Bitcoin2Go: Secure Offline and Fast Payments with Bitcoins

Alexandra Dmitrienko, David Noack, Ahmad-Reza Sadeghi, and Moti Yung

**Motivation**
- Bitcoin thrives to be the most successful crypto currency
- Transaction verification in Bitcoin requires online access and time
- Not applicable in offline payment scenarios (e.g., PoS, vending machines, etc.)
- Fast payments with Bitcoins are vulnerable to double spending attacks

**Related Work**
- Covers security, privacy and economical aspects of Bitcoin online payments
- No attempts to adapt to offline scenarios

**Challenges of Offline Payments**
- Double-spending prevention and detection of forged coins in offline settings

**System Architecture**

**Security Mechanisms**
- Secure, but resource constraint wallet
  - Prevents double spending in offline phase
  - Resource constraints raise challenge to verify pre-loading transactions
  - Time-based transaction confirmation
    - Restricts standard Bitcoin transaction confirmation generation by max. time
    - Probability to produce valid confirmation is high for the Bitcoin network, but low for an attacker
  - Limited transaction amounts
    - Attack costs become higher than benefits

**Features**
- Compatible to Bitcoin system
- Designed for resource constraint wallets
- Can be used by PCs and smartphone-based clients

**Prototype Implementation**

**Risk Analysis**

<table>
<thead>
<tr>
<th>Risk Analysis</th>
<th>Set 1</th>
<th>Set 2</th>
<th>Set 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attacker’s hashrate, %</td>
<td>10</td>
<td>15</td>
<td>20</td>
</tr>
<tr>
<td>Min. length of the trans. conf., blocks</td>
<td>6</td>
<td>7</td>
<td>10</td>
</tr>
<tr>
<td>Max. time for generation of a single block, sec.</td>
<td>1000</td>
<td>1200</td>
<td>1200</td>
</tr>
<tr>
<td>Attack probability, %</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Attack costs per block, EUR</td>
<td>560</td>
<td>560</td>
<td>560</td>
</tr>
</tbody>
</table>

Host platform:
- Galaxy S3 with Android 4.0.4

Wallet:
- Secure element in microSD card
- JavaCard 2.2.2, JCOP 2.4.1
- 81 Kb EEPROM, 20 Kb code, 1 Kb RAM footprint