Insights for Edge Software Developers

Presented by: Michaela Vanderveen
5G Principal Security Architect
MITRE Corporation
ETSI ISG MEC delegate

For: everyone

Episode #14 – MEC API security
In this episode ...

• We will learn:
  • The difference between authentication, authorization, encryption and integrity
  • Basics of security protocols: TLS and OAuth2.0
  • How to avoid common pitfalls in implementation of such protocols
Difference between authentication and authorization

**Authentication**
- 1. Verify **identity**
- 2. Set up secure link based on it (encryption, integrity)
  Protocol example: TLS 1.3
  HTTP over TLS := HTTPS

- “All RESTful MEC service APIs shall support HTTP over TLS...”
  -- MEC 009

**Authorization**
- 1. Obtain **permission** (token)
- 2. Get resources based on it
  Protocol example: OAuth 2.0

- “This API shall use OAuth 2.0, .. only on TLS-protected connections”
  Authentication and Authorization (AA) entity assumed.
  -- MEC 009
Digital Certificate hygiene

A certificate: digital blob to prove your identity (client, device, ...)

A **valid** certificate means that the owner is considered **trustworthy**. A certificate authority that issued it vouches for it.

An **expired** certificate means that the owner is not trustworthy, even if they were at some point in the past. All TLS connections should be **dropped** once that expiration time is reached.

If an entity owning a certificate is found to cease to be trustworthy, then the issuing certificate authority may **revoke** that certificate→ this means that as part of certificate checking, a verifier should also check the appropriate CRL (cert revocation list)

**TLS connections** should be set up only if the other party proves it has a **valid** certificate. To verify validity, several fields in the cert should be checked: SubjectName, Expiry Time.

Most widely used certificate standard is X.509.
The version matters!!

TLS mutually **authenticates** two parties and **sets up a secure tunnel**

TLS 1.3 is **FASTER** and **MORE SECURE**

TLS 1.3 is not backward compatible.

Attacks on TLS 1.2:

DROWN, CRIME, [zombie/golden]POODLE, SLOTH, FREAK,...

Picture source: https://medium.com/quick-code/the-dangers-of-hacking-and-what-a-hacker-can-do-to-your-computer-38d3f638a95
The ciphersuites matter!!

Several algorithms form a **TLS “Ciphersuite”**

Ciphersuites are defined differently for TLS 1.2 and TLS 1.3

Some algorithms have weaknesses. Others are considered sufficiently secure “now”.

**TLS 1.3 only allows secure ciphersuites**

See IANA TLS Parameters or 3GPP TS 33.210 for ciphersuites selection recommendations.
OAuth 2.0: The way to authorize access to your API

**RFC 6749**

- A. Authorization Request
- B. Authorization Grant
- C. Authorization Grant
- D. Access Token
- E. Access Token
- F. Protected Resource

**ETSI MEC 009**

- Access token request
- Access token response
- HTTP Req with token
- HTTP response

Check that access token!

Tokens can be of type: bearer, certificate bound, ...
Tokens can be refreshed, or revoked
Protect your apps

Pitfalls to watch for and tips when applying the OAuth 2.0 framework to MEC:

→ Compromise of credentials -- they underpin security: Provision credentials securely into the REST client and AA entity (e.g., upload certificates, ensure can renew/revoke)

→ Always set up a TLS tunnel first between REST client and any server (AA or REST server)

→ Threats due to lack of cross-layer checking between TLS and OAuth
  → Check client... At OAuth time, REST server ("resource server") should have a way to check the REST client ("client") credentials match those used for the TLS pipe underneath

→ OAuth bearer tokens can be stolen. Treat your OAuth tokens like passwords

→ (server-side) Prevent over-privileged access: check token scope before granting access to resource
Conclusions and further resources

What we have learnt:

• The difference between authentication and authorization
• TLS 1.3 basics and OAuth 2.0 basics
• Common pitfalls in implementation

Interested to learn more?

• OWASP: Transport Layer Protection Cheat Sheet: Transport Layer Protection - OWASP Cheat Sheet Series
• Follow also the next episodes of the MEC TECH Series 😊
Enjoy the