Samba 2020: Why are we stuck in the 1980’s

Andrew Bartlett
Samba in 2020

A status update
Samba 4.11 released

First Samba AD released for the 300,000 user scale
GnuTLS used for cryptography (new to the fileserver)
SMB1 Disabled by default
LanMan and plaintext authentication deprecated
Python 3.4 required at runtime
Python 2.7 still supported for the build (only)
CI tested on OpenSUSE, Fedora, RHEL, Debian, Ubuntu
Samba 4.12 frozen

More in-tree cryptography replaced (DES, AES)
DES Kerberos keys no longer supported
More robust NDR parser due to fuzz testing
Python 3.5 required at runtime
Python 2.7 still supported for the build (only)
1980s authentication?

Just say no to NTLM?
LanMan Authentication

SMB started with DES back in 1982
Challenge/response was a great improvement at the time
DES(chal, uppercase(password[:7])).DES(chal, uppercase(password[7:]))
Attack each end of the password separately
Needed for DOS and Win9X client
To be clear, LanMan authtication is disabled by default
NTLM authentication

Unicode password support added for Windows NT in 1990s

Still quite weak:

DES(chal, MD4(UTF-16LE(password))[7:7]).
DES(chal, MD4(UTF-16LE(password))[7:14]).
DES(chal, MD4(UTF-16LE(password))[14:].[0][0][0][0][0][0][0])

Those zeros and the splitting is not good!

Still used extensively by MS-CHAPv2

Typically over TLS, but did you check the certificate?
NTLMv2 authentication

HMAC-MD5 based
Primary risk is offline brute force attacks
Can include provision for channel binding (for use under TLS)
TLS is not used in SMB and Samba doesn’t implement this so far
The default for all modern clients and Servers
Still requires storage of unsalted MD4-based password hash

Even HMAC-MD5 is a 1997 specification
Kerberos to the rescue
Yet another 1980s protocol

Kerberos is old

Kerberos first deployed in 1986!

Kerberos v5 is 1993

Suffers from a number of 1980s design decisions (ASN.1)

The solution to every problem

Except for the complexity of Kerberos
Kerberos should solve the issues

De-couples login (getting a ticket) from submitting that ticket
Provides Smart card support
So why so little innovation / nothing better?

- No client UI control (unlike the web)
- Hashed password methods (so poor OTP options)
- Kerberos is a poor match without a full domain
- Smart cards only work as part of a domain
- Clients speak Kerberos or NTLM, but nothing else
KDC on every device?

Apple has a mode where Kerberos, not NTLM is used in a workgroup

The fileserver is a KDC for itself only

Avoids the need to find the KDC

No support on Windows or Samba clients

Essentially a way of using the Kerberos key exchange in place of NTLM
Little interest from Microsoft

A better NTLM is not considered a priority
Focus is on Windows Hello and Hello for Business
Windows Hello
So what is this Hello thing anyway

Log into your windows computer directly to the cloud
Competing with Chromebooks and apple cloud-based login
Unlock the device with a per-device PIN
Hard to actually set up local accounts on new windows installs!
Hello for business

Essentially makes your PC the smart card
Unlock your PC and you have unlocked your smart card
Enrolment procedure into AD via Active Directory Federation Services

So not an easy add-on for Samba
What can we do?
Smart Cards could be easier

Smart cards (eg the Yubikey 5) are still a pain to set up
Typical enrolment involves a full CA infrastructure
Samba supports this but certificate revocation support is not great
Alternative is to record each key in the directory entry for the user
But not supported in Samba / Heimdal yet.
SSH Wrapping?

SSH keys have become the standard way to authenticate on Linux. Could we somehow forward over SSH and inherit the authentication? Or add the SSHv2 protocol as an additional mechanism in SPNEGO (eg using libssh).
NT hash-free Samba AD?

Should we just remove all MD4 hashes?

We could go pure kerberos, no fallback!

Quite a few bits of Samba’s protocols use the MD4 hash

Password history

Password change over SAMR

Plaintext passwords internally

These need to be re-implemented in terms of eg crypt()
A new or safer NTLM for Samba?

Difficult to negotiate new NTLM versions
Could allow NTLMv2 but not store the raw NT hash?
(salt it with username/domain)
Would require that the “domain” part be given correctly to Samba
Perhaps support some kind of predictable OTP or hardware response?
Something that can cope with being hashed like TOTP
Perhaps put a 2nd factor in the NTLMv2 response, encrypt with password?
What about U2F (eg Yubikey)?

Can Samba somehow make the jump into web security?
U2F is the only physical two-factor system that is simple to set up
Could we give Samba ‘API Keys’ as passwords, configured from the web?
Can we somehow do U2F without the web, like pam-u2f?
Are we headed to a pure-web world?

Does this matter anyway?

Our protocols are on-LAN and the real threats on the big bad web right?

The SAML gateway / IDP can handle all this proper security stuff?

Leave Samba to just check simple passwords?

I think we should do better, but how is not entirely clear

A recognised paring with Samba filling the ADFS niche would be a good start
So what should we do?

We need to innovate, not just follow

abartlet@catalyst.net.nz
abartlet@samba.org
https://samba.org/~abartlet
https://catalyst.net.nz/services/samba