SDC

STORAGE DEVELOPER CONFERENCE

SNIA SANTA CLARA, 2017

Windows Authentication With Multiple Domains and Forests

Stefan Metzmacher <metze@samba.org>

Samba Team / SerNet

2017-09-13

Check for updates: https://samba.org/~metze/presentations/2017/SDC/

Update from SambaXP 2017

- This is an update to my talk at SambaXP.
- "The Important Details Of Windows Authentication"
- Please have a look at the slides:
- https://samba.org/~metze/presentations/2017/SambaXP/
- An audio recording is also available here:
- https://sambaxp.org/archive_data/SambaXP2017-AUDIO/Day3/Track2/

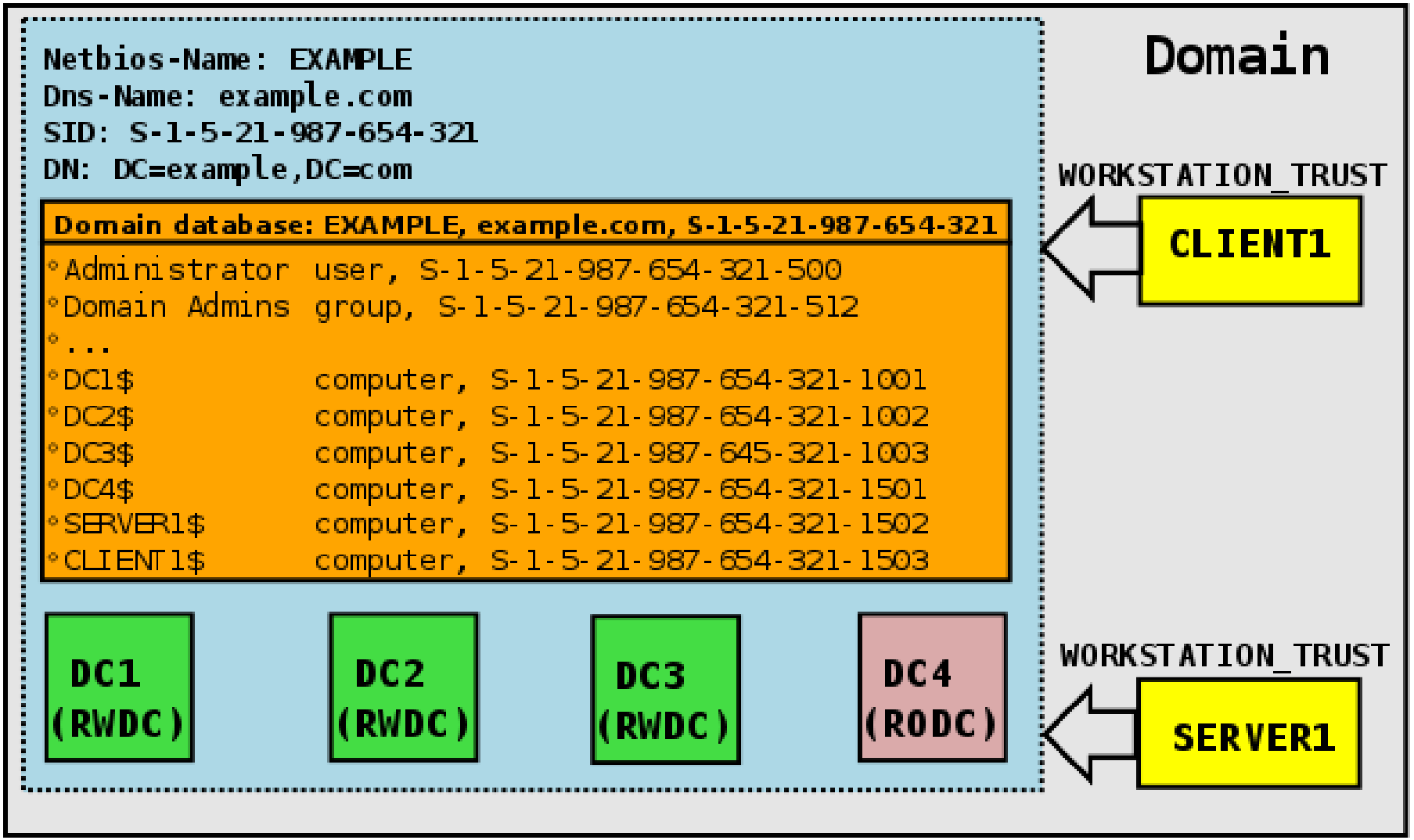
- Check for an updated version of this slides here:
- https://samba.org/~metze/presentations/2017/SDC/

(draft)

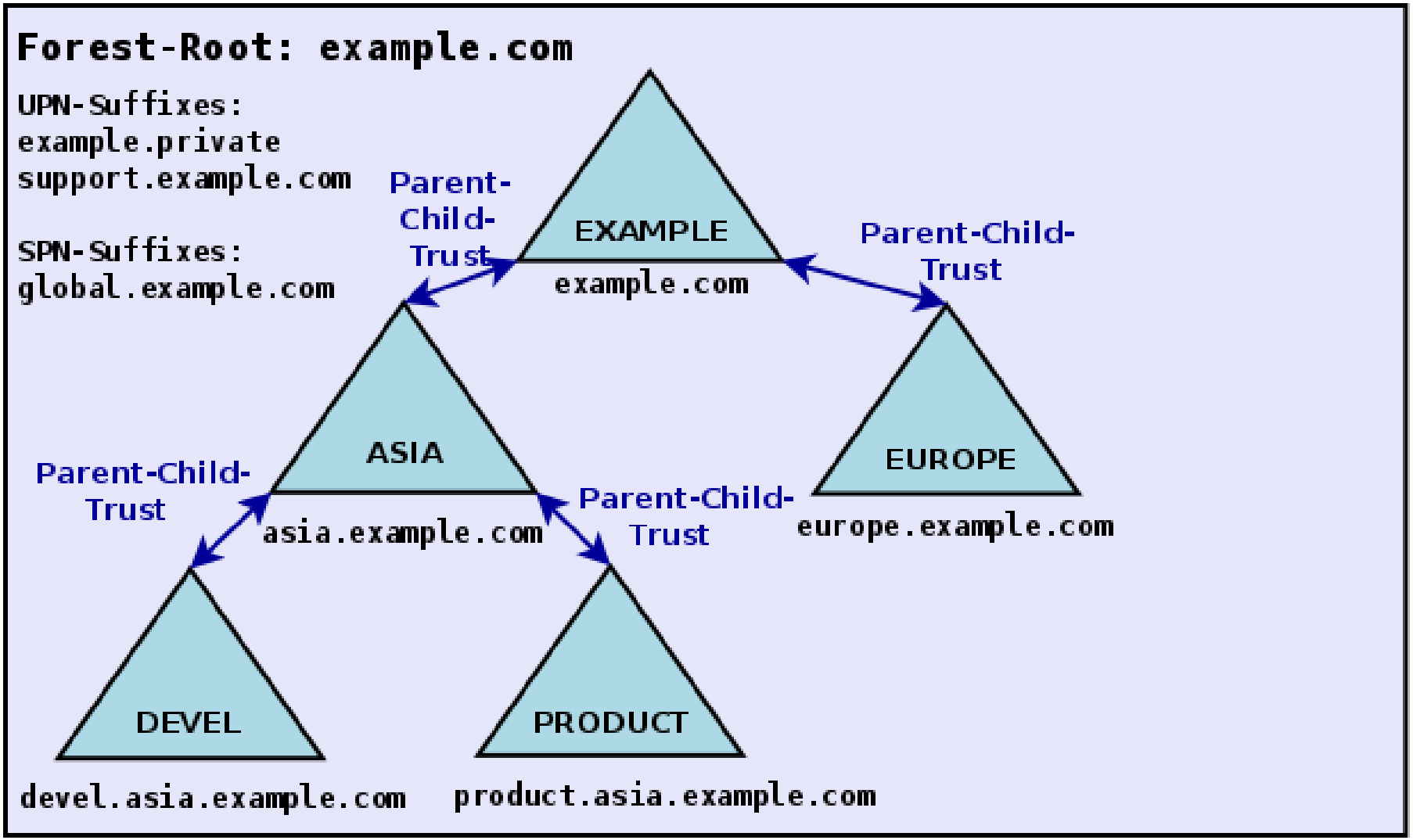
Topics

- Windows Domains, Forests and Trusts
- Netlogon Secure Channel
- Authentication Protocols
- Authorization Token
- Trust Routing Table
- New Kerberos Features
- Thanks!
- Questions?

Layout of a single Windows Domain



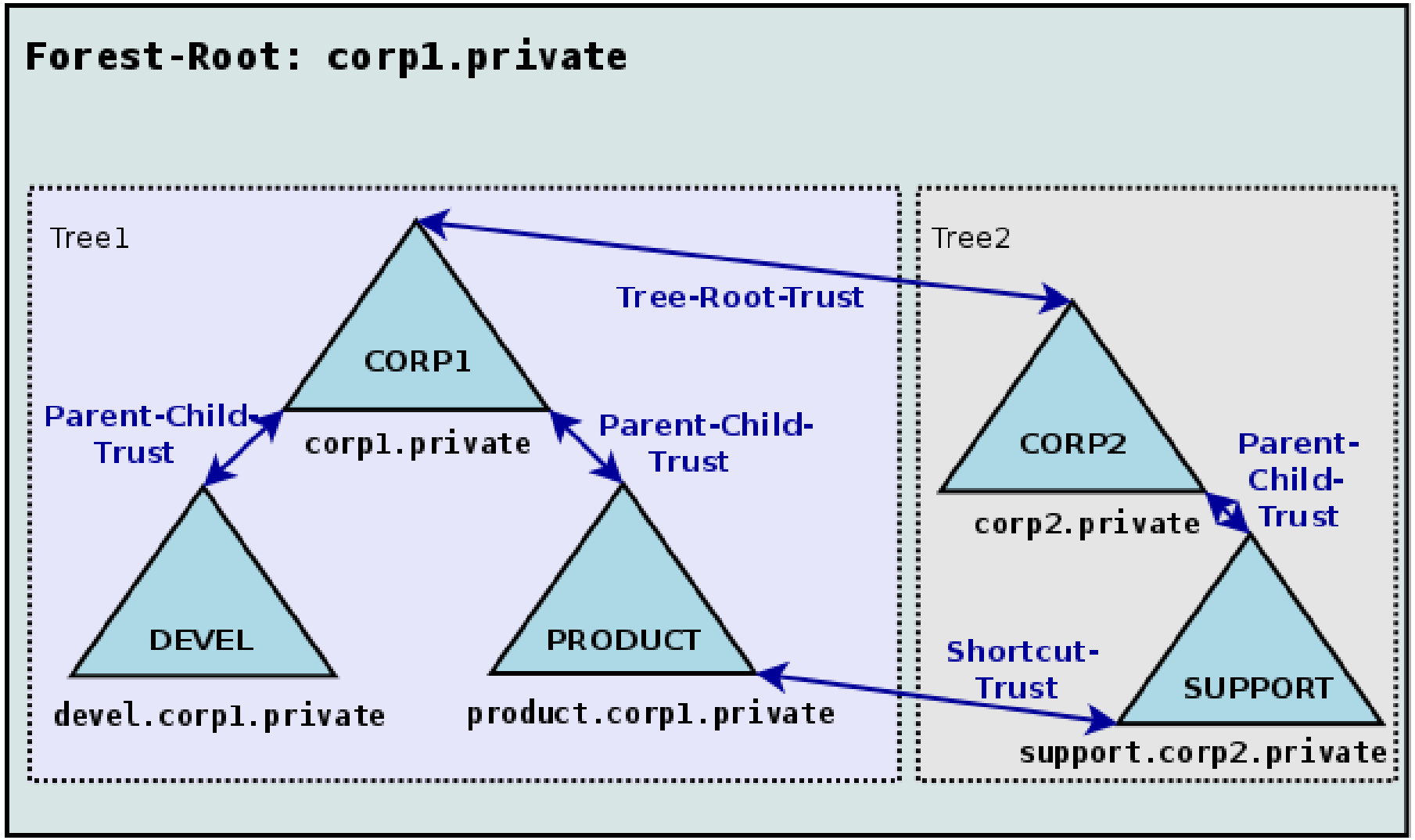
Layout of an Active Directory Forest (with one Tree)



Forest Information (with one Tree)

- TOP_LEVEL_NAME: example.com
- TOP_LEVEL_NAME: example.private
- DOMAIN_INFO: EXAMPLE; example.com; S-1-5-21-99-88-11
- DOMAIN_INFO: ASIA; asia.example.com; S-1-5-21-99-88-22
- DOMAIN_INFO: DEVEL; devel.asia.example.com; S-1-5-21-99-88-33
- DOMAIN_INFO: PRODUCT; product.asia.example.com; S-1-5-21-99-88-44
- DOMAIN_INFO: EUROPE; europe.example.com; S-1-5-21-99-88-44

Layout of an Active Directory Forest (with multiple Trees)



Forest Information (with multiple Tree)

- TOP_LEVEL_NAME: corp1.private
- TOP_LEVEL_NAME: corp2.private
- DOMAIN_INFO: CORP1; corp1.private; S-1-5-21-77-88-11
- DOMAIN_INFO: DEVEL; devel.corp1.private; S-1-5-21-77-88-22
- DOMAIN_INFO: PRODUCT; product.corp1.private; S-1-5-21-99-88-33
- DOMAIN_INFO: CORP2; corp2.private; S-1-5-21-99-88-44
- DOMAIN_INFO: SUPPORT; support.corp2.private; S-1-5-21-99-88-55

Trust Types (low level)

- LSA_TRUST_TYPE_DOWNLEVEL
  - This is used for NT4 Domains.
  - It can only handle NTLMSSP.
- LSA_TRUST_TYPE_UPLEVEL
  - This is used for AD Domains.
  - It supports NTLMSSP by default.
  - It supports Kerberos, the Realm is the Dns-Domain-Name.
- LSA_TRUST_TYPE_MIT
  - This is used for trusts to RFC4120-compliant Kerberos.
  - Unlikely to be implemented in Samba.
- LSA_TRUST_TYPE_DCE
  - Not used in Windows.

Trust Types (low level)

- LSA_TRUST_TYPE_DOWNLEVEL
  - This is used for NT4 Domains.
  - It can only handle NTLMSSP.
- LSA_TRUST_TYPE_UPLEVEL
  - This is used for AD Domains.
  - It supports NTLMSSP by default.
  - It supports Kerberos, the Realm is the Dns-Domain-Name.
- LSA_TRUST_TYPE_MIT
  - This is used for trusts to RFC4120-compliant Kerberos.
  - Unlikely to be implemented in Samba.
- LSA_TRUST_TYPE_DCE
  - Not used in Windows.

Trust Types (low level)

- LSA_TRUST_TYPE_DOWNLEVEL
  - This is used for NT4 Domains.
  - It can only handle NTLMSSP.
- LSA_TRUST_TYPE_UPLEVEL
  - This is used for AD Domains.
  - It supports NTLMSSP by default.
  - It supports Kerberos, the Realm is the Dns-Domain-Name.
- LSA_TRUST_TYPE_MIT
  - This is used for trusts to RFC4120-compliant Kerberos.
  - Unlikely to be implemented in Samba.
- LSA_TRUST_TYPE_DCE
  - Not used in Windows.

Trust Types (low level)

- LSA_TRUST_TYPE_DOWNLEVEL
  - This is used for NT4 Domains.
  - It can only handle NTLMSSP.
- LSA_TRUST_TYPE_UPLEVEL
  - This is used for AD Domains.
  - It supports NTLMSSP by default.
  - It supports Kerberos, the Realm is the Dns-Domain-Name.
- LSA_TRUST_TYPE_MIT
  - This is used for trusts to RFC4120-compliant Kerberos.
  - Unlikely to be implemented in Samba.
- LSA_TRUST_TYPE_DCE
  - Not used in Windows.

Trust Directions

- Trusting vs. Trusted Domain
  - Users of the "trusted" domain can access resources of the "trusting" domain.
- LSA_TRUST_DIRECTION_INBOUND
  - The local domain is the "trusted" domain.
  - The specified/remote domain is the "trusting" domain.
  - Also known as INCOMING.
- LSA_TRUST_DIRECTION_OUTBOUND
  - The local domain is the "trusting" domain.
  - The specified/remote domain is the "trusted" domain.
  - Also known as OUTGOING.

Trust Directions

- Trusting vs. Trusted Domain
  - Users of the "trusted" domain can access resources of the "trusting" domain.
- LSA_TRUST_DIRECTION_INBOUND
  - The local domain is the "trusted" domain.
  - The specified/remote domain is the "trusting" domain.
  - Also known as INCOMING.
- LSA_TRUST_DIRECTION_OUTBOUND
  - The local domain is the "trusting" domain.
  - The specified/remote domain is the "trusted" domain.
  - Also known as OUTGOING.

Trust Directions

- Trusting vs. Trusted Domain
  - Users of the "trusted" domain can access resources of the "trusting" domain.
- LSA_TRUST_DIRECTION_INBOUND
  - The local domain is the "trusted" domain.
  - The specified/remote domain is the "trusting" domain.
  - Also known as INCOMING.
- LSA_TRUST_DIRECTION_OUTBOUND
  - The local domain is the "trusting" domain.
  - The specified/remote domain is the "trusted" domain.
  - Also known as OUTGOING.

Transitive vs. Non-Transitive Trusts

- Non-Transitive Trust
  - This is just a trust between two single domains.
- Transitive Trust
  - The trust between two single domains is expanded to indirect trusts.
  - DOM1 trusts DOM2, while DOM2 trusts DOM3, so DOM1 implicitly trusts DOM3.
  - In some situations a transitive trust is some kind of default route.

Transitive vs. Non-Transitive Trusts

- Non-Transitive Trust
  - This is just a trust between two single domains.
- Transitive Trust
  - The trust between two single domains is expanded to indirect trusts.
  - DOM1 trusts DOM2, while DOM2 trusts DOM3, so DOM1 implicitly trusts DOM3.
  - In some situations a transitive trust is some kind of default route.

Trust Types (high level, Part 1)

- Workstation (Domain Member) Trust
  - LSA_TRUST_DIRECTION_OUTBOUND to the primary domain.
  - LSA_TRUST_TYPE_DOWNLEVEL (NT4) or LSA_TRUST_TYPE_UPLEVEL (AD).
  - Transitive Trust as default route.
  - computer account can only reliable access its primary domain.
- External Domain Trust
  - LSA_TRUST_TYPE_DOWNLEVEL (NT4) or LSA_TRUST_TYPE_UPLEVEL (AD).
  - Non-Transitive
- Forest Trust
  - LSA_TRUST_TYPE_UPLEVEL (AD) between two forest root domains.
  - Transitive Trust (by default) between the two forests only.

Trust Types (high level, Part 1)

- Workstation (Domain Member) Trust
  - LSA_TRUST_DIRECTION_OUTBOUND to the primary domain.
  - LSA_TRUST_TYPE_DOWNLEVEL (NT4) or LSA_TRUST_TYPE_UPLEVEL (AD).
  - Transitive Trust as default route.
  - computer account can only reliable access its primary domain.
- External Domain Trust
  - LSA_TRUST_TYPE_DOWNLEVEL (NT4) or LSA_TRUST_TYPE_UPLEVEL (AD).
  - Non-Transitive
- Forest Trust
  - LSA_TRUST_TYPE_UPLEVEL (AD) between two forest root domains.
  - Transitive Trust (by default) between the two forests only.

Trust Types (high level, Part 1)

- Workstation (Domain Member) Trust
  - LSA_TRUST_DIRECTION_OUTBOUND to the primary domain.
  - LSA_TRUST_TYPE_DOWNLEVEL (NT4) or LSA_TRUST_TYPE_UPLEVEL (AD).
  - Transitive Trust as default route.
  - computer account can only reliable access its primary domain.
- External Domain Trust
  - LSA_TRUST_TYPE_DOWNLEVEL (NT4) or LSA_TRUST_TYPE_UPLEVEL (AD).
  - Non-Transitive
- Forest Trust
  - LSA_TRUST_TYPE_UPLEVEL (AD) between two forest root domains.
  - Transitive Trust (by default) between the two forests only.

Trust Types (high level, Part 2 within Forests)

- Parent Child Trusts
  - LSA_TRUST_DIRECTION_INBOUND and LSA_TRUST_DIRECTION_OUTBOUND
  - LSA_TRUST_TYPE_UPLEVEL (AD).
  - LSA_TRUST_ATTRIBUTE_WITHIN_FOREST.
  - The child is a DNS-subdomain of the parent
  - Transitive Trust, on the parent with a route to the child and the related grandchildren.
  - Transitive Trust, on the child as default route.
  - Automatically created together with the child domain.
- Tree Root Trusts
  - Similar to Parent Child Trust.
  - The new tree root is not DNS-domain below the forest root.
  - Transitive Trust, on the forest root with a route to the new tree root and the related grandchildren.
  - Transitive Trust, on the child as default route.
  - Automatically created together with the new tree root domain.

Trust Types (high level, Part 2 within Forests)

- Parent Child Trusts
  - LSA_TRUST_DIRECTION_INBOUND and LSA_TRUST_DIRECTION_OUTBOUND
  - LSA_TRUST_TYPE_UPLEVEL (AD).
  - LSA_TRUST_ATTRIBUTE_WITHIN_FOREST.
  - The child is a DNS-subdomain of the parent
  - Transitive Trust, on the parent with a route to the child and the related grandchildren.
  - Transitive Trust, on the child as default route.
  - Automatically created together with the child domain.
- Tree Root Trusts
  - Similar to Parent Child Trust.
  - The new tree root is not DNS-domain below the forest root.
  - Transitive Trust, on the forest root with a route to the new tree root and the related grandchildren.
  - Transitive Trust, on the child as default route.
  - Automatically created together with the new tree root domain.

Trust Types (high level, Part 3 within Forests)

- Shortcut Trust
  - LSA_TRUST_DIRECTION_INBOUND and/or LSA_TRUST_DIRECTION_OUTBOUND
  - LSA_TRUST_TYPE_UPLEVEL (AD).
  - LSA_TRUST_ATTRIBUTE_WITHIN_FOREST.
  - Non-Transitive, acts as direct route to the specified domain.
  - Created by an administrator for performance reasons.

Trust Attributes (low level)

The content of the trustAttributes attribute in Samba:

```
typedef [public,bitmap32bit] bitmap {
    LSA_TRUST_ATTRIBUTE_NON_TRANSITIVE      = 0x00000001,
    LSA_TRUST_ATTRIBUTE_UPLEVEL_ONLY        = 0x00000002,
    LSA_TRUST_ATTRIBUTE_QUARANTINED_DOMAIN  = 0x00000004,
    LSA_TRUST_ATTRIBUTE_FOREST_TRANSITIVE   = 0x00000008,
    LSA_TRUST_ATTRIBUTE_CROSS_ORGANIZATION  = 0x00000010,
    LSA_TRUST_ATTRIBUTE_WITHIN_FOREST       = 0x00000020,
    LSA_TRUST_ATTRIBUTE_TREAT_AS_EXTERNAL   = 0x00000040,
    LSA_TRUST_ATTRIBUTE_USES_RC4_ENCRYPTION = 0x00000080
    // TODO LSA_TRUST_ATTRIBUTE_CROSS_ORGANIZATION_NO_TGT_DELEGATION = 0x00000200
    // TODO LSA_TRUST_ATTRIBUTE_PIM_TRUST           = 0x00000400
} lsa_TrustAttributes;
```

Forest (routing) Information

- The information about a forest:
  - can be queried from the forest root of the "trusted" forest by netr_GetForestTrustInformation() constructed by the information under CN=Partitions,CN=Configuration,...
  - is stored in the "msDS-TrustForestTrustInfo" attribute in the root domain of the "trusting" forest.
- It is an array of records of the following types:
  - FOREST_TRUST_DOMAIN_INFO includes Netbios-Name, DNS-Name and Domain-Sid.
  - FOREST_TRUST_TOP_LEVEL_NAME includes a top level DNS-Name that part of the forest (including all DNS-subdomains).
  - FOREST_TRUST_TOP_LEVEL_NAME_EX includes a top level DNS-Name that is explicitly excluded from the forest (including all DNS-subdomains).
  - Individual records will be disabled if conflicts with other trusts are detected.
  - Individual records can also be disabled by the admin.

Forest (routing) Information

- The information about a forest:
  - can be queried from the forest root of the "trusted" forest by netr_GetForestTrustInformation() constructed by the information under CN=Partitions,CN=Configuration,...
  - is stored in the "msDS-TrustForestTrustInfo" attribute in the root domain of the "trusting" forest.

- It is an array of records of the following types:
  - FOREST_TRUST_DOMAIN_INFO includes Netbios-Name, DNS-Name and Domain-Sid.
  - FOREST_TRUST_TOP_LEVEL_NAME includes a top level DNS-Name that part of the forest (including all DNS-subdomains).
  - FOREST_TRUST_TOP_LEVEL_NAME_EX includes a top level DNS-Name that is explicitly excluded from the forest (including all DNS-subdomains).
  - Individual records will be disabled if conflicts with other trusts are detected.
  - Individual records can also be disabled by the admin.

Netlogon Secure Schannel (Part1)

- Having an LSA_TRUST_DIRECTION_OUTBOUND Trust:
  - Means the "trusting" workstation/domain can establish a Netlogon Secure Channel to DCs of the "trusted" domain using the computer/trust account.
  - The NETLOGON protocol is bases on DCERPC, see [MS-NRPC].
- Establishing a global session state with a "trusted" DC:
  - netr_ServerReqChallenge() and netr_ServerAuthenticate[2,3]() are used to do a challenge/response authentication
  - The global session state is indexed by the computer name of the "client".
  - The global session state contains the initial session key, a sequence number.
  - Samba uses 'struct netlogon_creds_CredentialState' for this state.
  - This state is stored in netlogon_creds_cli.tdb (on the client) and schannel_store.tdb (on the server).

Netlogon Secure Schannel (Part1)

- Having an LSA_TRUST_DIRECTION_OUTBOUND Trust:
  - Means the "trusting" workstation/domain can establish a Netlogon Secure Channel to DCs of the "trusted" domain using the computer/trust account.
  - The NETLOGON protocol is bases on DCERPC, see [MS-NRPC].
- Establishing a global session state with a "trusted" DC:
  - netr_ServerReqChallenge() and netr_ServerAuthenticate[2,3]() are used to do a challenge/response authentication
  - The global session state is indexed by the computer name of the "client".
  - The global session state contains the initial session key, a sequence number.
  - Samba uses 'struct netlogon_creds_CredentialState' for this state.
  - This state is stored in netlogon_creds_cli.tdb (on the client) and schannel_store.tdb (on the server).

Netlogon Secure Schannel (Part2)

- A lot of functions operate on the global session state:
  - netr_LogonSamLogon[WithFlags](), netr_ServerPasswordSet[2](), netr_LogonGetDomainInfo(), netr_GetForestTrustInformation() and others.
  - All functions using 'netr_Authenticator' arguments.
  - These functions do some rolling crypto on the global session state.
  - These functions need to be strictly ordered (globally!)
  - Some of them also encrypt some application level fields with the current global session key.

Netlogon Secure Schannel (Part3)

- The NETLOGON protocol implements a custom DCERPC authentication type (auth_type=68):
  - The DCERPC Bind/AlterContext just passes the domain and computer names to the server.
  - The server takes a copy of the current global session based on the provided computer name.
  - This copy will be the session key for the lifetime of the DCERPC auth context.
  - Client and server provide DCERPC_AUTH_LEVEL_INTEGRITY or DCERPC_AUTH_LEVEL_PRIVACY protection for the auth context.
  - The connection doesn't support concurrent multiplexing and only one request at a time.

Netlogon Secure Schannel (Part4)

- Usage of DCERPC authentication type (auth_type=68)
  - It is typically used for a protected NETLOGON connection.
  - It is also used for LSA connections and the lsa_LookupNames4() and lsa_LookupSids3() calls.
  - Typically the "trusting" side of the trust should only use these NETLOGON and LSA connections to communicate with the "trusted" domain.

Netlogon Secure Schannel (Part5)

- Authentication verification uses NETLOGON:
  - netr_LogonSamLogon[WithFlags,Ex]() is typically used to verify NTLMSSP authentication.
  - But it's not limited to NTLMSSP, e.g. Kerberos PAC-Validation.
- Authentication should scale:
  - netr_LogonSamLogonEx() is an optimization of netr_LogonSamLogon[WithFlags]().
  - It isn't bound to the netr_Authenticator global ordering.
  - It avoids application level encryption with the current global session key in most cases if DCERPC_AUTH_LEVEL_PRIVACY is in use.
  - It can use multiple DCERPC connections to scale.

Netlogon Secure Schannel (Part5)

- Authentication verification uses NETLOGON:
  - netr_LogonSamLogon[WithFlags,Ex]() is typically used to verify NTLMSSP authentication.
  - But it's not limited to NTLMSSP, e.g. Kerberos PAC-Validation.
- Authentication should scale:
  - netr_LogonSamLogonEx() is an optimization of netr_LogonSamLogon[WithFlags]().
  - It isn't bound to the netr_Authenticator global ordering.
  - It avoids application level encryption with the current global session key in most cases if DCERPC_AUTH_LEVEL_PRIVACY is in use.
  - It can use multiple DCERPC connections to scale.

SPNEGO Authentication example

- All application protocols used in active directory domains use SPNEGO (RFC 4178, [MS-SPNG]) in order to negotiate between NTLMSSP ([MS-NLMP]) or Kerberos (RFC 4120, [MS-KILE])

```
▼ SMB2 (Server Message Block Protocol version 2)
  ▶ SMB2 Header
  ▼ Session Setup Request (0x01)
    ▶ StructureSize: 0x0019
    ▶ Flags: 0
    ▶ Security mode: 0x02, Signing required
    ▶ Capabilities: 0x00000001, DFS
      Channel: None (0x00000000)
      Previous Session Id: 0x0000000000000000
    ▼ Security Blob: 60820c9306062b0601050502a0820c8730820c83a0243022...
        Offset: 0x00000058
        Length: 3223
      ▼ GSS-API Generic Security Service Application Program Interface
          OID: 1.3.6.1.5.5.2 (SPNEGO - Simple Protected Negotiation)
        ▼ Simple Protected Negotiation
          ▼ negTokenInit
            ▼ mechTypes: 3 items
                MechType: 1.2.840.48018.1.2.2 (MS KRB5 - Microsoft Kerberos 5)
                MechType: 1.2.840.113554.1.2.2 (KRB5 - Kerberos 5)
                MechType: 1.3.6.1.4.1.311.2.2.10 (NTLMSSP - Microsoft NTLM Security Support Provider)
              mechToken: 60820c5106092a864886f71201020201006e820c4030820c...
            ▶ krb5_blob: 60820c5106092a864886f71201020201006e820c4030820c...
```

Kerberos Network Traffic With Trusts

- Client (administrator@W2012R2-L4.BASE) (HW 00:00:00:09:00:01)
- DC in Client-Domain (W2012R2-L4.BASE) (HW 00:00:00:09:01:83)

- Forest-Trust between W2012R2-L4.BASE and W4EDOM-L4.BASE

- DC in Server-Domain (W4EDOM-L4.BASE) (HW 00:00:00:09:01:33)
- Server (w2008r8-132) in W4EDOM-L4.BASE (HW 00:00:00:09:01:32)

- Access to \\w2008r2-132.w4edom-l4.base using Kerberos

AS-REQ	administrator@W2012R2-L4.BASE	00:00:00:09:00:01	00:00:00:09:01:83
AS-REP	krbtgt/W2012R2-L4.BASE@W2012R2-L4.BASE	00:00:00:09:01:83	00:00:00:09:00:01
TGS-REQ	cifs/w2008r2-133.w4edom-l4.base@W2012R2-L4.BASE	00:00:00:09:00:01	00:00:00:09:01:83
TGS-REP	krbtgt/W4EDOM-L4.BASE@W2012R2-L4.BASE	00:00:00:09:01:83	00:00:00:09:00:01
TGS-REQ	cifs/w2008r2-133.w4edom-l4.base@W4EDOM-L4.BASE	00:00:00:09:00:01	00:00:00:09:01:33
TGS-REP	cifs/w2008r2-133.w4edom-l4.base@W4EDOM-L4.BASE	00:00:00:09:01:33	00:00:00:09:00:01
Session Setup Request		00:00:00:09:00:01	00:00:00:09:01:32
Session Setup Response		00:00:00:09:01:32	00:00:00:09:00:01

- The client talks to DCs directly.
- The server gets the authorization data from the kerberos ticket

NTLMSSP Network Traffic With Trusts

- Client (administrator@W2012R2-L4.BASE) (HW 00:00:00:09:00:01)
- DC in Client-Domain (W2012R2-L4.BASE) (HW 00:00:00:09:01:83)

- Forest-Trust between W2012R2-L4.BASE and W4EDOM-L4.BASE

- DC in Server-Domain (W4EDOM-L4.BASE) (HW 00:00:00:09:01:33)
- Server (w2008r8-132) in W4EDOM-L4.BASE (HW 00:00:00:09:01:32)

- Access to \\w2008r2-132.w4edom-l4.base using NTLMSSP

Session Setup Request, NTLMSSP_NEGOTIATE	00:00:00:09:00:01	00:00:00:09:01:32
Session Setup Response, Error: STATUS_MORE_PROCESSING_REQUIRED, NTLMSSP…	00:00:00:09:01:32	00:00:00:09:00:01
Session Setup Request, NTLMSSP_AUTH, User: W2012R2-L4.BASE\administrator	00:00:00:09:00:01	00:00:00:09:01:32
NetrLogonSamLogonEx request	00:00:00:09:01:32	00:00:00:09:01:33
NetrLogonSamLogonWithFlags request	00:00:00:09:01:33	00:00:00:09:01:83
NetrLogonSamLogonWithFlags response	00:00:00:09:01:83	00:00:00:09:01:33
NetrLogonSamLogonEx response	00:00:00:09:01:33	00:00:00:09:01:32
Session Setup Response	00:00:00:09:01:32	00:00:00:09:00:01

- The server talks to the DC in its own domain only.
- The DC may forward the request to trusted domains.

The result of a successful authentication

- Inputs to authentication:
  - The client typically provides a full qualified username together with a password.
  - Smartcards can also be used to do Kerberos (PKINIT) authentication.
- Output from authentication:
  - The target server needs to make sure the client is authenticated.
  - Typically client and server negotiate a session key.
  - The target server gets an authorization token for the authenticated user.
  - The authorization token is contained in the Kerberos service ticket.
  - netr_LogonSamLogon[WithFlags,Ex]() provides the authorization token for NTLMSSP.

The result of a successful authentication

- Inputs to authentication:
  - The client typically provides a full qualified username together with a password.
  - Smartcards can also be used to do Kerberos (PKINIT) authentication.
- Output from authentication:
  - The target server needs to make sure the client is authenticated.
  - Typically client and server negotiate a session key.
  - The target server gets an authorization token for the authenticated user.
  - The authorization token is contained in the Kerberos service ticket.
  - netr_LogonSamLogon[WithFlags,Ex]() provides the authorization token for NTLMSSP.

The authorization token

- Elements in the token:
  - It contains things like username, fullname, logon_domain, various timestamps.
  - The most important information is the list of group memberships.
- The token provided by the "trusted" domain:
  - Needs to be expanded with local groups on the "trusting" side.
  - Needs to be do SID-Filtering on "trusting" side to avoid faked group memberships.
  - The exact SID-Filtering rules depend on the trustAttribute values.
  - It is important to do the expanding and filtering on all trust boundaries of a transitive chain.
  - Currently Samba does not do any SID-Filtering at all!
- In Samba we use 'struct auth_session_info' for the expanded token:
  - It contains a list of SIDS.
  - The details of the Windows user.
  - It contains a uid and a list of gid's.
  - The unix username.

The authorization token

- Elements in the token:
  - It contains things like username, fullname, logon_domain, various timestamps.
  - The most important information is the list of group memberships.
- The token provided by the "trusted" domain:
  - Needs to be expanded with local groups on the "trusting" side.
  - Needs to be do SID-Filtering on "trusting" side to avoid faked group memberships.
  - The exact SID-Filtering rules depend on the trustAttribute values.
  - It is important to do the expanding and filtering on all trust boundaries of a transitive chain.
  - Currently Samba does not do any SID-Filtering at all!
- In Samba we use 'struct auth_session_info' for the expanded token:
  - It contains a list of SIDS.
  - The details of the Windows user.
  - It contains a uid and a list of gid's.
  - The unix username.

The authorization token

- Elements in the token:
  - It contains things like username, fullname, logon_domain, various timestamps.
  - The most important information is the list of group memberships.
- The token provided by the "trusted" domain:
  - Needs to be expanded with local groups on the "trusting" side.
  - Needs to be do SID-Filtering on "trusting" side to avoid faked group memberships.
  - The exact SID-Filtering rules depend on the trustAttribute values.
  - It is important to do the expanding and filtering on all trust boundaries of a transitive chain.
  - Currently Samba does not do any SID-Filtering at all!
- In Samba we use 'struct auth_session_info' for the expanded token:
  - It contains a list of SIDS.
  - The details of the Windows user.
  - It contains a uid and a list of gid's.
  - The unix username.

Authorization Token without Authentication (Part1)

- There're some situations when a service needs to impersonate a user locally:
  - This can happen without getting an authentication for that user.
  - SSH public-key authentication, sudo or nfs3 access are tyipical usecases.

- Getting an authorization token without authentication is tricky:
  - Currently winbindd tries to get the 'tokenGroups' of the user object via LDAP
  - In situations with trusted domains it means that winbindd will try to connect a DC of the user's primary domain without having a direct trust to it.
  - There're a lot of situations where this doesn't work, e.g. with OUTBOUND only trusts.
  - It is a very hard task because the expanding and filtering at the trust boundaries of the transitive chain can't be simulated.
  - So the result is often wrong!

Authorization Token without Authentication (Part1)

- There're some situations when a service needs to impersonate a user locally:
  - This can happen without getting an authentication for that user.
  - SSH public-key authentication, sudo or nfs3 access are tyipical usecases.

- Getting an authorization token without authentication is tricky:
  - Currently winbindd tries to get the 'tokenGroups' of the user object via LDAP
  - In situations with trusted domains it means that winbindd will try to connect a DC of the users primary domain without having a direct trust to it.
  - There're a lot of situations where this doesn't work, e.g. with OUTBOUND only trusts.
  - It is a very hard task because the expanding and filtering at the trust boundaries of the transitive chain can't be simulated.
  - So the result is often wrong!

Authorization Token without Authentication (Part2)

- The only reliable solution is S4U2Self:
  - S4U2Self ([MS-SFU]), a Kerberos extension, allows a service to ask a KDC for an service ticket for a given user.
  - Sadly there're quite some bugs in current versions of MIT Kerberos and Heimdal.
  - But the bugs can be fixed.

Using a strict trust routing table (Part1)

- Making efficient and robust usage of trust relationships:
  - It is required to construct a routing table that knows about routing via transitive trusts.
  - The table is constructed by the list of direct trusts and their (optionally) related forest information.
  - The goal is that communication only appears between direct trusts.

Using a strict trust routing table (Part2)

- Using the routing table for Kerberos:
  - The routing table is mainly used in the KDC, which means the basics for two-way (INBOUND and OUTBOUD) trusts as an AD DC are already in place.
  - The client just talk to a KDC in the primary domain and follow referrals, it doesn't really need the routing table.
- Using the routing table for NTLMSSP:
  - It also needs to be used the NETLOGON and LSA servers in order to find out if a requests should be routed via winbindd to a trusted domain.
  - The routing table needs to be used within winbindd.
  - This will make the code much more robust as a domain member.
  - And it will also provide the basics for two-way (INBOUND and OUTBOUD) trusts as an AD DC.

Using a strict trust routing table (Part2)

- Using the routing table for Kerberos:
  - The routing table is mainly used in the KDC, which means the basics for two-way (INBOUND and OUTBOUD) trusts as an AD DC are already in place.
  - The client just talk to a KDC in the primary domain and follow referrals, it doesn't really need the routing table.
- Using the routing table for NTLMSSP:
  - It also needs to be used the NETLOGON and LSA servers in order to find out if a requests should be routed via winbindd to a trusted domain.
  - The routing table needs to be used within winbindd.
  - This will make the code much more robust as a domain member.
  - And it will also provide the basics for two-way (INBOUND and OUTBOUD) trusts as an AD DC.

New Kerberos Features (Part 1)

- Samba provided features
  - We try to emulate the features of the Windows 2008R2 DC functional level
  - Everything else will need some development effort.
- Windows 2012 introduced support for Kerberos FAST:
  - Typically Kerberos authentication requests (AS-Req) use the password of the user to encrypt a timestamp.
  - This allows attackers to do offline dictionary against the users typically less random password.
  - Typically the passwords of trust accounts, e.g. computer accounts have trully random passwords.
  - The solution is to use a ticket created with the computer account to protect the users AS-REQ.

New Kerberos Features (Part 1)

- Samba provided features
  - We try to emulate the features of the Windows 2008R2 DC functional level
  - Everything else will need some development effort.
- Windows 2012 introduced support for Kerberos FAST:
  - Typically Kerberos authentication requests (AS-Req) use the password of the user to encrypt a timestamp.
  - This allows attackers to do offline dictionary against the users typically less random password.
  - Typically the passwords of trust accounts, e.g. computer accounts have trully random passwords.
  - The solution is to use a ticket created with the computer account to protect the users AS-REQ.

New Kerberos Features (Part 2)

- Windows 2012 introduced support for Compound Identities:
  - If the client uses FAST, the KDC is able to know from which device (computer) the user is coming.
  - This KDC add a new PAC_DEVICE_INFO element to the Kerberos ticket.
  - As result the autorization token of the user will also have information of the device, which can be used to use more advanced access restrictions.
- Windows 2012 introduced support for CLAIMS:
  - An administrator can define and assign "claims".
  - It allows more flexible access control beside using groups.
  - The Kerberos ticket will contain PAC_CLIENT_CLAIMS_INFO and PAC_DEVICE_CLAIMS_INFO
  - More research is required to fully understand how CLAIMS work.

New Kerberos Features (Part 2)

- Windows 2012 introduced support for Compound Identities:
  - If the client uses FAST, the KDC is able to know from which device (computer) the user is coming.
  - This KDC add a new PAC_DEVICE_INFO element to the Kerberos ticket.
  - As result the autorization token of the user will also have information of the device, which can be used to use more advanced access restrictions.
- Windows 2012 introduced support for CLAIMS:
  - An administrator can define and assign "claims".
  - It allows more flexible access control beside using groups.
  - The Kerberos ticket will contain PAC_CLIENT_CLAIMS_INFO and PAC_DEVICE_CLAIMS_INFO
  - More research is required to fully understand how CLAIMS work.

New Kerberos Features (Part 3)

- Windows 2016 introduced support for Privileged Identity Management (PIM):
  - This feature will add timed group memberships
  - E.g. an administrative user will only be a member of the domain admins group for an hour.
  - The lifetime for Kerberos tickets is very limited compared to the default of 10 hours, with a renew up to a week.
  - There's also a special forest trust mode for PIM.
  - More research is required to fully understand how PIM work.

Useful links

- TECHNET: "How Domain and Forest Trusts Work"
- [MS-AUTHSOD]
- [MS-PAC]
- [MS-LSAD]
- [MS-LSAT]
- [MS-DTYP]

Questions?

- Stefan Metzmacher, metze@samba.org
- https://www.sernet.com