Attested TLS Token Binding

Giri Mandyam

draft-mandyam-tokbind-attest
Introduction

• Identity federation systems often use bearer tokens for client verification
  – 3rd parties can validate client when receiving token from trusted identity provider
  – Tokens can come in many forms (JSON web tokens, cookies)
• Bearer tokens that are extracted from the client device can be used to impersonate the end user
• Problem can also occur when encrypted connection such as TLS is subject to MITM
  – Attacker extracts token
• As a result, the IETF is standardizing token binding for TLS
  – https://tools.ietf.org/wg/tokbind/
Introduction (cont.)

• Bearer tokens are still applicable, but client must prove possession of a private key on every TLS connection to a server

• Current specification requires signing of payload that includes
  – Exported Key Material (RFC 5705)
  – Tokbind.type and Tokbind.KeyParameters

• User agent (browser) could maintain private keys associated with TLS token binding
  – Problem: User agents are usually implemented in user space; private keys may be vulnerable
    • Attacker that obtains private key and bearer token can impersonate client
  – Problem not much better for native applications
    • Many OS’s use open source libraries such as OpenSSL to implement secure socket connection
      – Private keys may still be stored in user space
Hardware-Secured Signing for TLS Token Binding

• Definition, “signing process” – any application or platform functionality that can execute crypto operations such as signing
  – “HW-bound” or “HW-secured” signing process: process runs in the context of a root-of-trust

• Many existing HW-bound signing processes protect private keys in trusted environments (trusted execution environment - TEE, secure element, TPM)
  – Examples include HW-secured authenticators (e.g. authenticators running in TEE)

• Such processes can be used for generating the signature for token binding

• Relying parties can make decisions as to whether to continue TLS session with clients based on storage of private keys
Remote Attestation

• Describes the process by which software executing on a device provides an assertion to a relying party about the integrity of its platform
  – The platform in question is the one controlling the tokbind private key
• The attestation can be based on several criteria, including ‘health’ measurements of platform
  – An assessment of the operating system kernel
  – Enumeration of 3rd-party applications installed in environment where credential is stored
  – Suspicious events such as protected memory access
• Attestation data is formed by combining these indications into a compact data structure that can be sent to a relying party
  – Attestation data is used to form an attestation statement, which is the actual message sent to the relying party
  – Attestation statement should be cryptographically-verifiable (signed and/or encrypted)
Tokbind Impact

• Inclusion of an attestation in the tokbind message enabled through an extension
  – Sec. 3.4, draft-ietf-tokbind-protocol: “One of the possible uses of extensions envisioned at the time of this writing is attestation …”

• Current I.-D. proposes a pairing of attestation type and attestation data for extension

• Types can be extracted from sources such as
  – TCG – TPM v1.2 specifications
  – Pre-registered attestation formats in W3C Web Authentication API spec
    • https://www.w3.org/TR/webauthn/#defined-attestation-formats
Recommendations

• Tokbind WG adopt this draft
  – Can be informational

• Determine initial attestation formats for Tokbind
  – While still allowing for extensibility