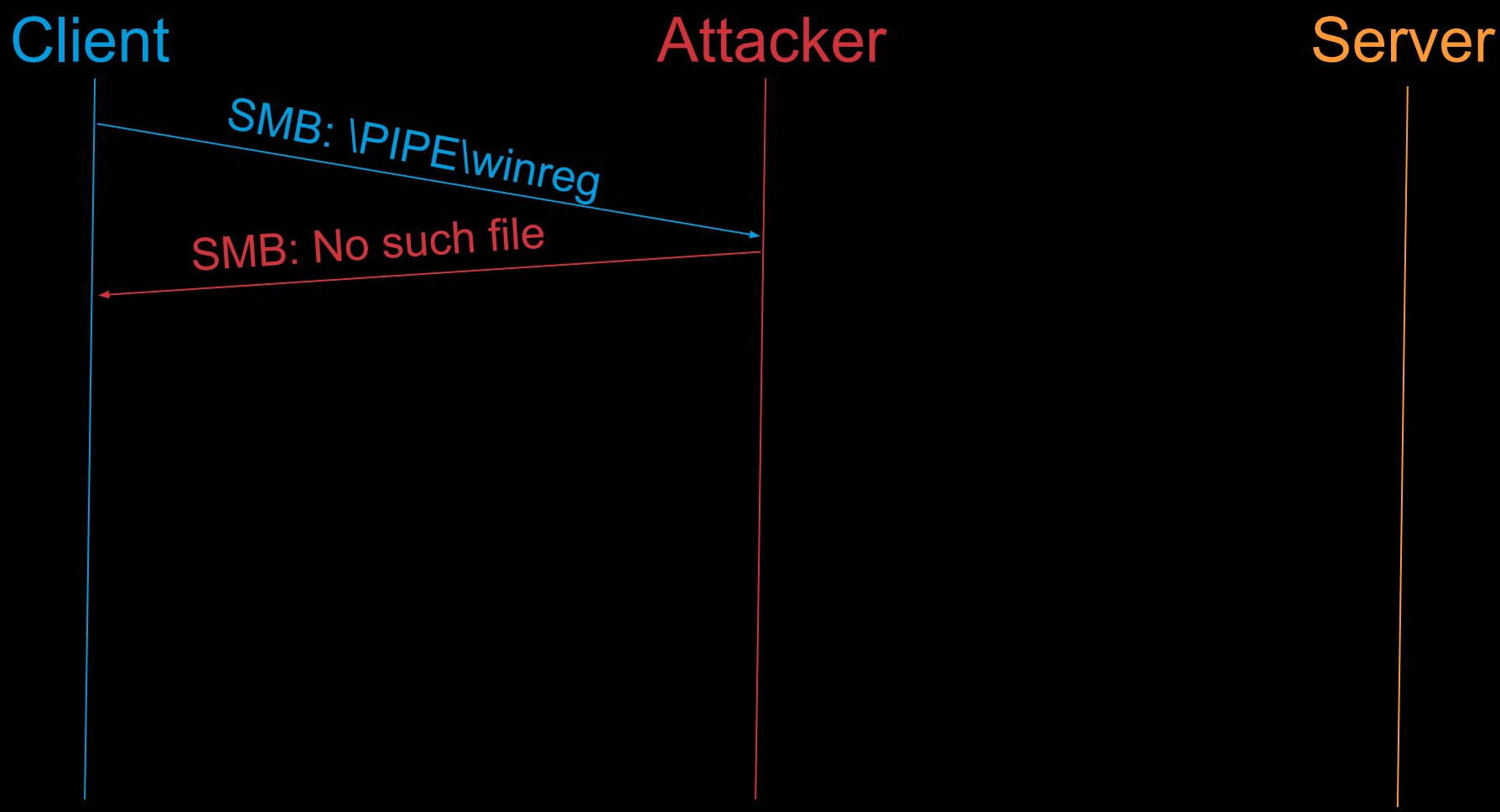
- Tries to connect via RPC over SMB (ncacn_np) — \PIPE\winreg
- On failure, tries to use other transport protocols:
  - SPX — ncacn_spx OBSOLETE
  - TCP — ncacn_ip_tcp
  - NetBEUI — ncacn_nb_nb OBSOLETE
  - NetBIOS — ncacn_nb_tcp OBSOLETE
  - IPX — ncacn_nb_ipx OBSOLETE

SMB transport is PKT_PRIVACY, fallbacks are CONNECT

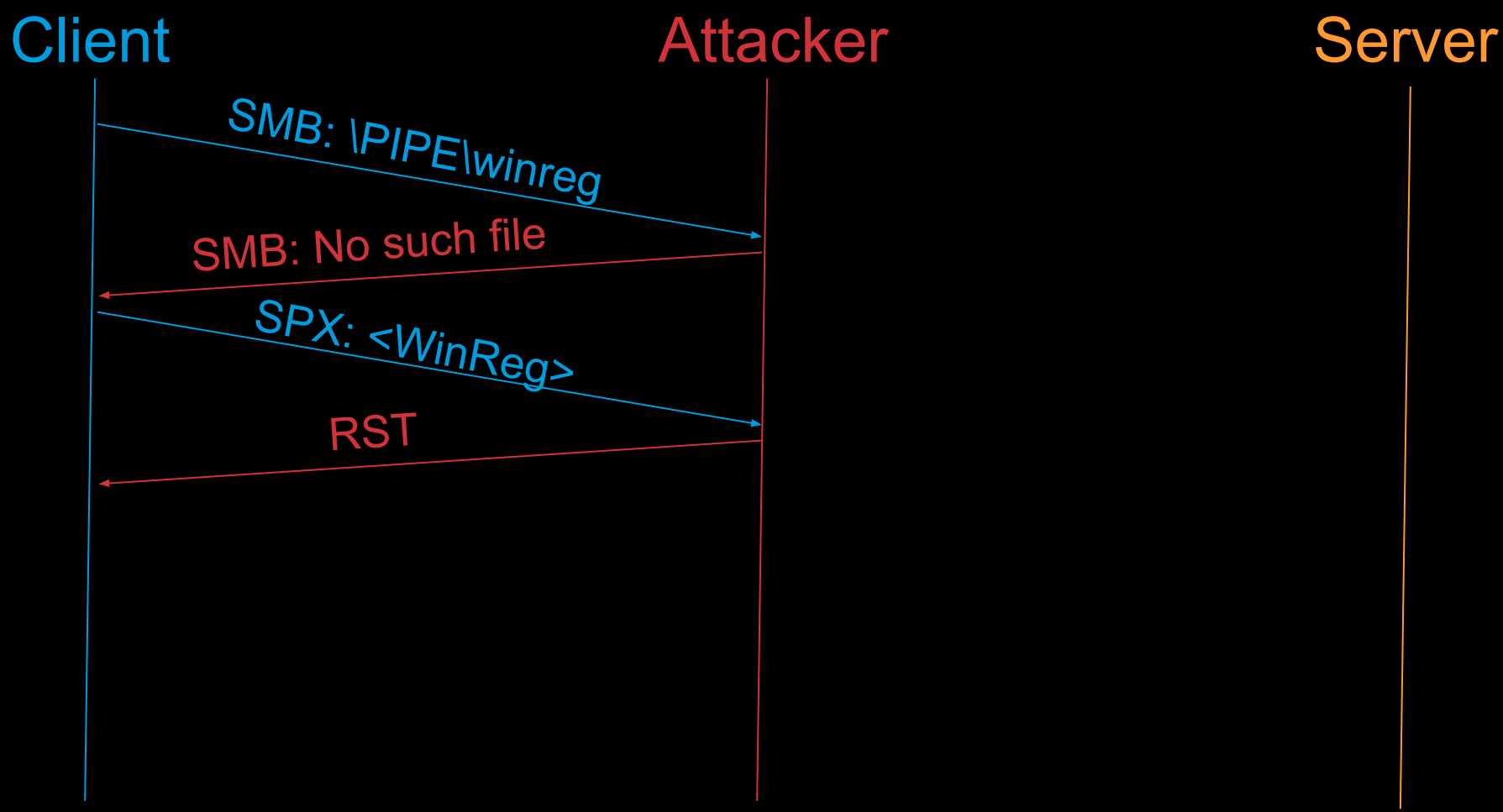
Attack Chain



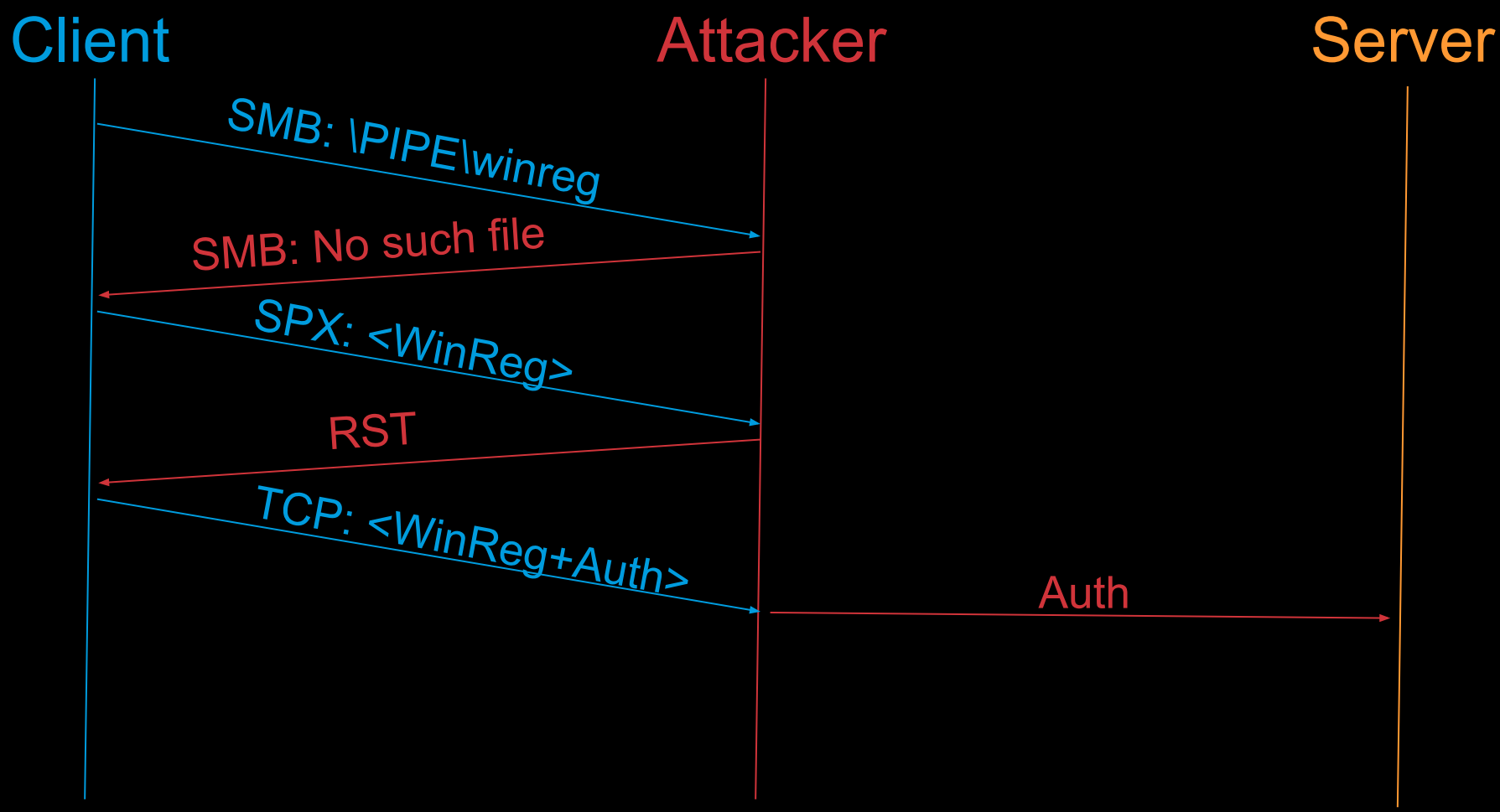
Attack Chain



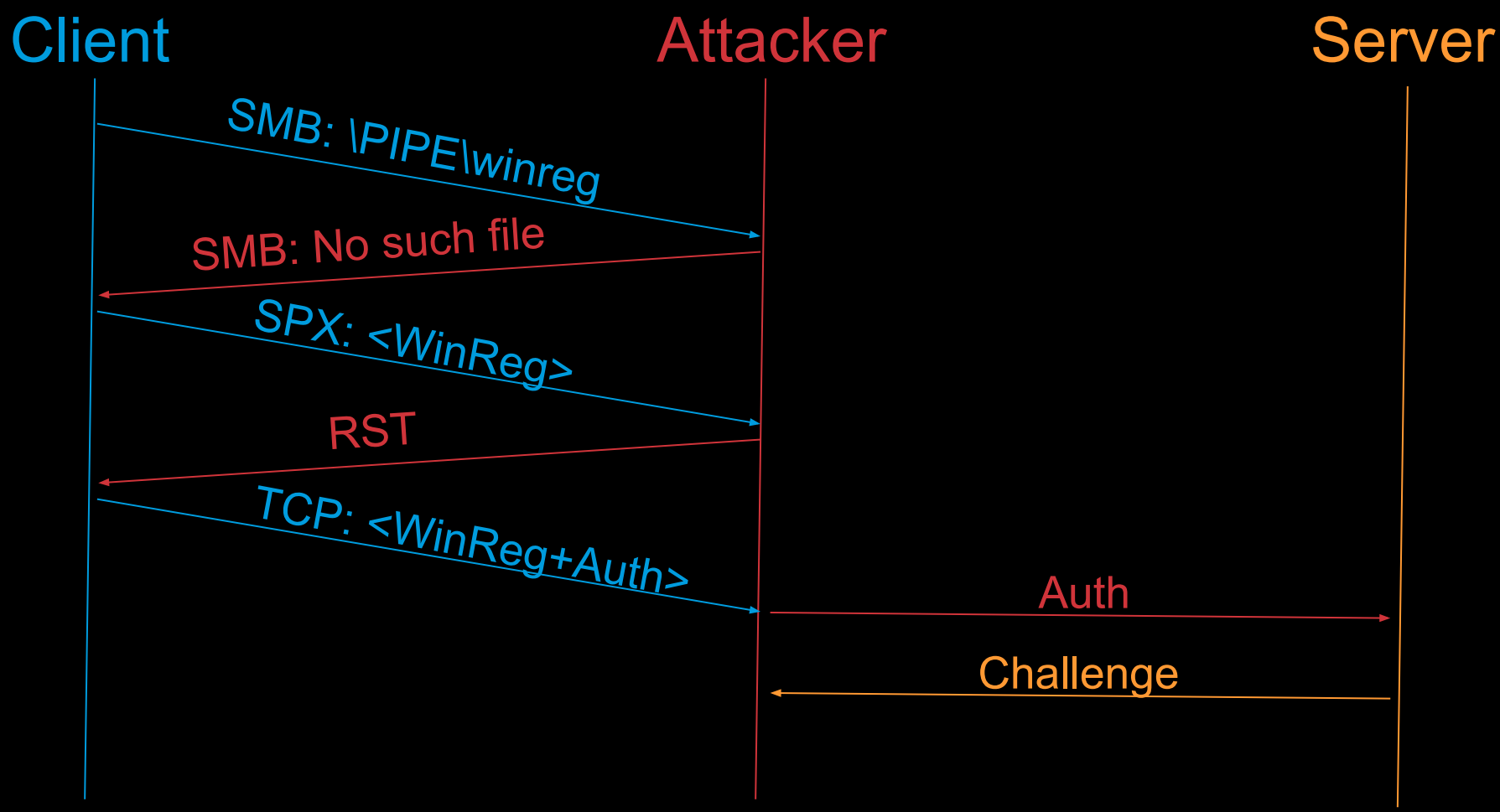
Attack Chain



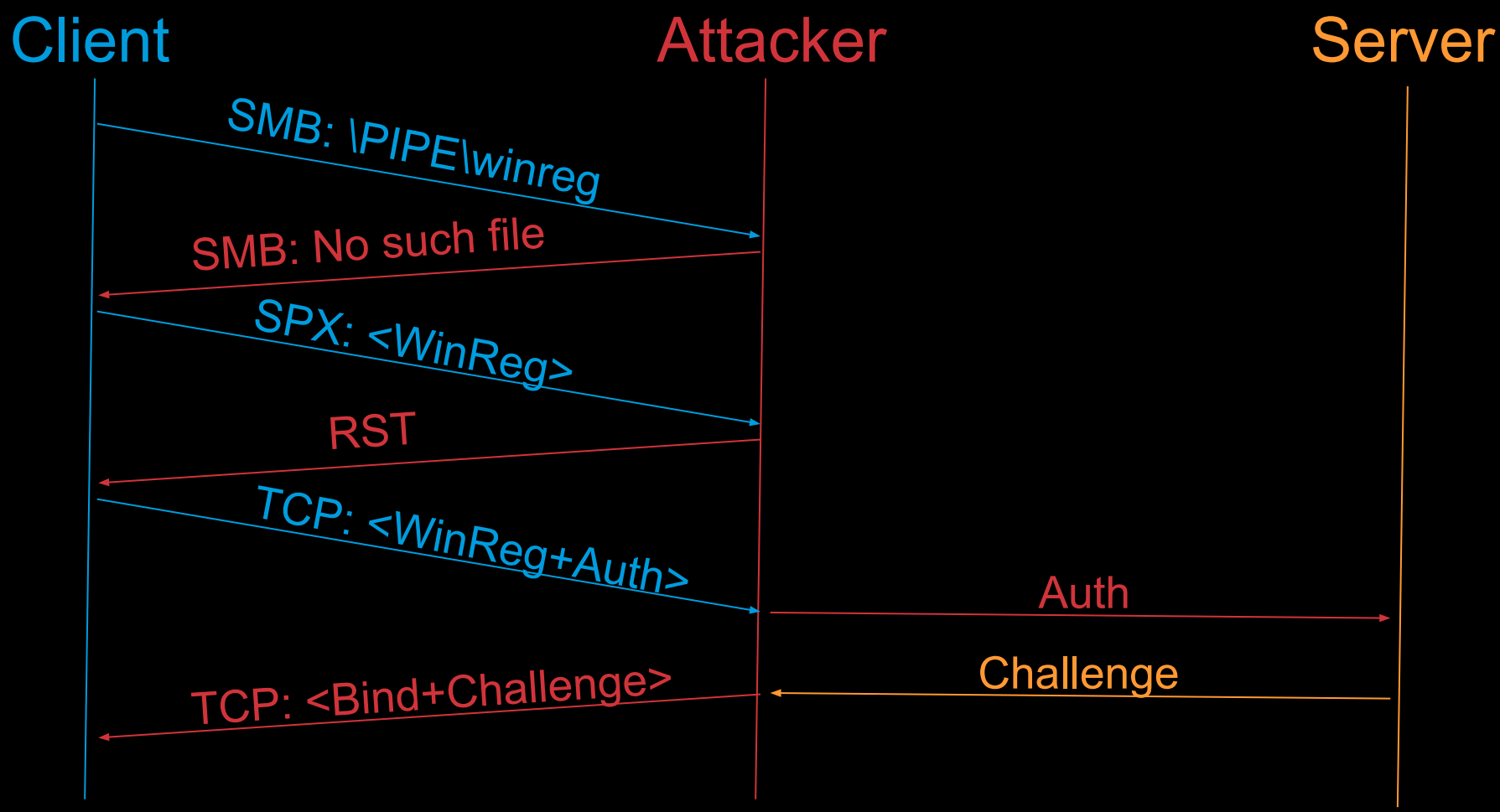
Attack Chain



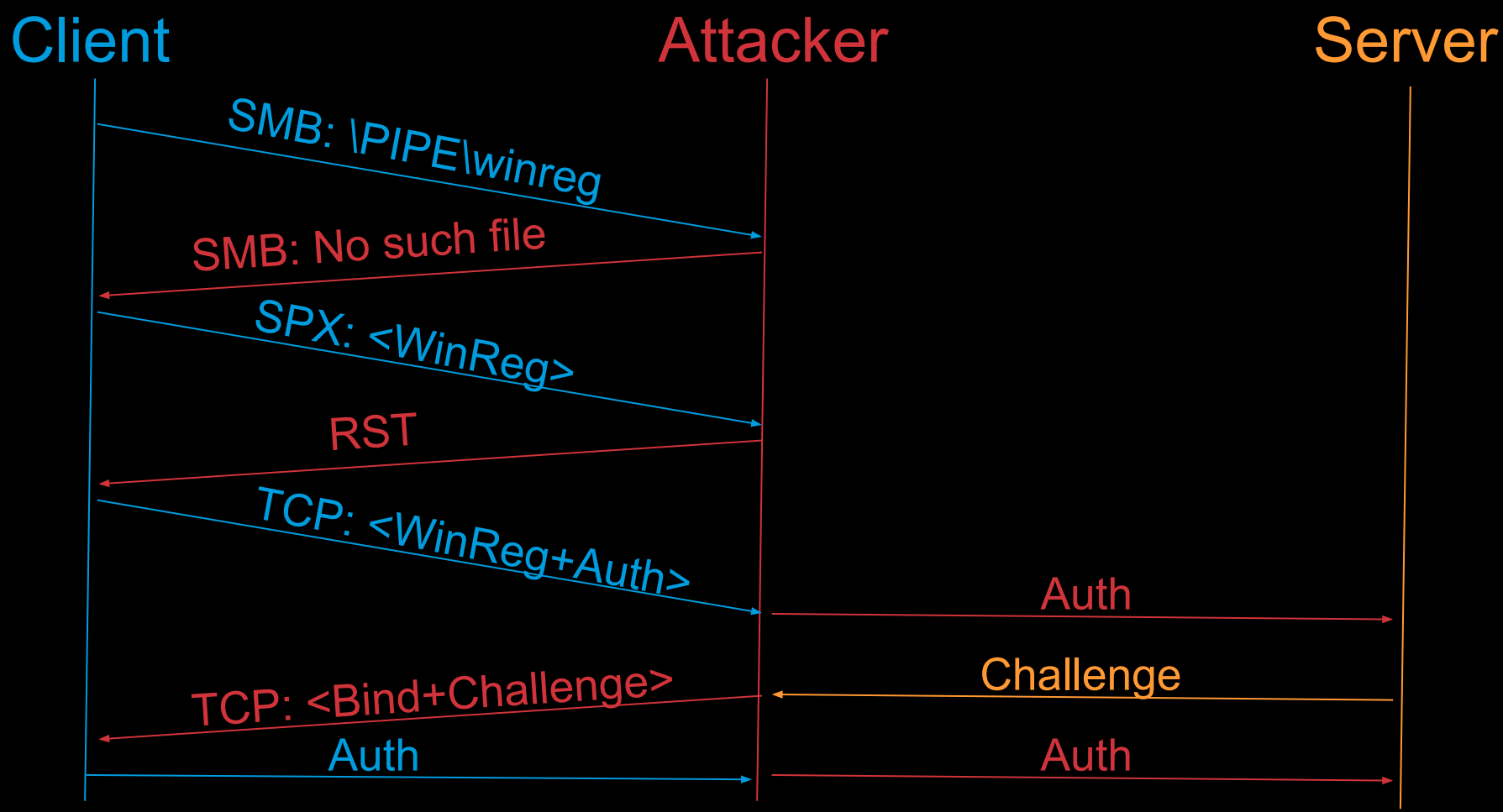
Attack Chain



Attack Chain



Attack Chain



Relay Targets

Can’t relay to RPC servers that require PKT_PRIVACY

Relay Targets

Can’t relay to RPC servers that require PKT_PRIVACY

- Encryption is derived from shared secret — the password
- SCM, Task scheduler

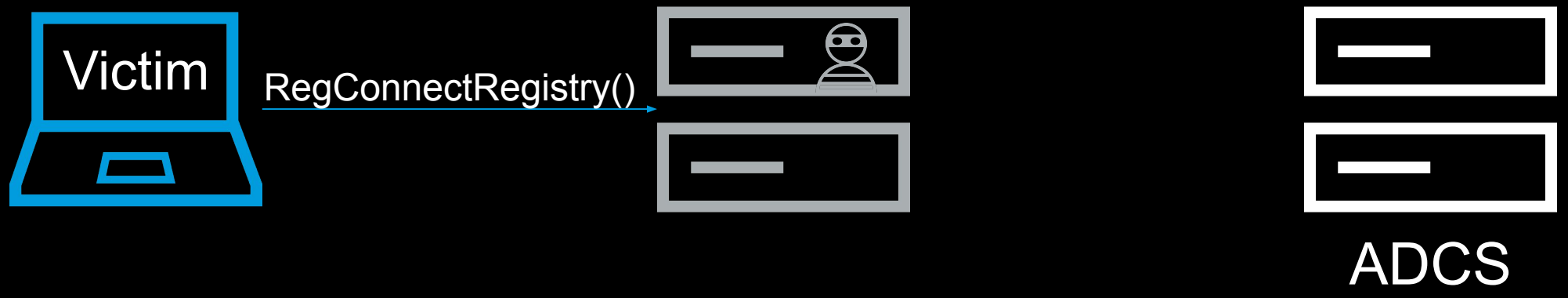
Relay Targets

Can’t relay to RPC servers that require PKT_PRIVACY

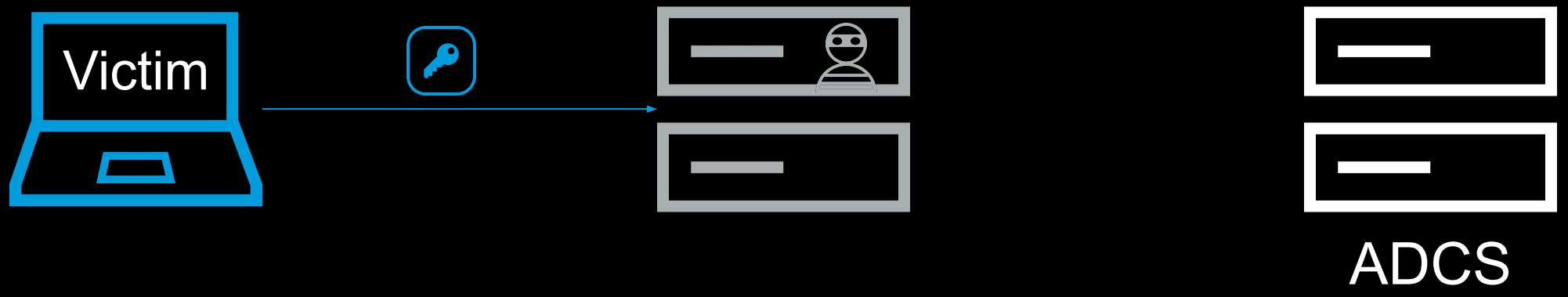
- Encryption is derived from shared secret — the password
- SCM, Task scheduler
- Winreg server as well, surprisingly

```
v5 = RpcServerInqCallAttributesW(0i64, &RpcCallAttributes);
if ( v5 != 1746 )                               // RPC_S_BINDING_HAS_NO_AUTH
{
  if ( v5 )
    RpcRaiseException(v5);
  if ( RpcCallAttributes.AuthenticationLevel < 6 )// RPC_C_AUTHN_LEVEL_PKT_PRIVACY
    RpcRaiseException(5);
}
```

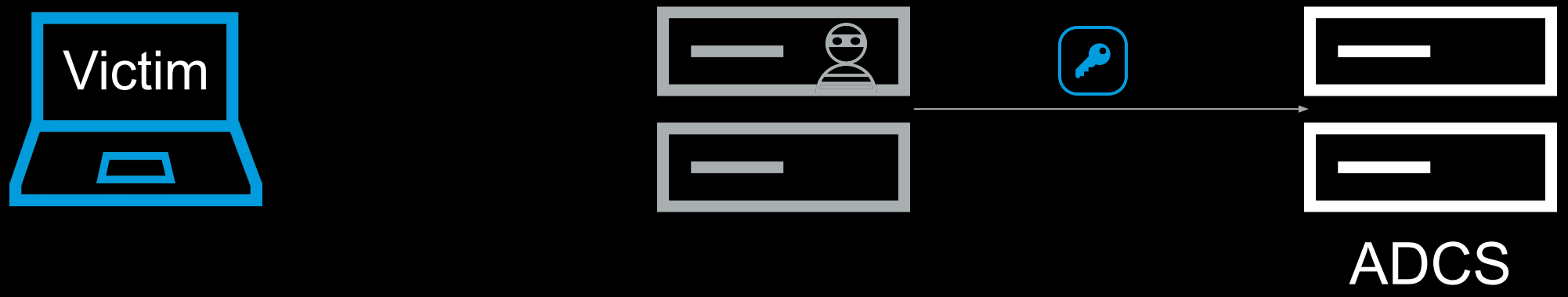
ADCS Relay



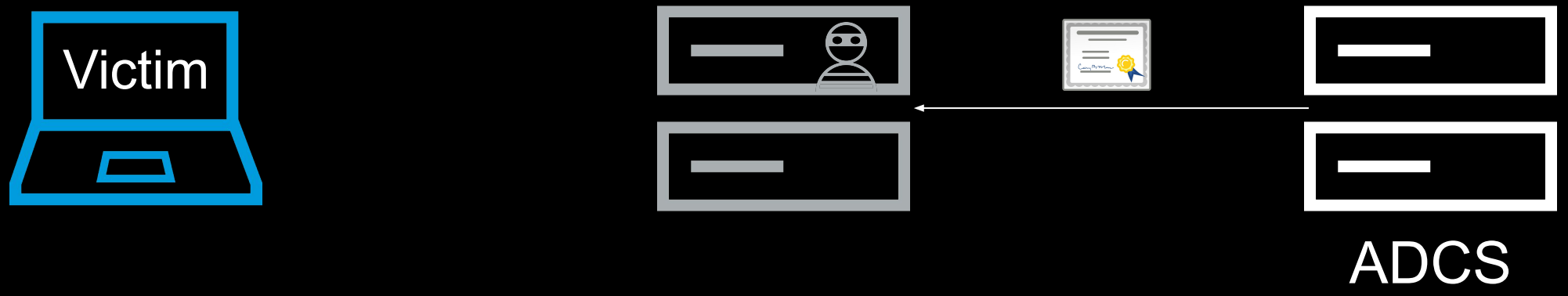
ADCS Relay



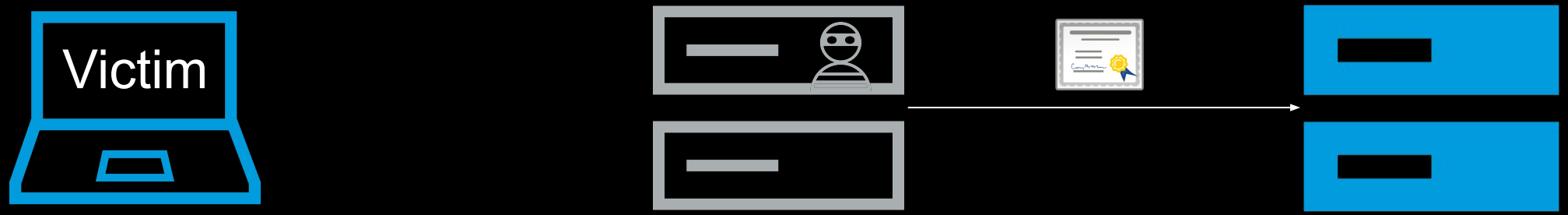
ADCS Relay



ADCS Relay



ADCS Relay



Demo

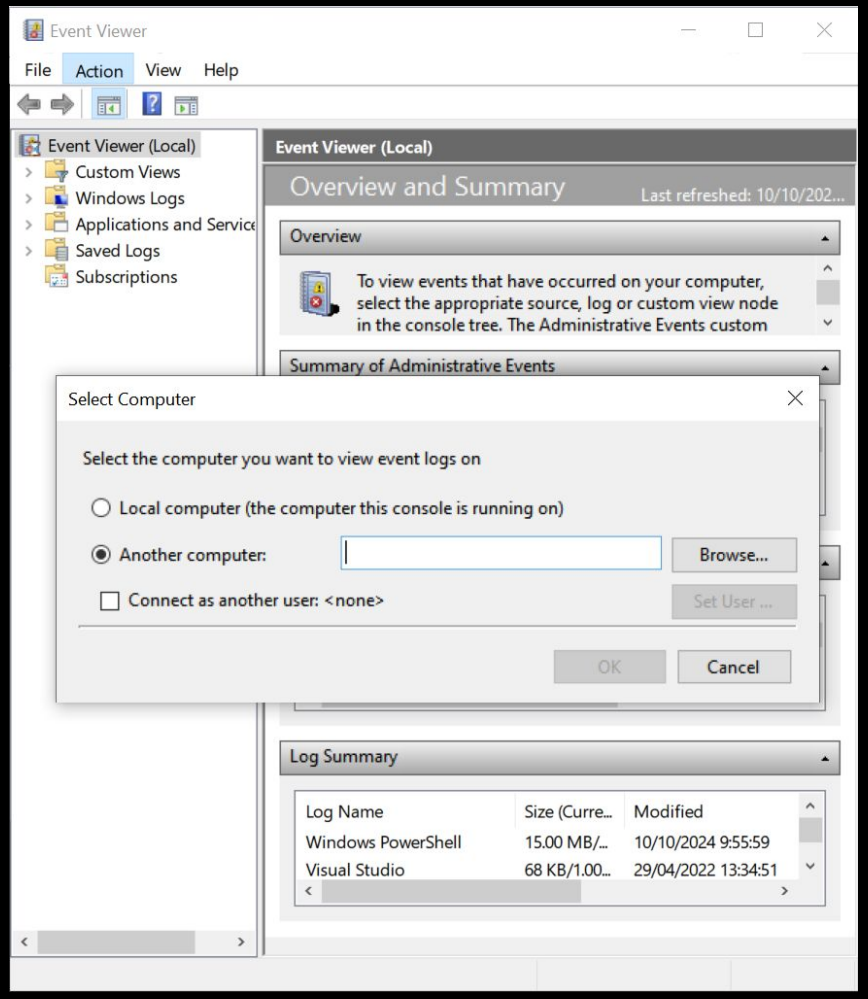
Potential Impact

Vulnerability is in Windows API — who the hell calls it?

- regedit
- ADCS
- Certutil
- DFS Namespaces
- taskview/taskkill
- Windows Event Viewer

Event Viewer?

- Can query logs remotely via RPC MS-EVEN[6]
  - Quick Google search shows people actually use it to view logs remotely
- Implemented as an .msc for mmc.exe
- Most logic is in els.dll

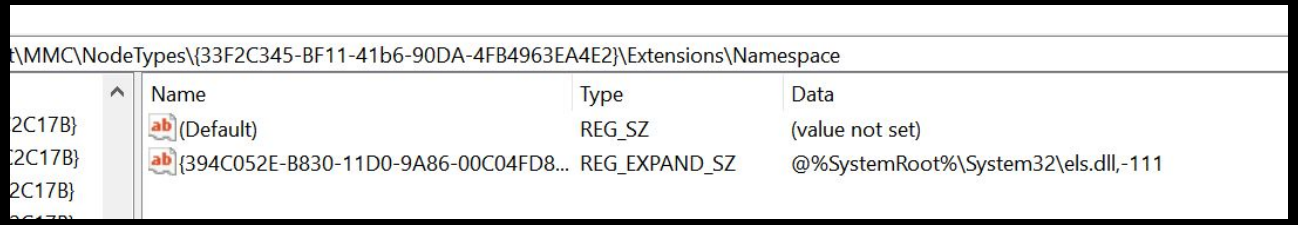


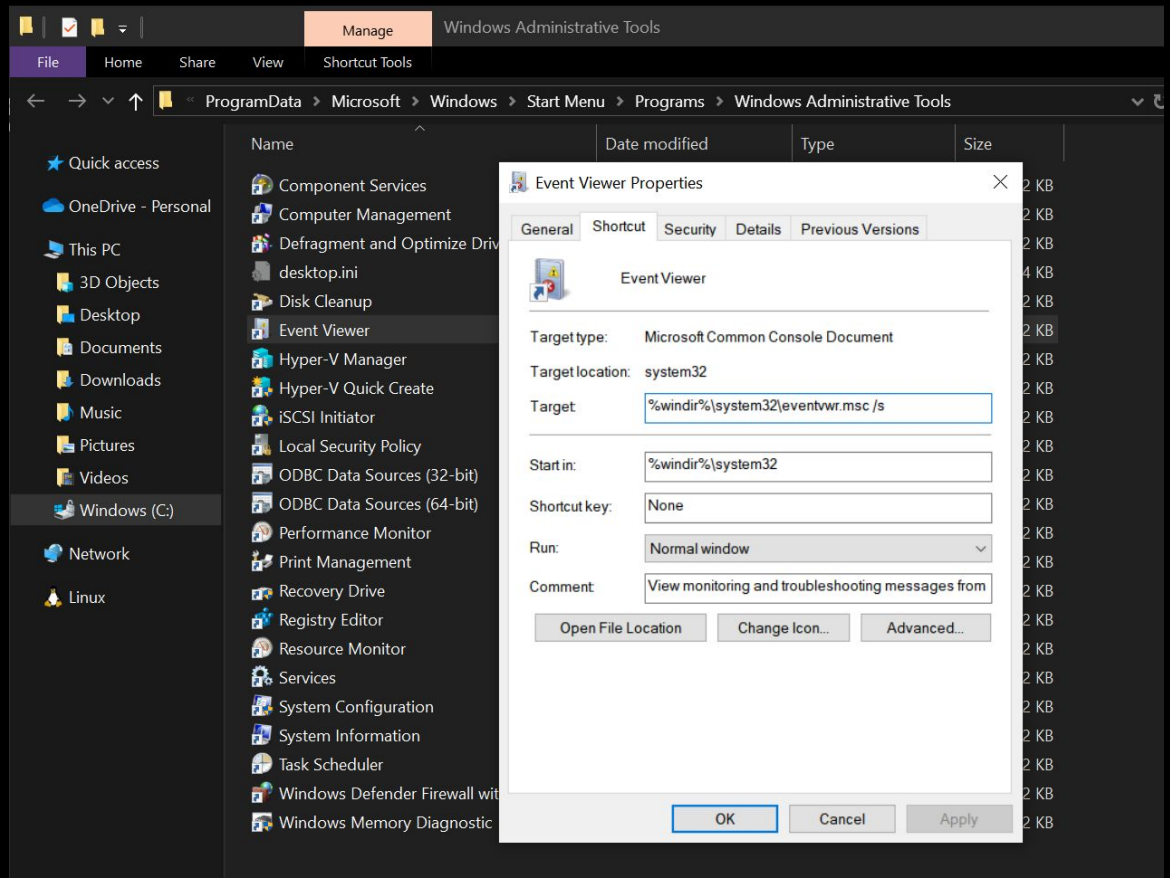
Event Viewer - Exploitation

- Classic MitM for NTLM relay

Event Viewer - Exploitation

- Classic MitM for NTLM relay
- Modify the Event Viewer shortcut for persistent pseudo coercion
- Modify the registry to load a modified els.dll





Event Viewer - Exploitation

- Classic MitM for NTLM relay
- Modify the Event Viewer shortcut for persistent pseudo coercion
- Modify the registry to load a modified

  els.dll

Requires administrative privileges

Event Viewer - Exploitation Benefits

- CVE patch uses new internal function - RegpCalculateConnectionFlags
  - Function checks new registry values for verdict

Event Viewer - Exploitation Benefits

- CVE patch uses new internal function - RegpCalculateConnectionFlags
  - Function checks new registry values for verdict

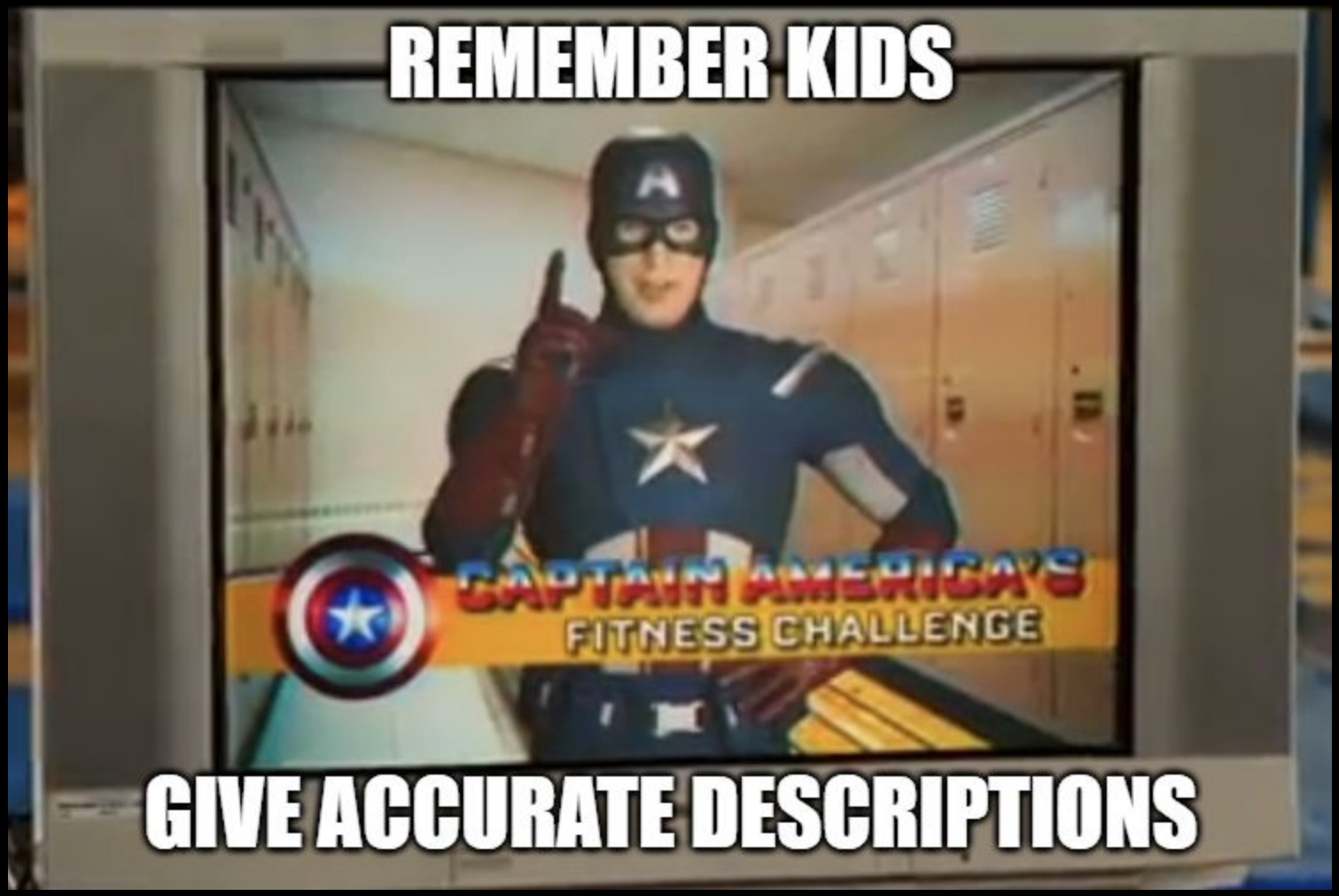
Software\Microsoft\RemoteRegistryClient\TransportFallbackPolicy

Event Viewer - Exploitation Benefits

- CVE patch uses new internal function - RegpCalculateConnectionFlags
  - Function checks new registry values for verdict

Software\Microsoft\RemoteRegistryClient\TransportFallbackPolicy

Attacker can set Event Viewer “backdoor” == Attacker can set transport policy


REMEMBER KIDS
CAPTAIN AMERICA'S
FITNESS CHALLENGE
GIVE ACCURATE DESCRIPTIONS

The Disclosure Process

Open vulnerability disclosure

Report on *RegConnectRegistryExW* with half working PoC — no actual relay

Reopen vulnerability disclosure

Report on *RegConnectRegistryW* with better PoC — adcs relay

Vulnerability patched

CVE-2024-43532 - EoP 8.8 CVSS

Report closed

Marked as documentation issue as *RegConnectRegistryExW* contains a flag to prevent fallback

Vulnerability confirmed

Further Research

- Look for more relay-able RPC clients
  - They seem more overlooked compared to RPC servers
- Coerce outbound Winreg on components that use RegConnectRegistryW

Thanks for listening

Questions?