Life without NTLM

… or how to trust in FIPS

Alexander Bokovoy, Andreas Schneider

Samba Team, Red Hat

Why FIPS?

- Short answer:
  - some customers are required to operate in an environment that is compliant to FIPS set of standards
  - some customers are relying on FIPS requirements to define their operating environment in certain industries
  - PCI-DSS, Common Criteria, and many other standards refer to FIPS set of standards for encryption related requirements

Why FIPS?

- FIPS compliance is always a blend
  - software certification
  - operational environment
  - business processes
  - auditor's point of view
- Samba perspective
  - relies on cryptography primitives provided by an operating system
  - requires certain behavior to comply with protocol specifications
  - needs to be able to work in FIPS environment even with features cut short

FIPS mode

- "FIPS mode"
  - single host mode where only FIPS-approved cryptography primitives are available
  - FIPS certified crypto libraries will fail use of non-approved crypto
  - "Approved crypto" is defined by FIPS-140-2 (and FIPS-140-3) NIST standards

3.1.5.2 Encryption Types

08/24/2020 • 2 minutes to read

KILE SHOULD<23> support the Advanced Encryption Standard (AES) encryption types:

- AES256-CTS-HMAC-SHA1-96 [18] ([RFC3962] section 7)
- AES128-CTS-HMAC-SHA1-96 [17] ([RFC3962] section 7)

and SHOULD<24> support the following encryption types, which are listed in order of relative strength:

- RC4-HMAC [23] [RFC4757]
- RC4-HMAC-EXP [24] [RFC4757]
- DES-CBC-MD5 [3] [RFC3961]
- DES-CBC-CRC [1] [RFC3961]

Kerberos V5 encryption type assigned numbers are specified in [RFC3961] section 8, [RFC4757] section 5, and [RFC3962] section 7. <25>

3.1.5.2 Encryption Types

08/24/2020 • 2 minutes to read

KILE SHOULD<23> support the Advanced Encryption Standard (AES) encryption types:

- AES256-CTS-HMAC-SHA1-96 [18] ([RFC3962] section 7)
- AES128-CTS-HMAC-SHA1-96 [17] ([RFC3962] section 7)

and SHOULD<24> support the following encryption types, which are listed in order of relative strength:

- [illegible]HMAC [23] [RFC4757]
- RC4-H[illegible]C-EXP [24] [RFC[illegible]]
- DES-CBC-M[illegible] [RFC3961]
- DES [illegible]-CRC [1] [RFC[illegible]61]

Kerberos V5 encryption type assigned numbers are specified in [RFC3961] section 8, [RFC4757] section 5, and [RFC3962] section 7.<25>

Protocol fallbacks

- Client and server negotiation
  - allows to find a common supported subset
  - in FIPS mode we cannot fall back to unsupported crypto to establish a session

5.1.2 Preferred Security Providers

02/14/2019 • 2 minutes to read

Implementations can create programming interfaces (3) and corresponding documentation for accessing functionality offered by these extensions in a way that encourages higher-level protocols to not use NTLM as the security provider. SPNEGO and Kerberos offer stronger security.

- In MS-RPCE, use of `RPC_C_AUTHN_GSS_NEGOTIATE` (SPNEGO) would allow downgrade to NTLM

RC4 is everywhere

- Even if NTLM is forbidden and downgrade is not possible,
  - Setting trust requires RC4 to encrypt trusted domain object credentials
    - Covered by MS-LSAD section 3.1.4.7.10 and MS-LSAD section 5.1.1
  - Establishing AES session key between domain member and a domain controller requires RC4-HMAC of a machine account credential in MS-NRPC section 3.1.4.3.1

A way forward

- It is possible to use “unapproved crypto” when it is equivalent to a plain-text in otherwise encrypted channel
  - For example, RC4-HMAC inside a channel wrapped within AES-encrypted session
- This would cover both LSA `CreateTrustedDomainEx2` and NRPC `ServerAuthenticate3`
- We need to ensure no fall back happens to unapproved crypto anywhere else and the session key uses approved crypto (AES)
- We need new DCE RPC calls that do not use RC4 internally
  - BackupKey, NETLOGON, LSA, SAMR, …

Crypto consolidation

- Samba now is out of crypto business for most of its cryptography use
- The migration to GnuTLS started with

```
commit 382d5908a45f7a4a0bb6df98b3b8fa884ed9729a
Author:     Andreas Schneider <asn@samba.org>
AuthorDate: Wed Oct 10 14:20:11 2018 +0200
Commit:     Andrew Bartlett <abartlet@samba.org>
CommitDate: Tue Apr 30 23:18:26 2019 +0000

    waf: Add mandatory requirement for GnuTLS >= 3.2.0

    We plan to move to GnuTLS for crypto in Samba, this is the first step to
    make it mandatory and to require a version which is in LTS
    distributions.
```

Crypto consolidation

- and finished with

```
commit 20b9cae63d5a5881cc6100a2533fab683cc307aa
Author:     Andreas Schneider <asn@samba.org>
AuthorDate: Tue Dec 10 18:06:29 2019 +0100
Commit:     Andreas Schneider <asn@cryptomilk.org>
CommitDate: Tue Dec 10 20:30:57 2019 +0000

    lib:crypto: Build intel aes-ni only if GnuTLS doesn't provide AES CMAC

    Signed-off-by: Andreas Schneider <asn@samba.org>
    Reviewed-by: Andrew Bartlett <abartlet@samba.org>
```

- around 380 commits in total

FIPS mode in Samba

- With the GnuTLS changes landing in 4.12, we can run in FIPS mode
  - however, NTLM is still advertized to clients (fixed in 4.13)
  - Kerberos is still not used by default (fixed, will be in 4.15)
  - Things abort or give strange errors

Samba and FreeIPA in FIPS mode

- FreeIPA uses Samba Python bindings to establish trust
- Samba DCE RPC code now ensures secure session is established before LSA calls with RC4 content are used
- Samba Python API exposes `samba.trust_utils.CreateTrustedDomainRelax` method
  - accepts raw trust domain object credentials
  - performs RC4 processing
  - wraps into AES session key
  - only communicates TDO creds to a DC to be trusted when the session key is strong enough

Samba FIPS mode = use of Kerberos

- In FIPS mode Samba can only use Kerberos tickets with AES encryption types
- … obtained with Kerberos pre-authentication methods approved by your FIPS auditor
- `client use kerberos = desired |required | off`
- More effort was needed to apply correct defaults everywhere
  - command line updates to use Kerberos, unify CLI behavior, etc
  - 650+ commits in total to enable FIPS support

The following Samba modes and features work in FIPS mode under the indicated conditions:

- Samba as a domain member only in Active Directory or FreeIPA environments with Kerberos authentication that uses AES ciphers.
- Samba as a file server on an Active Directory domain member. However, this requires that clients use Kerberos to authenticate to the server.

Due to the increased security of FIPS, the following Samba features and modes do not work if FIPS mode is enabled:

- NT LAN Manager (NTLM) authentication because RC4 ciphers are blocked
- The server message block version 1 (SMB1) protocol
- The stand-alone file server mode because it uses NTLM authentication
- NT4-style domain controllers
- NT4-style domain members. Note that Red Hat continues supporting the primary domain controller (PDC) functionality FreeIPA uses in the background.
- Password changes against the Samba server. You can only perform password changes using Kerberos against an Active Directory domain controller.

The following feature is not tested in FIPS mode and, therefore, is not supported by Red Hat:

- Running Samba as a print server

Future

- How to support non-enrolled clients?

Future

- How to support non-enrolled clients?
- We need a replacement for NTLM

Future

- How to support non-enrolled clients?
- We need a replacement for NTLM
  - MS-AUTHSOD 2.1.2.2 allows use of NTLM, Kerberos, PKU2U and a custom authentication protocol through NegoEX

Future

- How to support non-enrolled clients?
- We need a replacement for NTLM
  - MS-AUTHSOD 2.1.2.2 allows use of NTLM, Kerberos, PKU2U and a custom authentication protocol through NegoEX
- PKU2U is not supported by Samba

Future

- How to support non-enrolled clients?
- We need a replacement for NTLM
  - MS-AUTHSOD 2.1.2.2 allows use of NTLM, Kerberos, PKU2U and a custom authentication protocol through NegoEX
- PKU2U is not supported by Samba
- SPNEGO NegoEX

Future

- How to support non-enrolled clients?
- We need a replacement for NTLM
  - MS-AUTHSOD 2.1.2.2 allows use of NTLM, Kerberos, PKU2U and a custom authentication protocol through NegoEX
- PKU2U is not supported by Samba
- SPNEGO NegoEX
  - as a container, can allow new authentication methods without changing the protocol

Future

- How to support non-enrolled clients?
- We need a replacement for NTLM
  - MS-AUTHSOD 2.1.2.2 allows use of NTLM, Kerberos, PKU2U and a custom authentication protocol through NegoEX
- PKU2U is not supported by Samba
- SPNEGO NegoEX
  - as a container, can allow new authentication methods without changing the protocol
  - supported by Microsoft Windows, MIT Kerberos and Heimdal Kerberos

Future

- How to support non-enrolled clients?
- We need a replacement for NTLM
  - MS-AUTHSOD 2.1.2.2 allows use of NTLM, Kerberos, PKU2U and a custom authentication protocol through NegoEX
- PKU2U is not supported by Samba
- SPNEGO NegoEX
  - as a container, can allow new authentication methods without changing the protocol
  - supported by Microsoft Windows, MIT Kerberos and Heimdal Kerberos
  - needs implementation of the actual authentication method

Future

- How to support non-enrolled clients?
- We need a replacement for NTLM
  - MS-AUTHSOD 2.1.2.2 allows use of NTLM, Kerberos, PKU2U and a custom authentication protocol through NegoEX
- PKU2U is not supported by Samba
- SPNEGO NegoEX
  - as a container, can allow new authentication methods without changing the protocol
  - supported by Microsoft Windows, MIT Kerberos and Heimdal Kerberos
  - needs implementation of the actual authentication method
- For example, Password Authenticated Key Exchange (PAKE) implementation

Future

- How to support non-enrolled clients?
- We need a replacement for NTLM
  - MS-AUTHSOD 2.1.2.2 allows use of NTLM, Kerberos, PKU2U and a custom authentication protocol through NegoEX
- PKU2U is not supported by Samba
- SPNEGO NegoEX
  - as a container, can allow new authentication methods without changing the protocol
  - supported by Microsoft Windows, MIT Kerberos and Heimdal Kerberos
  - needs implementation of the actual authentication method
- For example, Password Authenticated Key Exchange (PAKE) implementation
  - Kerberos has SPAKE pre-authentication, implemented in MIT Kerberos and deployed already for several years

Thanks!