OPAQUE

draft-irtf-cfrg-opaque

Krawczyk, Lewi, Wood — IETF 110 — CFRG

OPAQUE is a compiler for translating an OPRF, hash function, memory hard function (MHF), and authenticated key exchange (AKE) protocol into a **strong, augmented PAKE**

OPAQUE

Overview

Two protocol phases:

- Offline registration: Clients use password to register public key credentials with the server
- Online login: Clients use their password to recover public key credentials from the server and complete an AKE

This document specifies **OPAQUE-3DH** with accommodations for future AKE instantiations (TLS 1.3, SIGMA-I/R, HMQV, etc.)

Updates
Draft status

Major:

- Massive document refactor
- Generalized cryptographic dependencies (OPRF, KDF, MAC, Hash, MHF, Group)
- Fixed-length wire format where applicable

Minor:

- 3DH transcript simplification and alignment with TLS 1.3
- Suggested password file serialization
- Added test vectors

Updates

Implementation status

Multiple interoperable implementations exist:

- Reference (Sage)
- opaque-ke (Rust)
- opaque (Go)

Other implementations underway!

Open Issues

Private key storage (#84)

Private keys are currently derived externally to OPAQUE and stored in encrypted credential files on the server

Clients with knowledge of username and password can interact with the server, ***compute the OPRF output***, and decrypt the credential file

Proposal: when applicable, use OPRF output to deterministically derive keys *internally*, else allow applications to provide *external* keys

Summary:

- Internal mode has no external application dependency and no credential file storage overhead
- External mode allows applications with existing keys (or reliable key generation code) to reuse it when appropriate

Open Issues

Client enumeration (#22)

Server response during registration changes if a user is registered or not

Active adversaries can use this signal to learn whether or not a user is registered for a given server (security regression compared to “password-over-TLS”)

Two options:

1. Specify optional server-side behavior that prevents an active attacker from distinguishing “does not exist” from “exists,” but does allow attacker to learn client credentials changed

2. Change protocol such that attackers cannot learn *any* useful information based on server response

Proposal: option (2)

Next steps

Steps to RGLC

Resolve #22 and #84 (accept both proposals)

Update test vectors and implementations

Ready to ship

Questions?