Stateless Approach to End-to-End Security for the Internet of Things
(OSCAR – Object Security Architecture for the IoT)

Mališa Vučinić ★, Bernard Tourancheau ❖, Franck Rousseau ❖, Andrzej Duda ❖, Laurent Damon ★, and Roberto Guizzetti ★.

❖ Grenoble Informatics Laboratory, France
★ STMicroelectronics

Montbonnot, November 6th 2014
Security in the traditional Internet (1/2)

Alice (D)TLS client

Bob (D)TLS server

Authenticated Secure Channel established once handshake completes

Bidirectional Information Flow

Bob

RSA and AES
Key: 0x9c6d3

Alice

RSA and AES
Key: 0x9c6d3
Security in the traditional Internet (2/2)

Alice
- RSA and AES
  Key: 0x9c6d3

Erin
- ECC and AES
  Key: 0x1e8c2

Carol
- ECC and AES
  Key: 0xdf71e

Wendy
- RSA and AES
  Key: 0x1f61a

Bob
Security in the Internet of Things

Alice
Erin
Carol
Wendy

Proxy Server

Bob

$P$

greenNet
Security in the Internet of Things

What happens to the End-to-End Security?
Motivation

• Features of the Constrained Application Protocol (CoAP) when secured by DTLS:

  • Group communication i.e. multicast support
  • Asynchronous message exchanges
  • Proxy and caching capabilities
  • Low overhead
  • Header mapping to HTTP
OSCAR – concepts behind (1/2)
Object Security Architecture for the Internet of Things [1]

• **Idea 1:** A stateless security architecture
  • Allows caching, eases group communication and asynchronous exchanges
  • Solution: Object security – Application data encapsulated within “secured objects”

  - Object Type: Signed
    - Origin: Temperature sensor in room D326
    - Key ID: 0xf61c
    - Signature: 0xc2779b20ea27a
    - Temperature reading: 25.5 °C
    - Timestamp: 10:42 AM

  - Object Type: Encrypted
    - Key ID: 0xfa1c

• Protect from communication-related attacks by binding object-security encryption keys with the underlying CoAP header

• **Idea 2**: Move the burden of security handshake away from sensors
  - Introduce a *semi*-trusted, non-constrained third party that will do the hard work
  - Sensors respond with secured objects (resource representations) regardless of the identity of the client

• **Idea 2:** Move the burden of security handshake away from sensors
  • Introduce a semi-trusted, non-constrained third party that will do the hard work
  • Sensors respond with secured objects (resource representations) regardless of the identity of the client

• **Idea 3:** Jointly approach problems of End-to-End security and Authorization
  • Split confidentiality and authenticity trust domains
  • Confidentiality used to provide access-control for group members
  • Authenticity strongly tied to the originator of the information (individual sensor)

OSCAR – dive deep (2/2)

Authorization

Server

Client

GET /Temp

Resource representation pre-signed with P's private key

On-the-fly symmetric encryption with key derived from access-secret

Producer P

Temp

Humidity

CO

Location

.well-known/core

Access-secret A

Access-secret B

Access-secret N
CoAP + OSCAR features:

- Group communication i.e. multicast support ✓
- Asynchronous message exchanges ✓
- Proxy and caching capabilities ✓
- Low overhead ±
- Header mapping to HTTP ✓
- End-to-End Security ✓
- Authorization and Access Control ✓
Sensor-side Total Energy Consumption

![Sensor-side Total Energy Consumption](image)

**WiSMote (MSP430)**

- DTLS-Lithe
- OSCAR $\beta = 30s$
- OSCAR $\beta = 60s$
- OSCAR $\beta = 120s$

**ST GreenNet (ARM Cortex M3)**

- DTLS-Lithe
- OSCAR $\beta = 30s$
- OSCAR $\beta = 60s$
- OSCAR $\beta = 120s$

Lower is better

OSCAR performs better as a client # >> slot #
Conclusions & Future Work

- E2E security and authorization framework that supports application requirements
- E2E security even in presence of application-level gateways
- Particularly useful for use-cases where high number of clients per sensor is expected
  - Smart city a very good example
- Future extensions
  - Use-cases that require streaming where constant digital signing is unfeasible
  - Key management and authorization policies
Hvala!* 
Questions?

*Thanks!