TF-M Profile Proposal
Why?

• Dramatic variation in device capabilities and use cases
  • Secure software takes significant portion of hardware resources
  • Diverse use-cases with differing security requirements

• PSA vision is to raise the bar on security and make security easier
  • Is the market ready to pay the price for security?

• All use cases don’t need same level of security

• ALL use cases don’t need ALL of the security

• TF-M current memory usage poses a challenge for usage in ultra constrained devices
Profile Proposal

- Predefined list of base profiles
- Targeted towards use-cases with different hardware constraints
- Proven to work, tested in CI
- Alignment with PSA specifications and certification requirements
## Memory Usage Today on MuscaB1e

<table>
<thead>
<tr>
<th>Build Config</th>
<th>Compiler</th>
<th>Code + RO Data</th>
<th>RW + ZI Data</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConfigCoreIPCTfmLevel2 (Level2 Isolation, IPC)</td>
<td>ARMCLANG</td>
<td>122k</td>
<td>64k</td>
<td>Audit Log Secure Partition Not Present. Separate Stack for each partition.</td>
</tr>
<tr>
<td></td>
<td>GCC</td>
<td>127kB</td>
<td>64K</td>
<td></td>
</tr>
<tr>
<td>ConfigDefault (Level1 Isolation, Lib Mode)</td>
<td>ARMCLANG</td>
<td>124k</td>
<td>49k</td>
<td></td>
</tr>
<tr>
<td></td>
<td>GCC</td>
<td>129K</td>
<td>49K</td>
<td></td>
</tr>
</tbody>
</table>
# Memory Usage Today on MuscaB1e

<table>
<thead>
<tr>
<th>Partition</th>
<th>Code + RO Data</th>
<th>RW + ZI Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>TF-M Core</td>
<td>24K</td>
<td>13K</td>
</tr>
<tr>
<td>Crypto</td>
<td>88K</td>
<td>36K</td>
</tr>
<tr>
<td>Secure Storage</td>
<td>6K</td>
<td>12K</td>
</tr>
<tr>
<td>Attestation</td>
<td>4K</td>
<td>3K</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>122k</strong></td>
<td><strong>64k</strong></td>
</tr>
<tr>
<td>Secure Boot</td>
<td>20K</td>
<td>22K</td>
</tr>
</tbody>
</table>

@10 Sept 19
TF-M Master
Profile 1

• Lightweight boot
  • No rollback protection, Single binary (SPE+NSPE)

• Lightweight Framework
  • L1 isolation, Library/SFC mode, Buffer sharing allowed
  • Single secure context, Secure stack defined at initialization

• Storage
  • eFlash available, ITS, No encryption
  • No internal transient buffers, client buffers used, No rollback protection

• Crypto
  • Symmetric (say AES), Cipher Suite for PSK TLS (say HMAC, SHA-256). Leverage HW Crypto

• Attestation
  • Compile time generated token structure, Only IAT
  • HMAC based authentication.
Profile 2

- Lightweight boot
  - Rollback protection, Single binary (SPE+NSPE)

- Lightweight Framework
  - L1/L2 isolation, buffer sharing allowed in L1
  - Multiple secure context, secure stack defined at initialization
  - Secure side shadows the NSPE scheduler

- Storage
  - eFlash available, ITS, No encryption, Protected Storage (Optional)
  - Scalable internal transient buffers, No rollback protection

- Crypto
  - Symmetric & Asymmetric (say AES), Cipher Suite for TLS1.2 (say AES-128-GCM/CCM, ECDSA, RSA,ECDH,SHA-256,HMAC)

- Attestation
  - Compile time generated token structure, Only IAT
Profile 3

- Profile 2 +
- Level3 Isolation
- Audit Log
- Everything else