Intro to OMA GotAPI
Open-Source Implementation
Supporting Web - Healthcare Use Cases

The information in this presentation is public.
What is GotAPI?

Generic Open Terminal API Framework
- A published specification of the Open Mobile Alliance (OMA)
- A design pattern and implementation framework enabling user-mediated web app access to arbitrary local or connected device APIs

GotAPI Open Source Project
- NTT Docomo’s DeviceConnect (see supported devices)
- GotAPI open source project, launched in October 2014
- Various devices supported
- Enables device vendors to reduce costs and time to develop applications interworking with smartphones

Device WebAPI Consortium
- An industry initiative to promote device Web APIs based on OMA GotAPI
- Launched in April 2015
Overview of GotAPI Design

- Device-local web server ("GotAPI Server") acts as bridge to API plugins
- Web apps discover GotAPI support and get approved access to APIs
- Web apps use standard web APIs (XHR, SSE, WebSockets, WebRTC) to access APIs thru the GotAPI Server or directly (if supported by plugin)
- Plugins provide access to device-local APIs/resources or connected devices, as applicable for the plugin
DeviceConnect Implementation

Applications to access external devices through device APIs using Web technologies

Plug-ins to absorb differences of underlying protocols and specifications, enabling apps to work all across with them
**Architecture**

- A “GotAPI app” implements servers for API access auth and operations
- Developers create plugins that integrate with GotAPI thru platform-specific methods
- Devices that can be accessed are only limited to the types of connections supported by the GotAPI host device, and the openness of connected device interfaces
Basic Flow

• App seeks and gains access to GotAPI-enabled services
• App discovers available APIs
• App requests and is allowed access to specific API
• App accesses the APIs thru the GotAPI Server or directly

* Actual flows add more security features on top of these simplified flows. Actual GotAPI adds security features on top of this flow.
Specific Example

GotAPI app
- GotAPI auth server
- GotAPI server

plug-in app
- plug-in

apps
- web apps (in browser)
- native/hybrid apps

device

vibrate smart watch twice

Pass-through: GotAPI server transfers requests from app to Plug-In via intents (Android)

app is accessing APIs in the plug-in through pass-through mechanism, and plug-in execute the instructions from the app

web API

program in app
- access web API in plug-in using HTTP REST

response in JSON
- Coming from Plug-In passing-through GotAPI server

vibration twice success

bluetooth

wifi

HTTP

Intents

pass-through

{ "result": 0 }

{ "result": 0 }

smart watch

vibration

access token-1

access token-2
Focus on Security

• **GotAPI contains specific measures to address**
  - Application registration and authenticity
  - Application spoofing by rogue apps
  - Privacy (user-mediated access to plug-ins and devices)
  - GotAPI Server spoofing by rogue apps
  - Plugin spoofing by rogue plugin
  - DOS attacks

• **GotAPI assumes**
  - Host device is not rooted
  - Apps are obtained from legitimate sources (App Stores, web servers)
Fujitsu Implementation demo
Fujitsu Implementation demo
Next Steps for GotAPI

- GotAPI 1.1 is complete specifications for WebSocket based APIs
- Device WebAPI 1.0 is developing specific plugin APIs for
  - Healthcare devices based upon published and implemented IEEE specs
    - Thermometer
    - Pulse Oximeter
    - Weight Scale and Body Composition Analyzer
    - Blood Pressure Monitor
    - Glucometer
    - Heart Rate Monitor
- 3D Printers
Links – Draft TS Specification

- [http://member.openmobilealliance.org/ftp/Public_documents/CD/DWAPI/Permanent_documents/OMA-TS-Weight_Scale_Body_Composition_Analyzer_APIs-V1_0-20151104-D.zip](http://member.openmobilealliance.org/ftp/Public_documents/CD/DWAPI/Permanent_documents/OMA-TS-Weight_Scale_Body_Composition_Analyzer_APIs-V1_0-20151104-D.zip)


