DRSUAPI-Replication

Stefan Metzmacher
SerNet – Service Network GmbH
Samba Team
metze@samba.org

http://samba.org/~metze/presentations/2007/
metze_sambaxp2007_drsuapi_repl.pdf

samba eXperience 2007

Agenda

- Problem Space
- How to learn the replication protocol?
- Basic replication concepts
- How does the replication protocol works?
- Input parameters of DsGetNCChanges()
- Output parameters of DsGetNCChanges()
- Replication conflict handling
- Implementation status

Problem Space

- Samba4 should work as DC side by side together with Windows Servers.
- The only real way to do this is to talk the same protocol as Windows DC's.
- First we had to find out how the replication protocol works.
- With knowledge of the protocol we want to create a server implementation.

How to learn the DRSUAPI protocol

- DRSUAPI traffic is always forced to use DCERPC sealing.
- First challenge was to look into the DRSUAPI traffic using Wireshark with KRB5 decryption support, using a keytab file with extracted NT-Hashes (with our RPC-SAMSYNC torture test).
- We implemented the DsCrackNames() call in our client library to get a raw overview how the DRSUAPI interface works.

How to learn the DRSUAPI protocol (2)

- We step by step worked out the IDL for the DsGetNCChanges() call (which implements the actual replication) using our “ndrdump” tool.
- Then we wrote the “RPC-DSSYNC” test to find what is needed to get all data (including password fields) out of a Windows DC.
- In order to interpret the returned information completely we needed to find out about how the prefix mapping works (more about this later).

How to learn the DRSUAPI protocol (3)

- The DsGetNCChanges() responses can be compressed with MSZIP or the not yet discovered XPRESS algorithm. But it's possible to disable or select the algorithm with some flags.
- Samba4's NDR-layer handles the decompression transparent for the programer.
- To make real use of the returned password information we needed to find out the session specific encryption (more about this later).

Basic replication concepts

- Multimaster replication: Conflicts are resolved by an algorithm
- Pull replication: The changes are pulled from the Source DSA to the Destination DSA
- Change notify: Source DSA's notify Destination DSA's about changes
- Status based replication: No information history is maintained for Objects
- Store and forward replication: A change is only pulled once by the Destination DSA from any of its Source DSA's.

Basic replication concepts (2)

- Each DSA is identified by the objectGUID of its “NTDS Settings” object, the DSA-GUID, which never changes.
- Each AD-Database has also a GUID the invocationId of the “NTDS Settings” object, the DSA-InvocationId, which can change on some rare events like restore from backup.
- Each AD-Database has a USN, which is relative to the DSA-InvocationId, and it's incremented with each change.

Replication with DsGetNCChanges()

- Directory partitions are replicated on their own.
- That means the store and forward replication state information is per partition.
- Each DC does only pull cycle at a time (for all partitions together).
- The smallest replication unit is an attribute, but for linked attributes it's a single attribute value.

Input of DsGetNCChanges()

- A bind handle from DsBind()
- Destination-DSA-GUID
- Source-DSA-InvocationId
- DN of the directory partition
- Highwater-Mark relative to Source-DSA-InvocationId
- The Destination-DSA's Up-To-Date-Vector
- A list of flags to turn some features on or off

LDAP-DN within DRSUAPI

```
naming_context: struct drsuapi_DsReplicaObjectIdentifier
    __ndr_size          : 0x00000078 (120)
    __ndr_size_sid      : 0x00000000 (0)
    guid                : 00000000-0000-0000-0000-000000000000
    sid                 : S-0-0
    dn                  : 'DC=w2k3,DC=vmnet1,DC=vm,DC=base'

naming_context: struct drsuapi_DsReplicaObjectIdentifier
    __ndr_size          : 0x00000078 (120)
    __ndr_size_sid      : 0x00000018 (24)
    guid                : 99efd389-502f-4d3f-aefb-e7a1d9acfb2f
    sid                 : S-1-5-21-769185814-1958994947-1641909093
    dn                  : 'DC=w2k3,DC=vmnet1,DC=vm,DC=base'
```

Highwater-Mark

old_highwatermark: struct drsuapi_DsReplicaHighWaterMark
    tmp_highest_usn : 0x000000000002d1ab (184747)
    reserved_usn   : 0x0000000000000000 (0)
    highest_usn   : 0x0000000000000000 (0)
new_highwatermark: struct drsuapi_DsReplicaHighWaterMark
    tmp_highest_usn : 0x00000000003113b (201019)
    reserved_usn   : 0x0000000000000000 (0)
    highest_usn   : 0x00000000003113b (201019)

Up-To-Date-Vector

```
uptodateness_vector: struct drsuapi_DsReplicaCursorCtr
  count                    : 0x00000002 (2)
  reserved                 : 0x00000000 (0)
  cursors: ARRAY(2)
   cursors: struct drsuapi_DsReplicaCursor
     source_dsa_invocation_id : 32194565-b3c0-4b7f-8a85-88cfb8318042
     highest_usn              : 0x00000000000030be (12478)
   cursors: struct drsuapi_DsReplicaCursor
     source_dsa_invocation_id : b173daa5-c401-443b-9028-e2f832fe466f
     highest_usn              : 0x0000000000031147 (201031)
```

replica_flags

replica_flags : 0x00200870 (2099312)
1: DRSUAPI_DS_REPLICA_NEIGHBOUR_WRITEABLE
1: DRSUAPI_DS_REPLICA_NEIGHBOUR_SYNC_ON_STARTUP
1: DRSUAPI_DS_REPLICA_NEIGHBOUR_DO_SCHEDULED_SYNCS
0: DRSUAPI_DS_REPLICA_NEIGHBOUR_USE_ASYNC_INTERSIDE_TRANSPORT
0: DRSUAPI_DS_REPLICA_NEIGHBOUR_TWO_WAY_SYNC
1: DRSUAPI_DS_REPLICA_NEIGHBOUR_RETURN_OBJECT_PARENTS
0: DRSUAPI_DS_REPLICA_NEIGHBOUR_FULL_IN_PROGRESS
0: DRSUAPI_DS_REPLICA_NEIGHBOUR_FULL_NEXT_PACKET
1: DRSUAPI_DS_REPLICA_NEIGHBOUR_NEVER_SYNCED
0: DRSUAPI_DS_REPLICA_NEIGHBOUR_PREEMPTED
0: DRSUAPI_DS_REPLICA_NEIGHBOUR_IGNORE_CHANGE_NOTIFICATIONS
0: DRSUAPI_DS_REPLICA_NEIGHBOUR_DISABLE_SCHEDULED_SYNC
0: DRSUAPI_DS_REPLICA_NEIGHBOUR_COMPRESS_CHANGES
0: DRSUAPI_DS_REPLICA_NEIGHBOUR_NO_CHANGE_NOTIFICATIONS
0: DRSUAPI_DS_REPLICA_NEIGHBOUR_PARTIAL_ATTRIBUTE_SET

Output of DsGetNCChanges()

- Source-DSA-GUID
- Source-DSA-InvocationId
- The Destination-DSA's Highwater-Mark
- The Source-DSA's Highwater-Mark
- The Source-DSA's Up-To-Date-Vector
- The Source-DSA's Prefix-Mappings-Table
- The list of replicated objects
- The list of replicated linked attribute values

OID-Prefix-Mapping

```
mapping_ctr: struct drsuapi_DsReplicaOIDMapping_Ctr
        num_mappings        : 0x00000021 (33)
        mappings            : *
        mappings: ARRAY(33)
          mappings: struct drsuapi_DsReplicaOIDMapping
            id_prefix           : 0x00000000 (0)
            oid: struct drsuapi_DsReplicaOID
                __ndr_size      : 0x00000002 (2)
                oid             : *
                    oid             : 2.5.4'
...
```

OID-Prefix-Mapping (2)

OID-prefix	=> UINT32-Id prefix
2.5.4.*	⇒ 0x00000000
2.5.6.*	⇒ 0x00010000
1.2.840.113556.1.2.*	⇒ 0x00020000
1.2.840.113556.1.3.*	⇒ 0x00030000
2.5.5.*	⇒ 0x00080000
1.2.840.113556.1.4.*	⇒ 0x00090000
1.2.840.113556.1.5.*	⇒ 0x000A0000
2.16.840.1.113730.3.*	⇒ 0x00140000
0.9.2342.19200300.100.1.*	⇒ 0x00150000
2.16.840.1.113730.3.1.*	⇒ 0x00160000
1.2.840.113556.1.5.7000.*	⇒ 0x00170000
2.5.21.*	⇒ 0x00180000

...

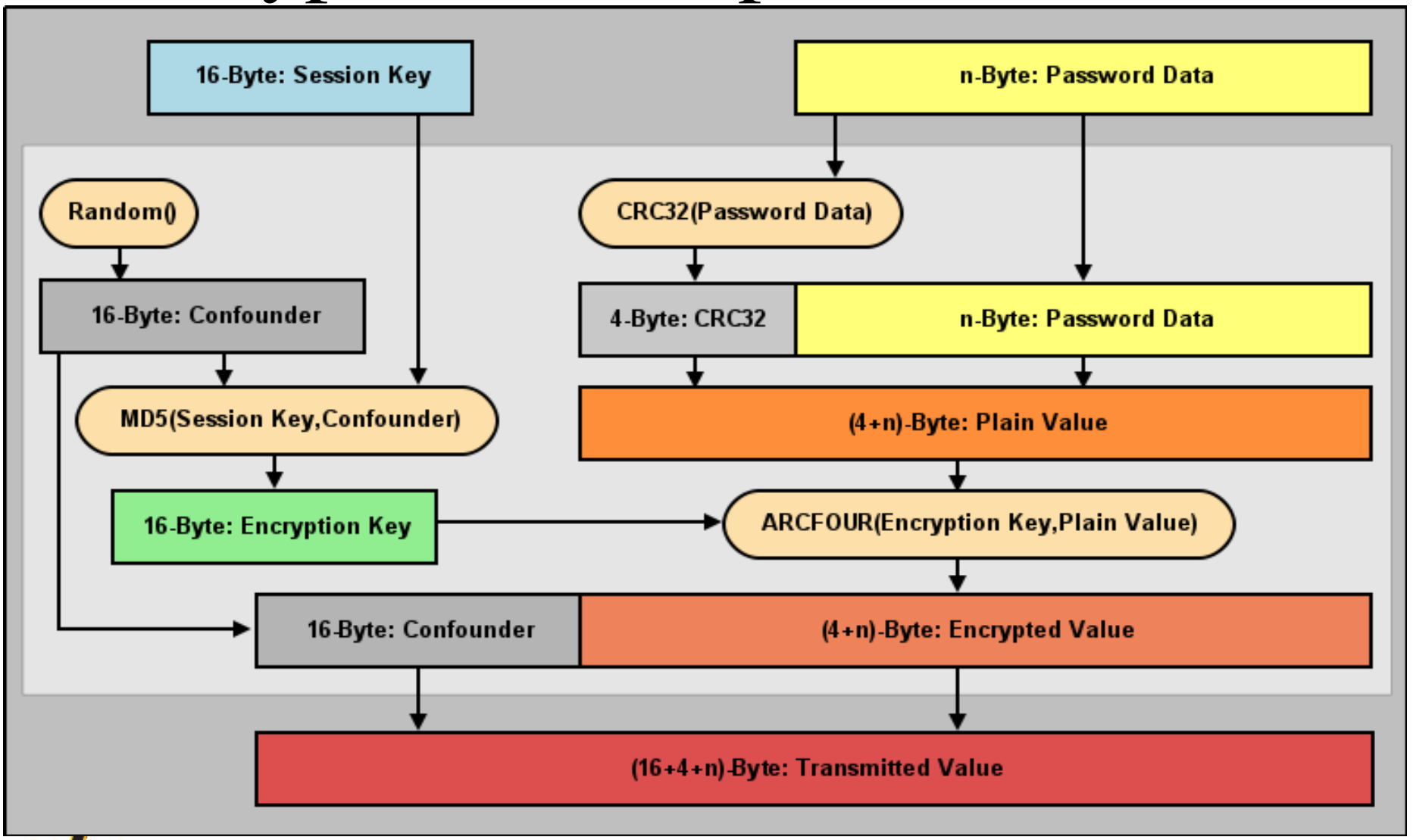
OID-Prefix-Mapping (3)

- In DRSUAPI all attributes with syntax 2.5.5.2 are identified by uint32 values, the following example shows the mapping used between the two representations e.g. - objectClass 'nTDSDSA' has governsID 1.2.840.113556.1.5.7000.47 and a UINT32-ID of 0x0017002F
- The OID 1.2.840.113556.1.5.7000.47 is splitted into a OID-prefix: 1.2.840.113556.1.5.7000 and a value: 47 => 0x2F
- The mapping table gives a UINT32-prefix: 0x00170000 and the UINT32-ID is 0x0017002F = 0x00170000 | 0x2F
- This prefix mapping table is replied in the drsuapi_DsReplicaOIDMapping_Ctr array.

Interpret replicated objects

- Create the attribute names using the Prefix-Mapping from the 32-Bit-ID.
- Convert the attribute values depending on the attribut-syntax.
- Decrypt the password fields.
- Create Backlinks of Linked-Attributes.

Decryption of the password fields



The replication meta-data array

```
meta_data_ctr: struct drsuapi_DsReplicaMetaDataCtr
  count                    : 0x00000024 (36)
  meta_data: ARRAY(36)
     meta_data: struct drsuapi_DsReplicaMetaData
        version                          : 0x00000001 (1)
        originating_change_time  : Tue Jul 13 11:10:25 2004 CEST
        originating_invocation_id: b173daa5-c401-443b-9028-e2f832fe466f
        originating_usn                : 0x0000000000001002 (4098)
     meta_data: struct drsuapi_DsReplicaMetaData
        version                          : 0x0000002c (44)
        originating_change_time  : Mon Feb 19 16:57:23 2007 CET
        originating_invocation_id: b173daa5-c401-443b-9028-e2f832fe466f
        originating_usn                : 0x000000000003035b (197467)
...
```

Current Implementation in Samba4

- We have the “NET-API-BECOME-DC” test, which pull all data from a Windows-Server and provisions the samba 4 server as DC.
- “smbd” has a “dreplsrv” service which pulls changes frequently from the Windows-Server.
- But there's a lot of work to do:
  - http://wiki.samba.org/index.php/Samba4/ActiveDirectory
- Demo...

Current Implementation in Samba4

- Fetch the samba4 source code from svn
  - See http://devel.samba.org/
- And take a look at the related source code
  - source/dsdb/repl/
  - source/libnet/libnet_become_dc.*
  - source/torture/libnet/libnet_BecomeDC.c
- I'll upload my thesis within the next weeks
  - See http://samba.org/~metze/presentations/2007/thesis/
  - It's in german only, sorry!

More Info...

- Are there any questions?
- Many thanks for your attention!