Status Report

• Publication status: The first RPC-with-TLS specification was published as RFC 9289 in September of 2022

• Implementation status:
  • FreeBSD client and server
  • Java-based client and server (DESY)
  • Hammerspace server
  • Linux client and server prototype
  • nginx module
Implementation Experience

- Linux kernel security community has insisted on the use of an out-of-kernel handshake implementation (via an upcall)

- Minimize kernel attack surface by delegating handshake to code running in a context with lesser privilege

- Use an actively-maintained TLS implementation rather than yet another new one

- However, TLS library APIs are quite rich; we implement only a bare few actual operations and features to keep the upcall protocol simple
Linux NFS Client Implementation

• Upcall TLS handshake mechanism nearing completion
  • Shared infrastructure with NVMe/TCP and possibly in-kernel QUIC
  • Kernel passes connected socket descriptor to a user space agent, which uses a standard TLS library to perform the handshake
• `xprtsec= none | tls | mtls` mount option
• Currently supports both server-only and mutual authentication
  • x.509 only at the moment; PSK coming later
Linux NFS Server Implementation

• Uses the same upcall TLS handshake mechanism as the client

• Currently supports only opportunistic TLS
  • If client requests TLS, server uses it, but cannot yet require encryption or peer authentication

• `xprtsec= <mode> : <mode> : <mode>` export option is planned
  • `<mode>` is a keyword where `none` means the export is accessible without TLS; `tls` means the export is accessible with TLS encryption-only; `mtls` means the export is accessible with TLS encryption plus peer authentication
Linux Prototype Source Code

- Kernel component:

- User TLS handshake agent:
  - https://github.com/oracle/ktls-utils

- Coming soon: nfs-utils with TLS mount and export options and man page updates