IoT SECURITY and MESH NETWORKS

Christoffer Jerkeby
Ericsson Platform Security Research 2015
About me

› Works at Ericsson Platform Security Research
› Active in Global Platform
  – In Trusted Execution environment specification working group
    › A secure environment for devices with high security requirements that can host Applications and Certificates
› Active in Bluetooth Special Interest Group (SIG)
  – In Bluetooth Smart Mesh
    › An IoT carrier based on Mesh distribution of nodes with Bluetooth as a carrier
› Researching security vulnerabilities in network protocols and implementations.
How things used to be
Valdemar Atterdag
Walled garden

› One key to the kingdom
› Just like WPA2, WPA or WEP
› But there are larger gardens than Wi-Fi
› Nation wide and global communication networks
  – SS7, Blue-ray & DVD,
› A multitude of IoT solutions
› Secret keys built in to devices leaks
  – Let me show you some ways!
Hardware attacks

› UARTS
  – Command prompts

› JTAGS
  – Binary extraction

› MMC’s
  – Editable boot loaders

› Targets are keys in all things
  – Modems, VoIP boxes, Lamps, Locks, TV’s etc.
Remote attacks

› Evil-grading
  – Apps
  – Firmware
  – Libraries

› Web interfaces
  – False sense of firewalling
  – Cross site scripting (XSS)
  – Browser attacks
  – Default passwords

› Radio snooping
  – Wi-Fi WPS/WPA(2)/WEP cracking
  – Bluetooth Low Energy 4.0 Cracking
Rules of IoT security

1. Only store unique asymmetric keys on devices
   1. Public Private keys
   2. Negotiate for temporary symmetric session keys

2. Only break the first rule of IoT security when there are no other options but do apply these rules:
   1. Only use one symmetric key per node
      1. Do not share one symmetric authentication key!
   2. Only use one symmetric key once
      1. For bootstrapping.
   3. Use secure storage like a Trusted Execution Environment - Secure Storage or Secure Element (SIM)
Mesh networks
TRUST in Mesh networks

› Distributed nodes
› Multiple relays to reach a destination
› Who to trust?
   – The Walled Garden?
   – Public Key Infrastructures?
     › Centralized on remote and vulnerable Certificate Authorities
   – Secure Device Management
     › Key infrastructure
     › Address management
     › Software management
     › Device Identity management
   – Multiple authorities based on real device owners
     › Persons can have a web of trust
Autonomous domains

› We want to create domains where the user is the authority
› We want to minimize the damage to the domain caused if one node is compromised
› We want to keep the node keys available in the domain regardless of the availability of one given node
  – No single point of failure
› We want the domain to be accessible regardless of the carrier
  – Type, latency, power profile and sleepiness
Distributed Authority

› A method for distributing authentication, authorization and message integrity verification on a network of nodes
› Can maintain multiple points of authority
› Can verify other nodes participation in a domain
› Can relay packets in a domain
   – authenticating the packet origin and integrity
› Can establish end to end encryption between two nodes
   – Using Datagram Transport Layer Security (DTLS 1.2)
Distributed Hash Tables

rfc7363

› A well known method to share information on multiple nodes
  – Invented in relation to file sharing protocols

› We are using it to
  – Distribute management knowledge
    › Key management
    › Address translation
    › Service announcement
  – Use redundancy to prevent single point of failure
  – Synchronize changes in management
Distributed Hash Tables

› All nodes in DHT has a routing table of known good nodes.
  – A **good** node has given an accurate reply within a given time period
  – A node is also **good** if it ever has replied and has sent us a query in the minimum timespan
  – After timespan nodes become **questionable**.
  – A node that fails to reply to multiple queries becomes **bad**.

› **ping**
  – Ask for a node’s availability

› **find_node**
  – Ask for a target node and get node ID back or the 8 closest good nodes (known as a **bucket of nodes**) in reply

› **announce_peer** (could be put_data)
  – Announce a service or data available from a given peer

› **get_peers** (could be get_data)
  – Ask for nodes that have access to a given data or data type.
Addressing

› Use commonly used addressing schemes
  – IPv6 (RFC2460)

› Derive remote node addresses from public keys
  – Cryptographically Generated Addresses (CGA RFC3972)

› Can be routed
  – Ad hoc On-Demand Distance Vector Routing (AODV RFC3561)
Unboxing a THING

– Locate QR Code (or pin) and scan it with the Manager Device
– The QR Code is a representation of the Devices pre-programmed Personal Public Key
– The Devices public key is temporarily saved on the Manager Device
Joining the DOMAIN

› When joining a domain the Manager node public keys are installed on the new node
› The Manager signs the node key
› The Manager distributes the node key using DHT

Add A.ID:[A.PUB, SIGN(A.PUB)] to its own DHT bucket
Key distribution in the DOMAIN

A → MANAGER → C → D → E

B.ID[B.PUB, SIGN]
C.ID[C.PUB, SIGN]
D.ID[D.PUB, SIGN]
E.ID[E.PUB, SIGN]

sign( )
Elliptic Curve Crypto

› Needed to make keys and messages shorter but as secure
› Requires processing power and some storage
   – Different curves require different processing power and time

<table>
<thead>
<tr>
<th>Platform</th>
<th>Author of implementation</th>
<th>Curve</th>
<th>ms</th>
<th>Micro Joule</th>
<th>Clock</th>
<th>Instruciton Set</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARM7TDMI</td>
<td>MIRACL</td>
<td>P-192</td>
<td>38r</td>
<td>182.4e</td>
<td>80Mhz</td>
<td>ARM 32bit</td>
</tr>
<tr>
<td>ARM7TDMI</td>
<td>MIRACL</td>
<td>P-224</td>
<td>53r</td>
<td>254.4e</td>
<td>80Mhz</td>
<td>ARM 32bit</td>
</tr>
<tr>
<td>ATMega128L</td>
<td>Aranha et al.</td>
<td>K-163</td>
<td>320r</td>
<td>9600e</td>
<td>7,37Mhz</td>
<td>Avr 8bit</td>
</tr>
<tr>
<td>ATMega128L</td>
<td>Karglet al</td>
<td>167-bitb</td>
<td>763r</td>
<td>24840e</td>
<td>8Mhz</td>
<td>Avr 8bit</td>
</tr>
<tr>
<td>ATMega128L</td>
<td>Aranha et al.</td>
<td>K-233</td>
<td>730r</td>
<td>21900e</td>
<td>7,37Mhz</td>
<td>Avr 8bit</td>
</tr>
<tr>
<td>Cortex-M0+</td>
<td>Ruan de Clercq</td>
<td>sect233kl</td>
<td>39.70f</td>
<td>20.63m</td>
<td>40Mhz</td>
<td>ARM 32bit</td>
</tr>
<tr>
<td>Cortex-M0+</td>
<td>Ruan de Clercq</td>
<td>sect233kl</td>
<td>59.18r</td>
<td>34.16m</td>
<td>40Mhz</td>
<td>ARM 32bit</td>
</tr>
<tr>
<td>Cortex-M0+</td>
<td>Relic kP</td>
<td>K-233</td>
<td>115.7r</td>
<td>69.48m</td>
<td>40Mhz</td>
<td>ARM 32bit</td>
</tr>
<tr>
<td>Cortex-M0+</td>
<td>Relic KG</td>
<td>K-233</td>
<td>117.1f</td>
<td>70.26m</td>
<td>40Mhz</td>
<td>ARM 32bit</td>
</tr>
<tr>
<td>MSP430F1611</td>
<td>NanoECC</td>
<td>P-160</td>
<td>720f</td>
<td>8847m</td>
<td>8.192Mhz</td>
<td>MSP430 16bit</td>
</tr>
<tr>
<td>MSP430F1611</td>
<td>NanoECC</td>
<td>K-163</td>
<td>1040f</td>
<td>12780m</td>
<td>8.192Mhz</td>
<td>MSP430 16bit</td>
</tr>
</tbody>
</table>

f=fixed point  
m=measurement  
r=random point  
e=estimated
A domain beyond the mesh

› A node can be residing
  – on the local carrier
    › ZigBee, Bluetooth, or Wi-Fi
  – on a remote carrier
    › Any topology
  – in a cloud entity
  – or as an actor in a (distributed) runtime
Manual upgrades?
Secure Upgrades

› Lightweight Machine To Machine protocol (LWM2M)
  – OMA-TS-LightweightM2M-V1
› The LWM2M upgrade server can act as a node in the domain
› Uses DTLS for End to End encrypted channels
› Supports Secure Channel Protocol from Global Platform to derive node bootstrap keys from a Secure Element
› Makes attacks like BEAST, POODLE and HEARTBLEED short-lived
› Retrieve seamless upgrades from a trusted remote upgrade server node
Availability