



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/>



- ▶ 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)



- ▶ Windows Domains, Forests and Trusts
- ▶ Netlogon Secure Channel
- ▶ Authentication Protocols
- ▶ Authorization Token/S4U2Self
- ▶ Selective Authentication/Restrict NTLM
- ▶ New Kerberos Features
- ▶ Trust Routing Table
- ▶ Improvements in Samba
- ▶ Further Authentication Topics
- ▶ Questions?
- ▶ Useful links



- ▶ Trust Types (only relevant ones):
 - ▶ LSA_TRUST_TYPE_DOWNLEVEL (NT4)
 - ▶ LSA_TRUST_TYPE_UPLEVEL (AD)
- ▶ Trust Directions:
 - ▶ LSA_TRUST_DIRECTION_INBOUND
 - ▶ LSA_TRUST_DIRECTION_OUTBOUND (like on a domain member)
- ▶ For further details see my SambaXP talk.

Trust Types and Directions (low level)



- ▶ Trust Types (only relevant ones):
 - ▶ LSA_TRUST_TYPE_DOWNLEVEL (NT4)
 - ▶ LSA_TRUST_TYPE_UPLEVEL (AD)
- ▶ Trust Directions:
 - ▶ LSA_TRUST_DIRECTION_INBOUND
 - ▶ LSA_TRUST_DIRECTION_OUTBOUND (like on a domain member)
- ▶ For further details see my SambaXP talk.

Trust Types and Directions (low level)



- ▶ Trust Types (only relevant ones):
 - ▶ LSA_TRUST_TYPE_DOWNLEVEL (NT4)
 - ▶ LSA_TRUST_TYPE_UPLEVEL (AD)
- ▶ Trust Directions:
 - ▶ LSA_TRUST_DIRECTION_INBOUND
 - ▶ LSA_TRUST_DIRECTION_OUTBOUND (like on a domain member)
- ▶ For further details see my SambaXP talk.

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, /* only kerberos */
    LSA_TRUST_ATTRIBUTE_QUARANTINED_DOMAIN     = 0x00000004,
    LSA_TRUST_ATTRIBUTE_FOREST_TRANSITIVE     = 0x00000008, /* cross forest trust */
    LSA_TRUST_ATTRIBUTE_CROSS_ORGANIZATION    = 0x00000010, /* selective auth */
    LSA_TRUST_ATTRIBUTE_WITHIN_FOREST        = 0x00000020, /* transitive by default */
    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;
```

Trust Types (high level, Part 1)



- ▶ Workstation (Domain Member) Trust
- ▶ External Domain Trust
- ▶ Forest Trust
- ▶ Parent Child Trusts (Within Forest)
- ▶ Tree Root Trusts (Within Forest)
- ▶ Shortcut Trust (Within Forest)
- ▶ For further details see my SambaXP talk.

Draft !!!

Trust Types (high level, Part 1)



- ▶ Workstation (Domain Member) Trust
- ▶ External Domain Trust
- ▶ Forest Trust
- ▶ Parent Child Trusts (Within Forest)
- ▶ Tree Root Trusts (Within Forest)
- ▶ Shortcut Trust (Within Forest)
- ▶ For further details see my SambaXP talk.

Draft !!!

Trust Types (high level, Part 1)



- ▶ Workstation (Domain Member) Trust
- ▶ External Domain Trust
- ▶ Forest Trust
- ▶ Parent Child Trusts (Within Forest)
- ▶ Tree Root Trusts (Within Forest)
- ▶ Shortcut Trust (Within Forest)
- ▶ For further details see my SambaXP talk.

Draft !!!

Trust Types (high level, Part 1)



- ▶ Workstation (Domain Member) Trust
- ▶ External Domain Trust
- ▶ Forest Trust
- ▶ Parent Child Trusts (Within Forest)
- ▶ Tree Root Trusts (Within Forest)
- ▶ Shortcut Trust (Within Forest)
- ▶ For further details see my SambaXP talk.

Trust Types (high level, Part 1)



- ▶ Workstation (Domain Member) Trust
- ▶ External Domain Trust
- ▶ Forest Trust
- ▶ Parent Child Trusts (Within Forest)
- ▶ Tree Root Trusts (Within Forest)
- ▶ Shortcut Trust (Within Forest)
- ▶ For further details see my SambaXP talk.

Trust Types (high level, Part 1)



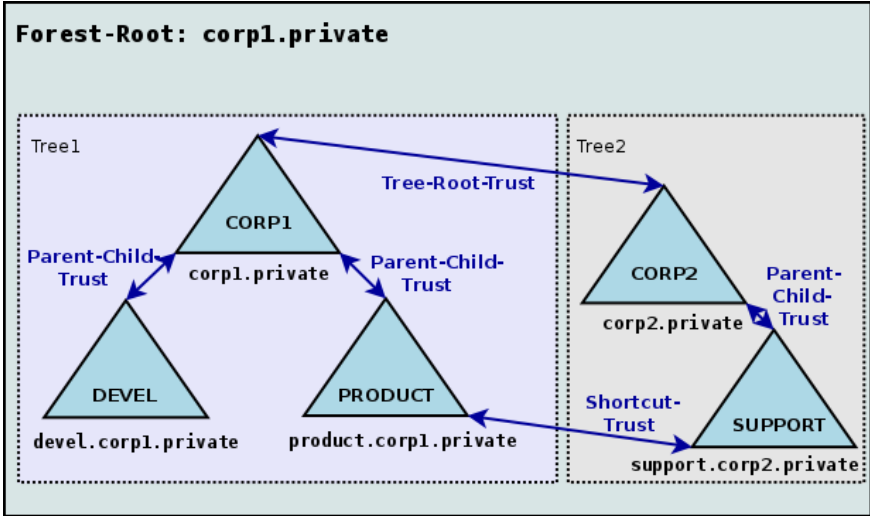
- ▶ Workstation (Domain Member) Trust
- ▶ External Domain Trust
- ▶ Forest Trust
- ▶ Parent Child Trusts (Within Forest)
- ▶ Tree Root Trusts (Within Forest)
- ▶ Shortcut Trust (Within Forest)
- ▶ For further details see my SambaXP talk.

Trust Types (high level, Part 1)



- ▶ Workstation (Domain Member) Trust
- ▶ External Domain Trust
- ▶ Forest Trust
- ▶ Parent Child Trusts (Within Forest)
- ▶ Tree Root Trusts (Within Forest)
- ▶ Shortcut Trust (Within Forest)
- ▶ For further details see my SambaXP talk.

Layout of an Active Directory Forest (with multiple Trees)



Forest Information (with multiple Trees)



- ▶ 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



- ▶ 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 based on DCERPC, see [MS-NRPC].
- ▶ 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.
- ▶ Forest Trust Information is available via NETLOGON:
 - ▶ `netr_GetForestTrustInformation()` is used to get the details
- ▶ For further details see my SambaXP talk.



- ▶ 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 based on DCERPC, see [MS-NRPC].
- ▶ 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.
- ▶ Forest Trust Information is available via NETLOGON:
 - ▶ `netr_GetForestTrustInformation()` is used to get the details
- ▶ For further details see my SambaXP talk.



- ▶ 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 based on DCERPC, see [MS-NRPC].
- ▶ 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.
- ▶ Forest Trust Information is available via NETLOGON:
 - ▶ netr_GetForestTrustInformation() is used to get the details
- ▶ For further details see my SambaXP talk.



- ▶ 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 based on DCERPC, see [MS-NRPC].
- ▶ 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.
- ▶ Forest Trust Information is available via NETLOGON:
 - ▶ netr_GetForestTrustInformation() is used to get the details
- ▶ For further details see my SambaXP talk.

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 (w2008r2-132) 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 typical 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 (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 typical 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.
- ▶ Details of S4U2Self:
 - ▶ The service needs a TGT for the user realm first.
 - ▶ Referrals are followed from the service realm to the user realm.
 - ▶ Then it requests a S4U2Self Ticket specifying the impersonated user principal and the service principal.
 - ▶ In order to get a usable ticket referrals are followed back to the service realm.
 - ▶ This requires a two-way trust.

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.
- ▶ Details of S4U2Self:
 - ▶ The service needs a TGT for the user realm first.
 - ▶ Referrals are followed from the service realm to the user realm.
 - ▶ Then it requests a S4U2Self Ticket specifying the impersonated user principal and the service principal.
 - ▶ In order to get a usable ticket referrals are followed back to the service realm.
 - ▶ This requires a two-way trust.



- ▶ Forest/Domain-wide Authentication (the default) allows:
 - ▶ Authentication of each principal of the trusted forest/domain
 - ▶ Authentication to each service in the trusting forest/domain
- ▶ Authorization is handled by:
 - ▶ Using ACLs on individual resources (objects, files, ...)
 - ▶ Access might be granted just by "Authenticated Users" ACEs.
- ▶ One-way trusts:
 - ▶ Often used to limit the authentication between organizations.
 - ▶ Make the use of S4U2Self impossible.



- ▶ Forest/Domain-wide Authentication (the default) allows:
 - ▶ Authentication of each principal of the trusted forest/domain
 - ▶ Authentication to each service in the trusting forest/domain
- ▶ Authorization is handled by:
 - ▶ Using ACLs on individual resources (objects, files, ...)
 - ▶ Access might be granted just by "Authenticated Users" ACEs.
- ▶ One-way trusts:
 - ▶ Often used to limit the authentication between organizations.
 - ▶ Make the use of S4U2Self impossible.



- ▶ Forest/Domain-wide Authentication (the default) allows:
 - ▶ Authentication of each principal of the trusted forest/domain
 - ▶ Authentication to each service in the trusting forest/domain
- ▶ Authorization is handled by:
 - ▶ Using ACLs on individual resources (objects, files, ...)
 - ▶ Access might be granted just by "Authenticated Users" ACEs.
- ▶ One-way trusts:
 - ▶ Often used to limit the authentication between organizations.
 - ▶ Make the use of S4U2Self impossible.

Selective Authentication (Cross Organization Trusts)

- ▶ Trusts can be marked for selective authentication:
 - ▶ Using LSA_TRUST_ATTRIBUTE_CROSS_ORGANIZATION
 - ▶ The trusting end adds the OTHER_ORGANIZATION SID (S-1-5-1000) to any token
 - ▶ By default authentication of trusted principals to trusting services is rejected with STATUS_AUTHENTICATION_FIREWALL_FAILED.
- ▶ Selective authentication checking:
 - ▶ Only done if the token contains S-1-5-1000
 - ▶ The "AllowedToAuthenticateTo" extended access right is required on the AD object of the service.
- ▶ Advantages of selective authentication:
 - ▶ It is much more flexible than the all or nothing of one-way trusts.
 - ▶ It allows S4U2Self to work.
- ▶ Status of selective authentication within Samba:
 - ▶ Not implemented yet, similar to all SID expanding/filtering.

Selective Authentication (Cross Organization Trusts)

- ▶ Trusts can be marked for selective authentication:
 - ▶ Using `LSA_TRUST_ATTRIBUTE_CROSS_ORGANIZATION`
 - ▶ The trusting end adds the `OTHER_ORGANIZATION SID (S-1-5-1000)` to any token
 - ▶ By default authentication of trusted principals to trusting services is rejected with `STATUS_AUTHENTICATION_FIREWALL_FAILED`.
- ▶ Selective authentication checking:
 - ▶ Only done if the token contains `S-1-5-1000`
 - ▶ The "AllowedToAuthenticateTo" extended access right is required on the AD object of the service.
- ▶ Advantages of selective authentication:
 - ▶ It is much more flexible than the all or nothing of one-way trusts.
 - ▶ It allows `S4U2Self` to work.
- ▶ Status of selective authentication within Samba:
 - ▶ Not implemented yet, similar to all SID expanding/filtering.

Selective Authentication (Cross Organization Trusts)

- ▶ Trusts can be marked for selective authentication:
 - ▶ Using `LSA_TRUST_ATTRIBUTE_CROSS_ORGANIZATION`
 - ▶ The trusting end adds the `OTHER_ORGANIZATION SID (S-1-5-1000)` to any token
 - ▶ By default authentication of trusted principals to trusting services is rejected with `STATUS_AUTHENTICATION_FIREWALL_FAILED`.
- ▶ Selective authentication checking:
 - ▶ Only done if the token contains `S-1-5-1000`
 - ▶ The "AllowedToAuthenticateTo" extended access right is required on the AD object of the service.
- ▶ Advantages of selective authentication:
 - ▶ It is much more flexible than the all or nothing of one-way trusts.
 - ▶ It allows `S4U2Self` to work.
- ▶ Status of selective authentication within Samba:
 - ▶ Not implemented yet, similar to all SID expanding/filtering.

Selective Authentication (Cross Organization Trusts)

- ▶ Trusts can be marked for selective authentication:
 - ▶ Using LSA_TRUST_ATTRIBUTE_CROSS_ORGANIZATION
 - ▶ The trusting end adds the OTHER_ORGANIZATION SID (S-1-5-1000) to any token
 - ▶ By default authentication of trusted principals to trusting services is rejected with STATUS_AUTHENTICATION_FIREWALL_FAILED.
- ▶ Selective authentication checking:
 - ▶ Only done if the token contains S-1-5-1000
 - ▶ The "AllowedToAuthenticateTo" extended access right is required on the AD object of the service.
- ▶ Advantages of selective authentication:
 - ▶ It is much more flexible than the all or nothing of one-way trusts.
 - ▶ It allows S4U2Self to work.
- ▶ Status of selective authentication within Samba:
 - ▶ Not implemented yet, similar to all SID expanding/filtering.

Restrict NTLM... (Part1)



- ▶ Windows has several ways to restrict the use of NTLM based authentication:
- ▶ Client:
 - ▶ Restrict NTLM: Outgoing NTLM traffic to remote servers
 - ▶ Restrict NTLM: Add remote server exceptions for NTLM authentication
 - ▶ NT_STATUS_NOT_SUPPORTED is generated if NTLM is not allowed
- ▶ Server:
 - ▶ Restrict NTLM: Incoming NTLM Traffic
 - ▶ Restrict NTLM: Audit incoming NTLM Traffic
 - ▶ NT_STATUS_NOT_SUPPORTED is generated if NTLM is not allowed
- ▶ Domain Controller:
 - ▶ Restrict NTLM: NTLM authentication in this domain
 - ▶ Restrict NTLM: Add server exceptions in this domain
 - ▶ Restrict NTLM: Audit NTLM authentication in this domain
 - ▶ NT_STATUS_NTLM_BLOCKED is generated if NTLM is not allowed



Restrict NTLM... (Part1)



- ▶ Windows has several ways to restrict the use of NTLM based authentication:
- ▶ Client:
 - ▶ Restrict NTLM: Outgoing NTLM traffic to remote servers
 - ▶ Restrict NTLM: Add remote server exceptions for NTLM authentication
 - ▶ NT_STATUS_NOT_SUPPORTED is generated if NTLM is not allowed
- ▶ Server:
 - ▶ Restrict NTLM: Incoming NTLM Traffic
 - ▶ Restrict NTLM: Audit incoming NTLM Traffic
 - ▶ NT_STATUS_NOT_SUPPORTED is generated if NTLM is not allowed
- ▶ Domain Controller:
 - ▶ Restrict NTLM: NTLM authentication in this domain
 - ▶ Restrict NTLM: Add server exceptions in this domain
 - ▶ Restrict NTLM: Audit NTLM authentication in this domain
 - ▶ NT_STATUS_NTLM_BLOCKED is generated if NTLM is not allowed

Restrict NTLM... (Part1)



- ▶ Windows has several ways to restrict the use of NTLM based authentication:
- ▶ Client:
 - ▶ Restrict NTLM: Outgoing NTLM traffic to remote servers
 - ▶ Restrict NTLM: Add remote server exceptions for NTLM authentication
 - ▶ NT_STATUS_NOT_SUPPORTED is generated if NTLM is not allowed
- ▶ Server:
 - ▶ Restrict NTLM: Incoming NTLM Traffic
 - ▶ Restrict NTLM: Audit Incoming NTLM Traffic
 - ▶ NT_STATUS_NOT_SUPPORTED is generated if NTLM is not allowed
- ▶ Domain Controller:
 - ▶ Restrict NTLM: NTLM authentication in this domain
 - ▶ Restrict NTLM: Add server exceptions in this domain
 - ▶ Restrict NTLM: Audit NTLM authentication in this domain
 - ▶ NT_STATUS_NTLM_BLOCKED is generated if NTLM is not allowed

Restrict NTLM... (Part1)



- ▶ Windows has several ways to restrict the use of NTLM based authentication:
- ▶ Client:
 - ▶ Restrict NTLM: Outgoing NTLM traffic to remote servers
 - ▶ Restrict NTLM: Add remote server exceptions for NTLM authentication
 - ▶ NT_STATUS_NOT_SUPPORTED is generated if NTLM is not allowed
- ▶ Server:
 - ▶ Restrict NTLM: Incoming NTLM Traffic
 - ▶ Restrict NTLM: Audit Incoming NTLM Traffic
 - ▶ NT_STATUS_NOT_SUPPORTED is generated if NTLM is not allowed
- ▶ Domain Controller:
 - ▶ Restrict NTLM: NTLM authentication in this domain
 - ▶ Restrict NTLM: Add server exceptions in this domain
 - ▶ Restrict NTLM: Audit NTLM authentication in this domain
 - ▶ NT_STATUS_NTLM_BLOCKED is generated if NTLM is not allowed

Restrict NTLM... (Part2)



- ▶ With Samba 4.7 we'll have the following options "ntlm auth":
 - ▶ "ntlmv1-permitted" (alias "yes") - Allow NTLMv1 and above for all clients.
 - ▶ "ntlmv2-only" (alias "no") - Do not allow NTLMv1 to be used, but permit NTLMv2.
 - ▶ "mschapv2-and-ntlmv2-only" - Only allow NTLMv1 when the client promises that it is providing MSCHAPv2 authentication (such as the ntlm_auth tool).
 - ▶ "disabled" - Do not accept NTLM (or LanMan) authentication of any level, nor permit NTLM password changes.
 - ▶ The default is "ntlmv2-only".
- ▶ Before Samba 4.7:
 - ▶ We just had "yes" and "no", just controlling NTLMv1 usage.
 - ▶ The default was "no".
- ▶ In future:
 - ▶ We may implement more flexible schema similar to Windows
 - ▶ This would allow us to keep NTLM alive for specific servers.

Restrict NTLM... (Part2)



- ▶ With Samba 4.7 we'll have the following options "ntlm auth":
 - ▶ "ntlmv1-permitted" (alias "yes") - Allow NTLMv1 and above for all clients.
 - ▶ "ntlmv2-only" (alias "no") - Do not allow NTLMv1 to be used, but permit NTLMv2.
 - ▶ "mschapv2-and-ntlmv2-only" - Only allow NTLMv1 when the client promises that it is providing MSCHAPv2 authentication (such as the ntlm_auth tool).
 - ▶ "disabled" - Do not accept NTLM (or LanMan) authentication of any level, nor permit NTLM password changes.
 - ▶ The default is "ntlmv2-only".
- ▶ Before Samba 4.7:
 - ▶ We just had "yes" and "no", just controlling NTLMv1 usage.
 - ▶ The default was "no".
- ▶ In future:
 - ▶ We may implement more flexible schema similar to Windows
 - ▶ This would allow us to keep NTLM alive for specific servers.

Restrict NTLM... (Part2)



- ▶ With Samba 4.7 we'll have the following options "ntlm auth":
 - ▶ "ntlmv1-permitted" (alias "yes") - Allow NTLMv1 and above for all clients.
 - ▶ "ntlmv2-only" (alias "no") - Do not allow NTLMv1 to be used, but permit NTLMv2.
 - ▶ "mschapv2-and-ntlmv2-only" - Only allow NTLMv1 when the client promises that it is providing MSCHAPv2 authentication (such as the ntlm_auth tool).
 - ▶ "disabled" - Do not accept NTLM (or LanMan) authentication of any level, nor permit NTLM password changes.
 - ▶ The default is "ntlmv2-only".
- ▶ Before Samba 4.7:
 - ▶ We just had "yes" and "no", just controlling NTLMv1 usage.
 - ▶ The default was "no"
- ▶ In future:
 - ▶ We may implement more flexible schema similar to Windows
 - ▶ This would allow us to keep NTLM alive for specific servers.

Restrict NTLM... (Part2)



- ▶ With Samba 4.7 we'll have the following options "ntlm auth":
 - ▶ "ntlmv1-permitted" (alias "yes") - Allow NTLMv1 and above for all clients.
 - ▶ "ntlmv2-only" (alias "no") - Do not allow NTLMv1 to be used, but permit NTLMv2.
 - ▶ "mschapv2-and-ntlmv2-only" - Only allow NTLMv1 when the client promises that it is providing MSCHAPv2 authentication (such as the ntlm_auth tool).
 - ▶ "disabled" - Do not accept NTLM (or LanMan) authentication of any level, nor permit NTLM password changes.
 - ▶ The default is "ntlmv2-only".
- ▶ Before Samba 4.7:
 - ▶ We just had "yes" and "no", just controlling NTLMv1 usage.
 - ▶ The default was "no"
- ▶ In future:
 - ▶ We may implement more flexible schema similar to Windows
 - ▶ This would allow us to keep NTLM alive for specific servers.

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 KDC resource group compression:
 - ▶ This reduced the size of the PAC with a large number of resource group memberships.
 - ▶ Samba should implement this once we implement the SID expanding/filtering.
- ▶ Windows 2012 introduced support for Kerberos FAST (armoring):
 - ▶ 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 KDC resource group compression:
 - ▶ This reduced the size of the PAC with a large number of resource group memberships.
 - ▶ Samba should implement this once we implement the SID expanding/filtering.
- ▶ Windows 2012 introduced support for Kerberos FAST (armoring):
 - ▶ 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 KDC resource group compression:
 - ▶ This reduced the size of the PAC with a large number of resource group memberships.
 - ▶ Samba should implement this once we implement the SID expanding/filtering.
- ▶ Windows 2012 introduced support for Kerberos FAST (armoring):
 - ▶ 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 Branch Aware clients:
 - ▶ The client can tell on RODC not to forward a TGS-REQ
 - ▶ The client can force a forward to an RWDC
- ▶ 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 authorization 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 Branch Aware clients:
 - ▶ The client can tell on RODC not to forward a TGS-REQ
 - ▶ The client can force a forward to an RWDC
- ▶ 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 authorization 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 Branch Aware clients:
 - ▶ The client can tell on RODC not to forward a TGS-REQ
 - ▶ The client can force a forward to an RWDC
- ▶ 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 authorization 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 2012R2 introduced the Protected Users Security Group
 - ▶ SID: S-1-5-21-`<domain>`-525
 - ▶ Members can use Kerberos with AES keys
 - ▶ Members can not use Kerberos delegation
 - ▶ The TGT is only valid for 4 hours by default
 - ▶ Credentials are never cached
- ▶ Windows 2012R2 introduced Authentication Policies and Authentication Policy Silos:
 - ▶ Like "Selective Authentication" within a Forest.
 - ▶ More research is required to fully understand all details
- ▶ 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.
 - ▶ TGTs are only valid for a short time.
 - ▶ There's also a special forest trust mode for PIM.
 - ▶ More research is required to fully understand how PIM works.

New Kerberos Features (Part 3)



- ▶ Windows 2012R2 introduced the Protected Users Security Group
 - ▶ SID: S-1-5-21-<domain>-525
 - ▶ Members can use Kerberos with AES keys
 - ▶ Members can not use Kerberos delegation
 - ▶ The TGT is only valid for 4 hours by default
 - ▶ Credentials are never cached
- ▶ Windows 2012R2 introduced Authentication Policies and Authentication Policy Silos:
 - ▶ Like "Selective Authentication" within a Forest.
 - ▶ More research is required to fully understand all details
- ▶ 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.
 - ▶ TGTs are only valid for a short time.
 - ▶ There's also a special forest trust mode for PIM.
 - ▶ More research is required to fully understand how PIM works.

New Kerberos Features (Part 3)



- ▶ Windows 2012R2 introduced the Protected Users Security Group
 - ▶ SID: S-1-5-21-<domain>-525
 - ▶ Members can use Kerberos with AES keys
 - ▶ Members can not use Kerberos delegation
 - ▶ The TGT is only valid for 4 hours by default
 - ▶ Credentials are never cached
- ▶ Windows 2012R2 introduced Authentication Policies and Authentication Policy Silos:
 - ▶ Like "Selective Authentication" within a Forest.
 - ▶ More research is required to fully understand all details
- ▶ 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.
 - ▶ TGTs are only valid for a short time.
 - ▶ There's also a special forest trust mode for PIM.
 - ▶ More research is required to fully understand how PIM works.



- ▶ We need:
 - ▶ A scalable and robust authentication subsystem on domain members.
 - ▶ Full support for trusted domains/forests as active directory domain controller.
- ▶ Most of the logic is handled by winbindd:
 - ▶ The requirements of DCs and domain members are similar
 - ▶ We just need to correct abstraction that can handle all possible trust flavours.
- ▶ Limit avoidable network communication:
 - ▶ Use idmap backends with IDMAP_TYPE_BOTH support => no LookupSid anymore
 - ▶ No domain controller communication when accepting Kerberos authentication
 - ▶ Reduce DNS and CLDAP requests, especially from the Kerberos libraries



- ▶ We need:
 - ▶ A scalable and robust authentication subsystem on domain members.
 - ▶ Full support for trusted domains/forests as active directory domain controller.
- ▶ Most of the logic is handled by winbindd:
 - ▶ The requirements of DCs and domain members are similar
 - ▶ We just need to correct abstraction that can handle all possible trust flavours.
- ▶ Limit avoidable network communication:
 - ▶ Use idmap backend with IDMAP_TYPE_BOTH support => no LookupSid anymore
 - ▶ No domain controller communication when accepting Kerberos authentication
 - ▶ Reduce DNS and CLDAP requests, especially from the Kerberos libraries



- ▶ We need:
 - ▶ A scalable and robust authentication subsystem on domain members.
 - ▶ Full support for trusted domains/forests as active directory domain controller.
- ▶ Most of the logic is handled by winbindd:
 - ▶ The requirements of DCs and domain members are similar
 - ▶ We just need to correct abstraction that can handle all possible trust flavours.
- ▶ Limit avoidable network communication:
 - ▶ Use idmap backends with IDMAP_TYPE_BOTH support => no LookupSid anymore
 - ▶ No domain controller communication when accepting Kerberos authentication
 - ▶ Reduce DNS and CLDAP requests, especially from the Kerberos libraries



- ▶ 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.
 - ▶ Only NETLOGON and LSA LookupSids/Names using Netlogon secure channel.
 - ▶ No SAMR and no LDAP anymore (at least by default)



- ▶ Using the routing table for Kerberos:
 - ▶ The routing table is mainly used in the KDC, which means the basics for two-way (INBOUND and OUTBOUND) trusts as an AD DC are already in place.
 - ▶ The client just talks to a KDC in the primary domain and follows referrals, it doesn't really need the routing table.
- ▶ Using the routing table for NETLOGON:
 - ▶ 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 OUTBOUND) trusts as an AD DC.



- ▶ 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 talks to a KDC in the primary domain and follows 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.

Removing "map untrusted to domain" option



- ▶ When a client authenticates as UNKNOWN\user it get silently mapped to LOCALSAMNAME\user
- ▶ Up to now we fetched a list of trusted domains from winbindd:
 - ▶ This list was used to evaluate if the domain is "untrusted"
 - ▶ "map untrusted to domain = yes/no" controls to what the "untrusted" domain name is mapped to.
 - ▶ But this is completely unreliable, e.g. with one-way trusts and other situations.
- ▶ It's the job of our DC to decide about trusts:
 - ▶ We need to pass on local authentication always (unchanged) to a DC.
 - ▶ NO_SUCH_USER together with authoritative=0 indicates a possible fallback.
 - ▶ We have this fixed by "map untrusted to domain = auto" in Samba 4.7
 - ▶ Samba 4.8 will remove that option completely while keeping the auto behavior.

Removing "map untrusted to domain" option



- ▶ When a client authenticates as UNKNOWN\user it get silently mapped to LOCALSAMNAME\user
- ▶ Up to now we fetched a list of trusted domains from winbindd:
 - ▶ This list was used to evaluate if the domain is "untrusted"
 - ▶ "map untrusted to domain = yes/no" controls to what the "untrusted" domain name is mapped to.
 - ▶ But this is completely unreliable, e.g. with one-way trusts and other situations.
- ▶ It's the job of our DC to decide about trusts:
 - ▶ We need to pass on local authentication always (unchanged) to a DC.
 - ▶ NO_SUCH_USER together with authoritative=0 indicates a possible fallback.
 - ▶ We have this fixed by "map untrusted to domain = auto" in Samba 4.7
 - ▶ Samba 4.8 will remove that option completely while keeping the auto behavior.

Removing "map untrusted to domain" option



- ▶ When a client authenticates as UNKNOWN\user it get silently mapped to LOCALSAMNAME\user
- ▶ Up to now we fetched a list of trusted domains from winbindd:
 - ▶ This list was used to evaluate if the domain is "untrusted"
 - ▶ "map untrusted to domain = yes/no" controls to what the "untrusted" domain name is mapped to.
 - ▶ But this is completely unreliable, e.g. with one-way trusts and other situations.
- ▶ It's the job of our DC to decide about trusts:
 - ▶ We need to pass non local authentication always (unchanged) to a DC.
 - ▶ NO_SUCH_USER together with authoritative=0 indicates a possible fallback.
 - ▶ We have this fixed by "map untrusted to domain = auto" in Samba 4.7
 - ▶ Samba 4.8 will remove that option completely while keeping the auto behavior.

Full async authentication stack (Part1)



old semi-async gensec_update api in Samba:

```
NTSTATUS gensec_update_ev(struct gensec_security *gensec_security,
                        TALLOC_CTX *out_mem_ctx,
                        struct tevent_context *ev,
                        const DATA_BLOB in, DATA_BLOB *out);
```

- ▶ Using gensec_update_ev() as a server:
 - ▶ Was possible for local non-blocking authentication on an AD DC
 - ▶ Is not usable with remote authentication at all
 - ▶ Nested event loops are like threads without mutexes

Async gensec_update api attribute in Samba:

```
struct tevent_req *gensec_update_send(TALLOC_CTX *mem_ctx,
                                      struct tevent_context *ev,
                                      struct gensec_security *gensec_security,
                                      const DATA_BLOB in);
NTSTATUS gensec_update_recv(struct tevent_req *req,
                          TALLOC_CTX *out_mem_ctx,
                          DATA_BLOB *out);
NTSTATUS gensec_update(struct gensec_security *gensec_security,
                      TALLOC_CTX *out_mem_ctx,
                      const DATA_BLOB in, DATA_BLOB *out);
```

Full async authentication stack (Part1)



old semi-async gensec_update api in Samba:

```
NTSTATUS gensec_update_ev(struct gensec_security *gensec_security,
                        TALLOC_CTX *out_mem_ctx,
                        struct tevent_context *ev,
                        const DATA_BLOB in, DATA_BLOB *out);
```

- ▶ Using gensec_update_ev() as a server:
 - ▶ Was possible for local non-blocking authentication on an AD DC
 - ▶ Is not usable with remote authentication at all
 - ▶ Nested event loops are like threads without mutexes

Async gensec_update api attribute in Samba:

```
struct tevent_req *gensec_update_send(TALLOC_CTX *mem_ctx,
                                      struct tevent_context *ev,
                                      struct gensec_security *gensec_security,
                                      const DATA_BLOB in);
NTSTATUS gensec_update_recv(struct tevent_req *req,
                          TALLOC_CTX *out_mem_ctx,
                          DATA_BLOB *out);
NTSTATUS gensec_update_async(struct gensec_security *gensec_security,
                            TALLOC_CTX *out_mem_ctx,
                            const DATA_BLOB in, DATA_BLOB *out);
```

Full async authentication stack (Part1)



old semi-async gensec_update api in Samba:

```
NTSTATUS gensec_update_ev(struct gensec_security *gensec_security,
                        TALLOC_CTX *out_mem_ctx,
                        struct tevent_context *ev,
                        const DATA_BLOB in, DATA_BLOB *out);
```

- ▶ Using gensec_update_ev() as a server:
 - ▶ Was possible for local non-blocking authentication on an AD DC
 - ▶ Is not usable with remote authentication at all
 - ▶ Nested event loops are like threads without mutexes

Async gensec_update api attribute in Samba:

```
struct tevent_req *gensec_update_send(TALLOC_CTX *mem_ctx,
                                      struct tevent_context *ev,
                                      struct gensec_security *gensec_security,
                                      const DATA_BLOB in);

NTSTATUS gensec_update_recv(struct tevent_req *req,
                          TALLOC_CTX *out_mem_ctx,
                          DATA_BLOB *out);

NTSTATUS gensec_update(struct gensec_security *gensec_security,
                     TALLOC_CTX *out_mem_ctx,
                     const DATA_BLOB in, DATA_BLOB *out);
```

Full async authentication stack (Part2)



- ▶ Changing the callers to:
 - ▶ Use the sync wrapper `gensec_update()` if they only act as server only accepting Kerberos
 - ▶ Make use of the fully async `gensec_update_send/recv()` pair.
- ▶ The hardest part was rewriting of `spnego.c`
 - ▶ That module needed 82 patches in order to untangle the logic and make it completely async.
- ▶ `auth_check_password_send/recv()` was rewritten:
 - ▶ To allow backends to optionally provide `check_password_send()/recv()`
 - ▶ Only `source4/auth/ntlm/auth_winbind.c` (used as AD DC) makes use of it (yet).
- ▶ Auth methods in use:
 - ▶ NTLM auth: "anonymous sam winbind sam_ignoredomain"
 - ▶ NETLOGON: "sam winbind"
 - ▶ winbindd: "sam"

Full async authentication stack (Part2)



- ▶ Changing the callers to:
 - ▶ Use the sync wrapper `gensec_update()` if they only act as server only accepting Kerberos
 - ▶ Make use of the fully async `gensec_update_send/recv()` pair.
- ▶ The hardest part was rewriting of `spnego.c`
 - ▶ That module needed 82 patches in order to untangle the logic and make it completely async.
- ▶ `auth_check_password_send/recv()` was rewritten:
 - ▶ To allow backends to optionally provide `check_password_send()/recv()`
 - ▶ Only `source4/auth/ntlm/auth_winbind.c` (used as AD DC) makes use of it (yet).
- ▶ Auth methods in use:
 - ▶ NTLM auth: "anonymous sam winbind sam_ignoredomain"
 - ▶ NETLOGON: "sam winbind"
 - ▶ winbindd: "sam"

Full async authentication stack (Part2)



- ▶ Changing the callers to:
 - ▶ Use the sync wrapper `gensec_update()` if they only act as server only accepting Kerberos
 - ▶ Make use of the fully async `gensec_update_send/recv()` pair.
- ▶ The hardest part was rewriting of `spnego.c`
 - ▶ That module needed 82 patches in order to untangle the logic and make it completely async.
- ▶ `auth_check_password_send/recv()` was rewritten:
 - ▶ To allow backends to optionally provide `check_password_send()/recv()`
 - ▶ Only `source4/auth/ntlm/auth_winbind.c` (used as AD DC) makes use of it (yet).
- ▶ Auth methods in use:
 - ▶ NTLM auth: "anonymous sam winbind sam_ignoredomain"
 - ▶ NETLOGON: "sam winbind"
 - ▶ winbindd: "sam"

Full async authentication stack (Part2)



- ▶ Changing the callers to:
 - ▶ Use the sync wrapper `gensec_update()` if they only act as server only accepting Kerberos
 - ▶ Make use of the fully async `gensec_update_send/recv()` pair.
- ▶ The hardest part was rewriting of `spnego.c`
 - ▶ That module needed 82 patches in order to untangle the logic and make it completely async.
- ▶ `auth_check_password_send/recv()` was rewritten:
 - ▶ To allow backends to optionally provide `check_password_send()/recv()`
 - ▶ Only `source4/auth/ntlm/auth_winbind.c` (used as AD DC) makes use of it (yet).
- ▶ Auth methods in use:
 - ▶ NTLM auth: "anonymous sam winbind sam_ignoredomain"
 - ▶ NETLOGON: "sam winbind"
 - ▶ winbindd: "sam"

Next Steps



- ▶ Disable SAMR and LDAP access as AD DC to trusted domains
- ▶ Make use of S4U2Self in winbindd
- ▶ Kerberos FAST in winbindd
- ▶ LSA LookupSids/LookupsNames
- ▶ Let winbindd use the trust routing table
- ▶ Automatic creation of foreignSecurityPrincipal objects
- ▶ Implement SID expanding/filtering
- ▶ Selective Authentication

Draft !!!

Next Steps



- ▶ Disable SAMR and LDAP access as AD DC to trusted domains
- ▶ Make use of S4U2Self in winbindd
- ▶ Kerberos FAST in winbindd
- ▶ LSA LookupSids/LookupsNames
- ▶ Let winbindd use the trust routing table
- ▶ Automatic creation of foreign SecurityPrincipal objects
- ▶ Implement SID expanding/filtering
- ▶ Selective Authentication

Next Steps



- ▶ Disable SAMR and LDAP access as AD DC to trusted domains
- ▶ Make use of S4U2Self in winbindd
- ▶ Kerberos FAST in winbindd
- ▶ LSA LookupSids/LookupsNames
- ▶ Let winbindd use the trust routing table
- ▶ Automatic creation of foreign SecurityPrincipal objects
- ▶ Implement SID expanding/filtering
- ▶ Selective Authentication

Next Steps



- ▶ Disable SAMR and LDAP access as AD DC to trusted domains
- ▶ Make use of S4U2Self in winbindd
- ▶ Kerberos FAST in winbindd
- ▶ LSA LookupSids/LookupsNames
- ▶ Let winbindd use the trust routing table
- ▶ Automatic creation of ForeignSecurityPrincipal objects
- ▶ Implement SID expanding/filtering
- ▶ Selective Authentication

Next Steps



- ▶ Disable SAMR and LDAP access as AD DC to trusted domains
- ▶ Make use of S4U2Self in winbindd
- ▶ Kerberos FAST in winbindd
- ▶ LSA LookupSids/LookupsNames
- ▶ Let winbindd use the trust routing table
- ▶ Automatic creation of ForeignSecurityPrincipal objects
- ▶ Implement SID expanding/filtering
- ▶ Selective Authentication

Next Steps



- ▶ Disable SAMR and LDAP access as AD DC to trusted domains
- ▶ Make use of S4U2Self in winbindd
- ▶ Kerberos FAST in winbindd
- ▶ LSA LookupSids/LookupsNames
- ▶ Let winbindd use the trust routing table
- ▶ Automatic creation of foreignSecurityPrincipal objects
- ▶ Implement SID expanding/filtering
- ▶ Selective Authentication

Next Steps



- ▶ Disable SAMR and LDAP access as AD DC to trusted domains
- ▶ Make use of S4U2Self in winbindd
- ▶ Kerberos FAST in winbindd
- ▶ LSA LookupSids/LookupsNames
- ▶ Let winbindd use the trust routing table
- ▶ Automatic creation of foreignSecurityPrincipal objects
- ▶ Implement SID expanding/filtering
- ▶ Selective Authentication

Next Steps



- ▶ Disable SAMR and LDAP access as AD DC to trusted domains
- ▶ Make use of S4U2Self in winbindd
- ▶ Kerberos FAST in winbindd
- ▶ LSA LookupSids/LookupsNames
- ▶ Let winbindd use the trust routing table
- ▶ Automatic creation of foreignSecurityPrincipal objects
- ▶ Implement SID expanding/filtering
- ▶ Selective Authentication

Further Authentication Topics



- ▶ Let lower privileged services use kerberos authentication:
 - ▶ Needs explicit PAC verification with a domain controller
 - ▶ Needs a gss-proxy like gensec module
 - ▶ Let winbindd proxy an gss-proxy like interface
- ▶ Kerberos (constrained) delegation (S4U2Proxy)
- ▶ Further hardening
 - ▶ Extended Protection TLS Channel Binding Token CBT
 - ▶ Server SPN target name validation level (SmbServerNameHardeningLevel, UnverifiedTargetName)
- ▶ Public Key Cryptography Based User-to-User Authentication
 - ▶ PKU2U (like Kerberos with PKINIT)
 - ▶ But the target server acts as a KDC over the gss_[init|accept]_sec_context() channel
 - ▶ Will replace NTLM in workgroup kind of setups
- ▶ [Group] Managed Service Accounts

Further Authentication Topics



- ▶ Let lower privileged services use kerberos authentication:
 - ▶ Needs explicit PAC verification with a domain controller
 - ▶ Needs a gss-proxy like gensec module
 - ▶ Let winbindd proxy an gss-proxy like interface
- ▶ Kerberos (constrained) delegation (S4U2Proxy)
- ▶ Further hardening
 - ▶ Extended Protection TLS Channel Binding Token CBT
 - ▶ Server SPN target name validation level (SmbServerNameHardeningLevel, UnverifiedTargetName)
- ▶ Public Key Cryptography Based User-to-User Authentication
 - ▶ PKU2U (like Kerberos with PKINIT)
 - ▶ But the target server acts as a KDC over the gss_[init|accept]_sec_context() channel
 - ▶ Will replace NTLM in workgroup kind of setups
- ▶ [Group] Managed Service Accounts

Further Authentication Topics



- ▶ Let lower privileged services use kerberos authentication:
 - ▶ Needs explicit PAC verification with a domain controller
 - ▶ Needs a gss-proxy like gensec module
 - ▶ Let winbindd proxy an gss-proxy like interface
- ▶ Kerberos (constrained) delegation (S4U2Proxy)
- ▶ Further hardening
 - ▶ Extended Protection TLS Channel Binding Token CBT
 - ▶ Server SPN target name validation level (SmbServerNameHardeningLevel, UnverifiedTargetName)
- ▶ Public Key Cryptography Based User-to-User Authentication
 - ▶ PKU2U (like Kerberos with PKINIT)
 - ▶ But the target server acts as a KDC over the gss_[init|accept]_sec_context() channel
 - ▶ Will replace NTLM in workgroup kind of setups
- ▶ [Group] Managed Service Accounts

Further Authentication Topics



- ▶ Let lower privileged services use kerberos authentication:
 - ▶ Needs explicit PAC verification with a domain controller
 - ▶ Needs a gss-proxy like gensec module
 - ▶ Let winbindd proxy an gss-proxy like interface
- ▶ Kerberos (constrained) delegation (S4U2Proxy)
- ▶ Further hardening
 - ▶ Extended Protection TLS Channel Binding Token CBT
 - ▶ Server SPN target name validation level (SmbServerNameHardeningLevel, UnverifiedTargetName)
- ▶ Public Key Cryptography Based User-to-User Authentication
 - ▶ PKU2U (like Kerberos with PKINIT)
 - ▶ But the target server acts as a KDC over the gss_[init,accept]_sec_context() channel
 - ▶ Will replace NTLM in workgroup kind of setups

▶ [Group] Managed Service Accounts



Further Authentication Topics



- ▶ Let lower privileged services use kerberos authentication:
 - ▶ Needs explicit PAC verification with a domain controller
 - ▶ Needs a gss-proxy like gensec module
 - ▶ Let winbindd proxy an gss-proxy like interface
- ▶ Kerberos (constrained) delegation (S4U2Proxy)
- ▶ Further hardening
 - ▶ Extended Protection TLS Channel Binding Token CBT
 - ▶ Server SPN target name validation level (SmbServerNameHardeningLevel, UnverifiedTargetName)
- ▶ Public Key Cryptography Based User-to-User Authentication
 - ▶ PKU2U (like Kerberos with PKINIT)
 - ▶ But the target server acts as a KDC over the gss_[init,accept]_sec_context() channel
 - ▶ Will replace NTLM in workgroup kind of setups
- ▶ [Group] Managed Service Accounts

Questions?



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

Useful links follow on the next page...

Draft !!!

Useful links (Part1)



- ▶ [MS-APDS] Authentication Protocol Domain Support
- ▶ [MS-AUTHSOD] Authentication Services Protocols Overview
- ▶ [MS-DTYP] Windows Data Types
- ▶ [MS-LSAD] Local Security Authority (Domain Policy) Remote Protocol
- ▶ [MS-LSAT] Local Security Authority (Translation Methods) Remote Protocol
- ▶ [MS-NLMP] NT LAN Manager (NTLM) Authentication Protocol
- ▶ [MS-PAC] Privilege Attribute Certificate Data Structure
- ▶ [MS-WMOD] Windows Management Protocols Overview

- ▶ draft-zhu-pku2u-09
- ▶ draft-zhu-negoex-04



- ▶ [TECHNET: Authentication Policies and Authentication Policy Silos](#)
- ▶ [TECHNET: Changes in Kerberos Authentication \(Windows 2008R2\)](#)
- ▶ [TECHNET: Introducing Forest Search Order \(Windows 2008R2\)](#)
- ▶ [TECHNET: How Domain and Forest Trusts Work](#)
- ▶ [TECHNET: Kerberos Constrained Delegation Overview](#)
- ▶ [TECHNET: Extended Protection for Authentication](#)
- ▶ [TECHNET: Public Key Cryptography based User to User Authentication Overview \(PKU2U\)](#)
- ▶ [TECHNET: Protected Users Security Group](#)
- ▶ [TECHNET: Security Considerations for Trusts](#)
- ▶ [TECHNET: Server SPN target name validation level](#)
- ▶ [TECHNET: Windows Authentication Technical Overview](#)
- ▶ [TECHNET: What's New in Kerberos Authentication \(Windows 2012\)](#)