MSRPC Socket Activation

SambaXP 2021

Volker Lendecke

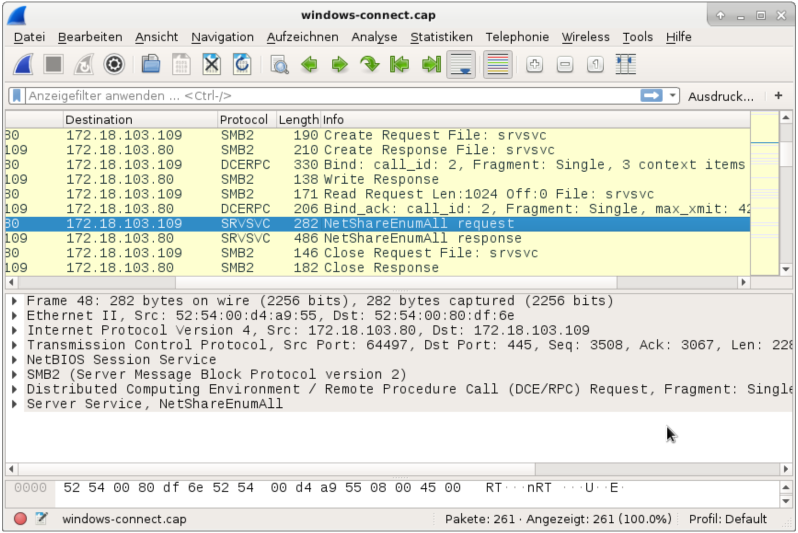
Samba Team / SerNet

May 2021

What is MSRPC?

- Microsoft-RPC, an extended version of DCE-RPC
- https://dcerpc.org:
  DCE/RPC is an implementation of the Remote Procedure Call technology developed by the Open Group as part of the Distributed Computing Environment. DCE/RPC is most commonly used to interact with Windows network services.
- A lot of Windows services until today depend on MSRPC:
  - Active Directory multi-master replication
  - Remote workstation management
  - Remote printing
  - Even listing shares
  - . . . and a lot more

Windows share listing



```
     Destination      Protocol  Length Info
30   172.18.103.109   SMB2         190 Create Request File: srvsvc
109  172.18.103.80    SMB2         210 Create Response File: srvsvc
30   172.18.103.109   DCERPC       330 Bind: call_id: 2, Fragment: Single, 3 context items
109  172.18.103.80    SMB2         138 Write Response
30   172.18.103.109   SMB2         171 Read Request Len:1024 Off:0 File: srvsvc
109  172.18.103.80    DCERPC       206 Bind_ack: call_id: 2, Fragment: Single, max_xmit: 42
30   172.18.103.109   SRVSVC       282 NetShareEnumAll request
109  172.18.103.80    SRVSVC       486 NetShareEnumAll response
30   172.18.103.109   SMB2         146 Close Request File: srvsvc
109  172.18.103.80    SMB2         182 Close Response
```

```
▸ Frame 48: 282 bytes on wire (2256 bits), 282 bytes captured (2256 bits)
▸ Ethernet II, Src: 52:54:00:d4:a9:55, Dst: 52:54:00:80:df:6e
▸ Internet Protocol Version 4, Src: 172.18.103.80, Dst: 172.18.103.109
▸ Transmission Control Protocol, Src Port: 64497, Dst Port: 445, Seq: 3508, Ack: 3067, Len: 228
▸ NetBIOS Session Service
▸ SMB2 (Server Message Block Protocol version 2)
▸ Distributed Computing Environment / Remote Procedure Call (DCE/RPC) Request, Fragment: Single
▸ Server Service, NetShareEnumAll
```

```
0000  52 54 00 80 df 6e 52 54  00 d4 a9 55 08 00 45 00   RT···nRT ···U··E·
```

```
windows-connect.cap          Pakete: 261 · Angezeigt: 261 (100.0%)   Profil: Default
```

Windows Interoperability

- SMB is just a tiny part of the full Windows client experience
  - Your artificial tests for open/read/write/close work fine
  - Then you try with Windows, and you can't see your shares
- File srvsvc in share IPC$??
  - Welcome to the wonderful world of Distributed Computing Environment Remote Procedure Calls
- Listing shares in the early days was easy
  - Anybody looked at `[MS-RAP]`?
  - OS/2 legacy protocol
  - Not sufficient for flexible RPCs
- Microsoft with Windows NT decided to use DCERPC

RPC protocol flow

- Open “srvsvc”
  - “srvsvc” is the server end of a named pipe, like a TCP socket
  - RPC also works over TCP, but MS-RPC predates the ubiquity of TCP, SMB worked over IPX and NetBEUI
  - RPC can even run directly on UDP, but that is less common
- SMB2 write into the srvsvc handle: RPC bind
  - Specify which daemon service to connect (LSA, SAMR, SRVSVC, etc)
  - Authenticate with the service, typically GSSAPI
  - Negotiate transport crypto (plaintext, sign, sign/seal)
- SMB2 IOCTL
  - RPC requests proper
  - NetShareEnumAll lists shares
- RPC over TCP just transmits the raw packets that are encapsulated in SMB2 read/write/ioctl

Samba RPC

- Started in the mid-1990s
- Luke Kenneth Casson Leighton's book
  - Windows NT domain interop
- Started with hand-marshalling packets
  - Still done today in fresh implementations
- In 2000, Tridge started PIDL for Samba
  - DCERPC IDL compiler with both omissions and extensions
  - Outputs readable C code
- Back then, the Windows IDLs were still secret
  - Big part of the EU vs MS case
  - Published as part of the Microsoft Protocols

Samba RPC implementation

- All RPC servers are linked into smbd
- Easy to implement, but not “the right thing”
- In May 2010, Simo Sorce started to split spoolssd
    - SMB is just a transport for RPC traffic
    - Goal: Separate printing into a daemon of its own
    - Infrastructure for other RPC services
    - RPC traffic passed through a unix domain socket
- This talk builds upon Simo’s work
- Spoolssd, lsasd and others fork from smbd
    - Separate processes, but still part of /usr/sbin/smbd

Demo

DEMO

New daemons

- samba_dcerpcd
  - "inetd" for Samba RPC daemons
  - Listens on behalf of RPC server implementations
- rpcd_epmapper
  - Implementation of DCERPC endpoint mapper
- rpcd_spoolss
  - Simo's spoolssd in a separate binary
- rpcd_winreg
  - You guessed it – the remote registry server
- rpcd_classic
  - Implement everything else (netlogon, samr, lsa, etc)
- No RPC server code in smbd anymore, just opening named pipes

samba_dcerpcd

- At startup, ask every rpcd about the interfaces it implements:

```
# ./rpcd_winreg --list-interfaces
338cd001-2244-31f1-aaaa-900038001003/0x00000001  winreg
 ncacn_np:[\pipe\winreg]
 ncacn_ip_tcp:
 ncalrpc:
```

- Listens on all sockets for the rpcd_* implementations
- From that specification, create and listen on sockets
- When a client connects, the corresponding rpcd implementation is forked/exec'ed and the socket is passed on via messaging
- samba_dcerpcd completely hands off handling of the connection
  - No DCERPC server implementation required in samba_dcerpcd

rpcd Implementation

- Two modes of operation
  - –list-interfaces just shows what services are provided
  - Without –list-interfaces listen on messages from samba_dcerpcd for sockets
- RPCD implementations don't create and listen on sockets
- Every process can handle multiple RPC connections
  - Based on earlier work in the RPC server space
- At client disconnect, report number of connections to samba_dcerpcd
- samba_dcerpc knows how many clients each process serves
  - Shutdown rpcd processes after a timeout (right now 10sec)

rpcd_epmapper

- samba_dcerpcd knows all interfaces and endpoints from –list-interfaces
- In current master, every source3 based RPC service registers explicitly using epm_Insert
- samba_dcerpcd fills a new tdb with all services:

```
key(48) = "338cd001-2244-31f1-aaaa-900038001003/0x00000001\00"
data(74) = "winreg ncacn_np:[\pipe\winreg] ncacn_ip_tcp:[49152]
            ncalrpc:[rpcd_winreg]"
```

- rpcd_epmapper queries and walks this tdb
  - epm_Insert/Delete right now not needed

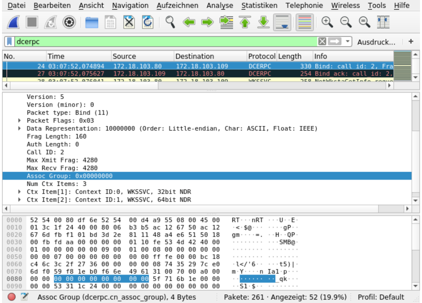
Association Groups

- Policy Handles (“pointers” to server-side objects) can be shared across multiple RPC connections
- An anology for Unix people

association group	unix process
network connection	unix thread
policy handle	file descriptor

- Not solved in source3 based servers
- “Solved” in source4 by putting all RPC servers that do policy handles into one process
- Clients can ask for association groups, the server assigns the ID

Association group ID assignment



```
No.  Time                Source           Destination      Protocol Length Info
  24 03:07:52,074894     172.18.103.80    172.18.103.109   DCERPC   330 Bind: call id: 2, Fra
  27 03:07:52,075627     172.18.103.109   172.18.103.80    DCERPC   254 Bind_ack: call_id: 2,
  28 03:07:52,076041     172.18.103.80    172.18.103.109   WKSSVC   258 NetWkstaGetInfo reque
```

```
      Version: 5
      Version (minor): 0
      Packet type: Bind (11)
    ▸ Packet Flags: 0x03
    ▸ Data Representation: 10000000 (Order: Little-endian, Char: ASCII, Float: IEEE)
      Frag Length: 160
      Auth Length: 0
      Call ID: 2
      Max Xmit Frag: 4280
      Max Recv Frag: 4280
      Assoc Group: 0x00000000
      Num Ctx Items: 3
    ▸ Ctx Item[1]: Context ID:0, WKSSVC, 32bit NDR
    ▸ Ctx Item[2]: Context ID:1, WKSSVC, 64bit NDR
```

```
0000  52 54 00 80 df 6e 52 54  00 d4 a9 55 08 00 45 00   RT···nRT ···U··E·
0010  01 3c 1f 24 40 00 80 06  b3 b5 ac 12 67 50 ac 12   ·<·$@··· ····gP··
0020  67 6d fb f1 01 bd 3d 2e  81 11 48 a4 e6 51 50 18   gm····=. ··H··QP·
0030  00 fb fd aa 00 00 00 00  01 10 fe 53 4d 42 40 00   ········ ···SMB@·
0040  01 00 00 00 00 00 09 00  01 00 08 00 00 00 00 00   ········ ········
0050  00 00 07 00 00 00 00 00  00 00 ff fe 00 00 bc 18   ········ ········
0060  c4 6c 3c 2f 27 36 00 00  00 00 08 74 35 29 7c e0   ·l</'6·· ···t5)|·
0070  6d f0 59 f8 1e b0 f6 6e  49 61 31 00 70 00 a0 00   m·Y····n Ia1·p···
0080  00 00 00 00 00 00 00 00  00 00 5f 71 6b 1e 00 00   ········ ·_qk···
0090  00 00 53 31 1c 24 00 00  00 00 00 00 00 00 00 00   ··S1·$·· ········
```

Assoc Group (dcerpc.cn_assoc_group), 4 Bytes | Pakete: 261 · Angezeigt: 52 (19.9%) | Profil: Default

C706 (dcerpc spec) 12.6.3.6: The client should set the assoc_group_id field either to 0 (zero), to indicate a new association group, or to the known value. When the server receives a value of 0, this indicates that the client has requested a new association group, and it assigns a server unique value to the group. This value is returned in the rpc_bind_ack PDU.

Association groups with samba_dcerpcd

- samba_dcerpcd accepts the socket and reads the bind packet
- How to pick one of the N winreg daemons?
- Metze's idea: 8 of the 32 assoc id bits are a process index
- The socket sent to rpcd_winreg also carries the bind packet

Thanks for you attention

`vl@samba.org / vl@sernet.de`

`http://www.sambaxp.org/`