Crypto-Conditions
A Standard for Composable Signatures

Stefan Thomas
Holds Are Dependent on Conditions + Expiries

ROLLBACK  EXECUTE
Condition Fulfillment Executes Transfer
Cryptography
Are Simple Signatures Sufficient?
Are Simple Signatures Sufficient?
Are Simple Signatures Sufficient?

THE WALL STREET JOURNAL.

FBI Suspects Insider Involvement in $81 Million Bangladesh Bank Heist

Computer hackers tried to steal nearly $1 billion in a brazen attack
How Multi-Signatures Can Help

2-of-3
How Multi-Signatures Can Help

2-of-3
Condition Types

Signatures

Thresholds
Condition Types

Signatures

Thresholds
Composition Use Case Example

2-of-3

2-of-2

2-of-3
Prior Art: Bitcoin Scripts

2 \langle K1 \rangle \langle K2 \rangle \langle K3 \rangle 3 \text{ CHECKMULTISIGVERIFY}

- Forth-like language
- Many opcodes disabled
- Primary use: m-of-n multi-signature
Related Standards

RFC 5652, RFC 5752
Multiple Signatures in Cryptographic Message Syntax (CMS)

RFC 7515 — Jones, Bradley, Sakimura
JSON Web Signature (JWS)

- Minimal support for multiple signatures
- **No structured keys**
Structured Keys

Suppose Alice receives a key from Bob...
Suppose Alice receives a key from Foo Bank. The **structure** of Foo Bank's key shouldn't matter to her.

(Within reason — we'll get to that later.)
Other Related Work

- RFC 5752 — Turner and Schaad
- Pay-to-script-hash (P2SH) — Andresen
- Tree Signatures — Wuille
- Merkleized Abstract Syntax Trees (MAST) — Rubin et al
- Smart Signatures — Allen et al
- Deterministic Expressions (DEX) — Todd
- State Channels — Coleman
- Multihash — Benet et al
Call for Standardization

May 21st 2016

**Smart Signatures**

Christopher Allen & Shannon Appelcline
Complexity Spectrum

Single Signature  Multi Signature  Programming Language
Complexity Spectrum

- Single Signature
- Multi Signature
- Programming Language
Complexity Spectrum

- Single Signature
- Multi Signature
- Programming Language
Why Do We Care About Minimalism?
Why Do We Care About Minimalism?
Conditions MUST Be **Bit-Perfect**

- ROLLBACK
- EXECUTE
Condition Implementations MUST Be **Bit-Perfect**
Flexibility vs Simplicity

Single Signature  Multi Signature  ?  Programming Language
Delegation
Smart Oracle
Smart Oracle

2-of-3
Boolean Threshold Circuits

Chandra  Lina  Bob

Millie

2
Proposal
draft-thomas-crypto-conditions-00
<table>
<thead>
<tr>
<th>Github</th>
<th>Language</th>
<th>Implementer</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>bigchaindb/cryptoconditions</td>
<td>Python</td>
<td>Ascribe/BigchainDB</td>
<td>Complete</td>
</tr>
<tr>
<td>interledger/five-bells-condition</td>
<td>JavaScript</td>
<td>Ripple</td>
<td>Complete</td>
</tr>
<tr>
<td>jtremback/crypto-conditions</td>
<td>Go</td>
<td>Althea</td>
<td>Partial</td>
</tr>
</tbody>
</table>
Condition — Printable Encoding

c:4:20:RCmTB1AEqh5MSPTdAVgZTAI0m8xmTN1uQA6iaZGKjVE:96
Condition — Binary Encoding

\[
\text{Condition} ::= \text{SEQUENCE} \{
\begin{align*}
\text{type} &\colon \text{ConditionType}, \\
\text{featureBitmask} &\colon \text{OCTET STRING}, \\
\text{fingerprint} &\colon \text{OCTET STRING}, \\
\text{maxFulfillmentLength} &\colon \text{INTEGER (0..MAX)}
\end{align*}
\}
\]

\[
\text{ConditionType} ::= \text{INTEGER} \{
\begin{align*}
\text{preimageSha256(0)}, \\
\text{rsaSha256(1)}, \\
\text{prefixSha256(2)}, \\
\text{thresholdSha256(3)}, \\
\text{ed25519(4)}
\end{align*}
\} (0..65535)\]
Fulfillment — Printable Encoding

cf:4:
RCmTB1AЕqh5MSPTdAVgZTAI0m8xmTN1uQA6iaZGKjVGfTbzglso5
Uo3i2O2WVP6abH1dz5k0H5DLyliTeL5UC0VSptUN4VCKhtbwx3B
00pCeWNy1H78rq6OTXzok-EH
Fulfillment — Binary Encoding

Fulfillment ::= SEQUENCE {
    type ConditionType,
    payload OCTET STRING
}

Ed25519FulfillmentPayload ::= SEQUENCE {
    publicKey OCTET STRING (SIZE(32)),
    signature OCTET STRING (SIZE(64))
}
Threshold Condition

ThresholdSha256FingerprintContents ::= SEQUENCE {
  threshold INTEGER (0..4294967295),
  subconditions SEQUENCE OF ThresholdSubcondition
}

ThresholdSubcondition ::= SEQUENCE {
  weight INTEGER (0..4294967295),
  condition Condition
}
Threshold Condition

cc:2:2b:d304epRC0_3rj17Bf3v8hp5ig7vq84ivPok07T9Rd10:146
Threshold Fulfillment

ThresholdSha256FulfillmentPayload ::= SEQUENCE {
  threshold INTEGER (0..4294967295),
  subfulfillments SEQUENCE OF ThresholdSubfulfillment
}

ThresholdSubfulfillment ::= SEQUENCE {
  weight INTEGER (0..4294967295),
  condition Condition,
  fulfillment Fulfillment
}
Threshold Fulfillment

cf: 2:
AQEBAgEBAAAQAAEJwAEASAgIHahWSBEpuT1ESZbynOmBNkLBSnR32Ar4woZqSV2YNEBYA
Merkle Circuits
Threshold Fulfillment

cf:2:
AQEBAgEBBAAAQAAAQEAEJwAEASAgIHahWSBEpuT1ESZbynOmBNkLBSnR32Ar4woZqSV2YNEBYA
Threshold Fulfillment

cf:2:
AQEBAgEBAAAAAQAAAEJwAEASAgIHahWSBEpuT1ESZbynOmBNkLBSnR32Ar4woZqSV2YNEBYA
Threshold Fulfillment

cf:2:
AQEBAgEBBAAAQAAAAQEAJwAEASAgIHahWSBEpuT1ESZbynOmBNkLBS
nR32Ar4woZqSV2YNEBYA
Bitmask
Condition Contains Feature Bitmask
## Bitmask — Simple Approach

### Types

<table>
<thead>
<tr>
<th>Type</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>preimageSha256</td>
</tr>
<tr>
<td>1</td>
<td>rsaSha256</td>
</tr>
<tr>
<td>2</td>
<td>prefixSha256</td>
</tr>
<tr>
<td>3</td>
<td>thresholdSha256</td>
</tr>
<tr>
<td>4</td>
<td>ed25519</td>
</tr>
</tbody>
</table>

### Feature Bits

<table>
<thead>
<tr>
<th>Feature Bits</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>______ 1</td>
<td>preimageSha256</td>
</tr>
<tr>
<td>______ 1</td>
<td>rsaSha256</td>
</tr>
<tr>
<td>______ 1</td>
<td>prefixSha256</td>
</tr>
<tr>
<td>______ 1</td>
<td>thresholdSha256</td>
</tr>
<tr>
<td>______ 1</td>
<td>ed25519</td>
</tr>
</tbody>
</table>
## Bitmask — Simple Approach

### Types

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>preimageSha256</td>
</tr>
<tr>
<td>1</td>
<td>rsaSha256</td>
</tr>
<tr>
<td>2</td>
<td>prefixSha256</td>
</tr>
<tr>
<td>3</td>
<td>thresholdSha256</td>
</tr>
<tr>
<td>4</td>
<td>ed25519</td>
</tr>
<tr>
<td>5</td>
<td>preimageSha512</td>
</tr>
<tr>
<td>6</td>
<td>rsaSha512</td>
</tr>
<tr>
<td>7</td>
<td>prefixSha512</td>
</tr>
<tr>
<td>8</td>
<td>thresholdSha512</td>
</tr>
</tbody>
</table>

### Feature Bits

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- preimageSha256
- rsaSha256
- prefixSha256
- thresholdSha256
- ed25519
- preimageSha512
- rsaSha512
- prefixSha512
- thresholdSha512
# Bitmask — Feature Suites

## Types

<table>
<thead>
<tr>
<th></th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>preimageSha256</td>
</tr>
<tr>
<td>1</td>
<td>rsaSha256</td>
</tr>
<tr>
<td>2</td>
<td>prefixSha256</td>
</tr>
<tr>
<td>3</td>
<td>thresholdSha256</td>
</tr>
<tr>
<td>4</td>
<td>ed25519</td>
</tr>
</tbody>
</table>

## Feature Bits

<table>
<thead>
<tr>
<th></th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>______1</td>
<td>sha256</td>
</tr>
<tr>
<td>_____<em>1</em></td>
<td>preimage</td>
</tr>
<tr>
<td>____<strong>1</strong></td>
<td>prefix</td>
</tr>
<tr>
<td>____<strong>1</strong></td>
<td>threshold</td>
</tr>
<tr>
<td>_<em><strong>1</strong></em></td>
<td>rsa</td>
</tr>
<tr>
<td><em><strong>1</strong></em>_</td>
<td>ed25519</td>
</tr>
</tbody>
</table>
# Bitmask — Feature Suites

## Types

<table>
<thead>
<tr>
<th></th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>preimageSha256</td>
</tr>
<tr>
<td>1</td>
<td>rsaSha256</td>
</tr>
<tr>
<td>2</td>
<td>prefixSha256</td>
</tr>
<tr>
<td>3</td>
<td>thresholdSha256</td>
</tr>
<tr>
<td>4</td>
<td>ed25519</td>
</tr>
<tr>
<td>5</td>
<td>preimageSha512</td>
</tr>
<tr>
<td>6</td>
<td>rsaSha512</td>
</tr>
<tr>
<td>7</td>
<td>prefixSha512</td>
</tr>
<tr>
<td>8</td>
<td>thresholdSha512</td>
</tr>
</tbody>
</table>

## Feature Bits

<table>
<thead>
<tr>
<th></th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>sha256</td>
</tr>
<tr>
<td></td>
<td>preimage</td>
</tr>
<tr>
<td></td>
<td>prefix</td>
</tr>
<tr>
<td></td>
<td>threshold</td>
</tr>
<tr>
<td></td>
<td>rsa</td>
</tr>
<tr>
<td></td>
<td>ed25519</td>
</tr>
<tr>
<td>___</td>
<td>sha512</td>
</tr>
</tbody>
</table>
## Bitmask — Allows Feature Restriction

#### Types

<table>
<thead>
<tr>
<th></th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>preimageSha256</td>
</tr>
<tr>
<td>1</td>
<td>rsaSha256</td>
</tr>
<tr>
<td>2</td>
<td>prefixSha256</td>
</tr>
<tr>
<td>3</td>
<td>thresholdSha256</td>
</tr>
<tr>
<td>4</td>
<td>ed25519</td>
</tr>
</tbody>
</table>

#### Feature Bits

<table>
<thead>
<tr>
<th></th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>sha256</td>
</tr>
<tr>
<td></td>
<td>preimage</td>
</tr>
<tr>
<td></td>
<td>prefix</td>
</tr>
<tr>
<td></td>
<td>threshold</td>
</tr>
<tr>
<td></td>
<td>rsa</td>
</tr>
<tr>
<td></td>
<td>ed25519</td>
</tr>
</tbody>
</table>

⇒ Smaller Implementation Size, Smaller Attack Surface
(Max) Fulfillment Length
Condition Contains Fulfillment Length

cc:4:20:RCmTB1AEqh5MSPTdAVgZTAI0m8xmTNl6uQA6iaZGKjVE:96
Fulfillment Size Must Be Supported By Everyone

ROLLBACK  EXECUTE
Prefix Conditions
Prefix Condition

Same key, but pretend the message has an additional prefix.

- Create a more limited scoped key
- Works with any condition type
- Enables fixed message conditions
Terminology
Signature Schemes Are a (Cryptographic) Condition

- **Condition** ... like a public key; provided up-front
- **Fulfillment** ... like a signature; cryptographic proof
- **Message** ... actual data to validate against
Other Types of Crypto-Conditions

<table>
<thead>
<tr>
<th>Condition</th>
<th>Fulfillment</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>preimage = hash</td>
<td>⇒ preimage</td>
<td>⇒ () ⇒ ...</td>
</tr>
<tr>
<td>signature = pubkey</td>
<td>⇒ signature</td>
<td>⇒ message ⇒ ...</td>
</tr>
<tr>
<td>prefix = (subcondition, prefix)</td>
<td>⇒ subfulfillment</td>
<td>⇒ message ⇒ ...</td>
</tr>
<tr>
<td>threshold = (subconditions, threshold)</td>
<td>⇒ subfulfillments</td>
<td>⇒ message ⇒ ...</td>
</tr>
</tbody>
</table>

- Condition ... like a public key; provided up-front
- Fulfillment ... like a signature; cryptographic proof
- Message ... actual data to validate against
Extensibility
Possible Future Extensions

- Larger hash sizes (512-bits)
- Quantum-secure signatures (SPHINCS, etc.)
- Subdelegation Condition
- Rehash/HMAC Condition
- Homomorphic Hashes
- ...


Recap
Crypto-Conditions

Standard for composable signatures
(aka multi-sig on steroids)

- Minimal
- Composable
- Extensible/Upgradable
- Verifiable keys
- Efficient (using Merkle circuits)
Open Questions

- Should the bitmask be variable or fixed length? (currently: variable)
- Would a rehash/HMAC condition be safer than a prefix condition?
- Should we support ECDSA (e.g. P-256), Ed25519 or both?
- Is OER encoding the right choice?
Discussion