Securing IS-04/05 - How to Lock My Media Streams

Arne Bönninghoff – Head of IP Research
Riedel Communications GmbH & Co. KG

Problem?

• IS-04/05 are open specifications
Problem?

- IS-04/05 are open specification

➢ Everyone that implements IS-04/05, could stop all videofeeds in this Showcase
Solution

• Secure Access to API

Solution

• Allow execution of Actions only to **authorized** controllers
Authorisation vs. Authentication

• Authentication:
  – verify that someone is who they claim to be
    ➢ Covered by exchange of certificates

• Authorization:
  – deciding which resource a user should be able to access, and what they should be allowed to do with those resources
    ➢ Additional techniques needed

How to become authorized?

Oauth2 and JWT (JSON Web Token)
Cookie vs. Token

Cookie based auth

- Stateful
  - Client + Server must keep record of active session
- Traditional cookie flow:
  1. User enters their login credentials
  2. Server verifies, creates a session which is stored in DB
  3. A cookie with session ID is placed in the users browser
  4. On subsequent requests, the session ID is verified against the database and if valid the request will be processed
  5. Once a user logs out, the session is destroyed on both sides
Token based auth

- Stateless
  - No record on Server about a session
- Traditional Token flow:
  1. User enters their login credentials
  2. Server verifies the credentials are correct and returns signed token
  3. Token is stored client-side (most common in local storage, but cookie is possible as well)
  4. Subsequent requests to the server include this token as an additional Authorization header
  5. Server decodes the token and if token is valid process the request
  6. Once a user logs out, the token is destroyed client-side. No interaction with server is needed.

Token > cookie

- Statelessness
- Self-containing information
- Performance
  - Cookie:
    - Lookup against DB if session is valid
    - Does user have access to the requested resource
  - Token:
    - Token valid?
    - Get additional data out of the token
JWT

- Header, Body, Signature
- Body containing claims
- Rfc7519

JWT Workflow
How to exchange JWT?

- OAuth2
- Authorization framework
- Enables applications to obtain limited access to user accounts on an HTTP service
- Delegates user authentication to the service that hosts the user account
- Authorizing third-party applications to access the user account
- OAuth2 provides authorization flows for web and desktop applications, and mobile devices
- RFC 6749
- Introspection: RFC 7662

Prototype using JWT & proposed OAuth2 workflow
Prototype using JWT & proposed OAuth workflow

- Locked devices can only be controlled by authorised application

OAuth2 + JWT for NMOS
OAuth2 + JWT for NMOS

- Authorisation server visible through DNS-SD (both unicast and multicast)
- Backwards compatible
  - Just send IS-04 and IS-05 without token
- Also applicable for IS-04 query API
  - Example: only some controllers are allowed to retrieve information
JWT Claims

- Define more granular claims
  - “control:transceivers”
  - “control:senders:audio”

JWT Lifetime & Transport

- Tokens need to be reissued before a timeout
  - Ability to revoke authorisation to controllers
- Tokens are readable in plaintext
  - HTTPS transport mandatory, to prevent unapproved access
Conclusion

• IS-04 and IS-05 are based on standard IT technology
• HTTP and JSON are used by many other applications
• Other applications use OAuth2 and JWT already in large scale
• Key exchange workflow provides the fundamental environment for HTTPS transport
• Secure Transport enables OAuth2
• Backwards compatibility given

Next steps

• Finish public proposal
  – Define grant types for different applications
  – Define workflows together with Key Exchange and an authorisation server
  – Define Token Lifetime
• Test interoperability
• Test backwards compatibility
• Get involved!
  – https://github.com/AMWA-TV/nmos-api-security
Thank You

Arne Bönninghoff, Riedel Communications GmbH
arine.boenninghoff@riedel.net // +49 177 8347500