Subterm-based proof techniques for improving the automation and scope of security protocol analysis

Cas Cremers ⧺ Charlie Jacomme ⧺ **Philip Lukert**
∃t. Cispa ⊏ Saarbrücken ⊏ Germany

Subterm-based p
techni
th scope of
s protocol analysis

Subterms for Tamarin

Cas Cremers ⧺ Charlie Jacomme ⧺ **Philip Lukert**
∃t. Cispa ⊏ Saarbrücken ⊏ Germany

Subterm-based p… techni… th… scope of s… y protocol analysis

Subterms for Tamarin

Cas Cremers ⧺ Charlie Jacomme ⧺ **Philip Lukert**
∃t. Cispa ⊏ Saarbrücken ⊏ Germany

Protocol Analysis

Subterm-based p…
techni… th… scope of s… protocol analysis

Subterms for Tamarin

Cas Cremers ⧺ Charlie Jacomme ⧺ **Philip Lukert**

∃t. Cispa ⊏ Saarbrücken ⊏ Germany

Protocol Analysis → **Data-structures**

Subterm-based p…
techniq…
th… scope of
s… protocol analysis

Subterms for Tamarin

Cas Cremers ⧺ Charlie Jacomme ⧺ **Philip Lukert**

∃t. Cispa ⊏ Saarbrücken ⊏ Germany

Protocol Analysis → **Data-structures** → **Subterms**

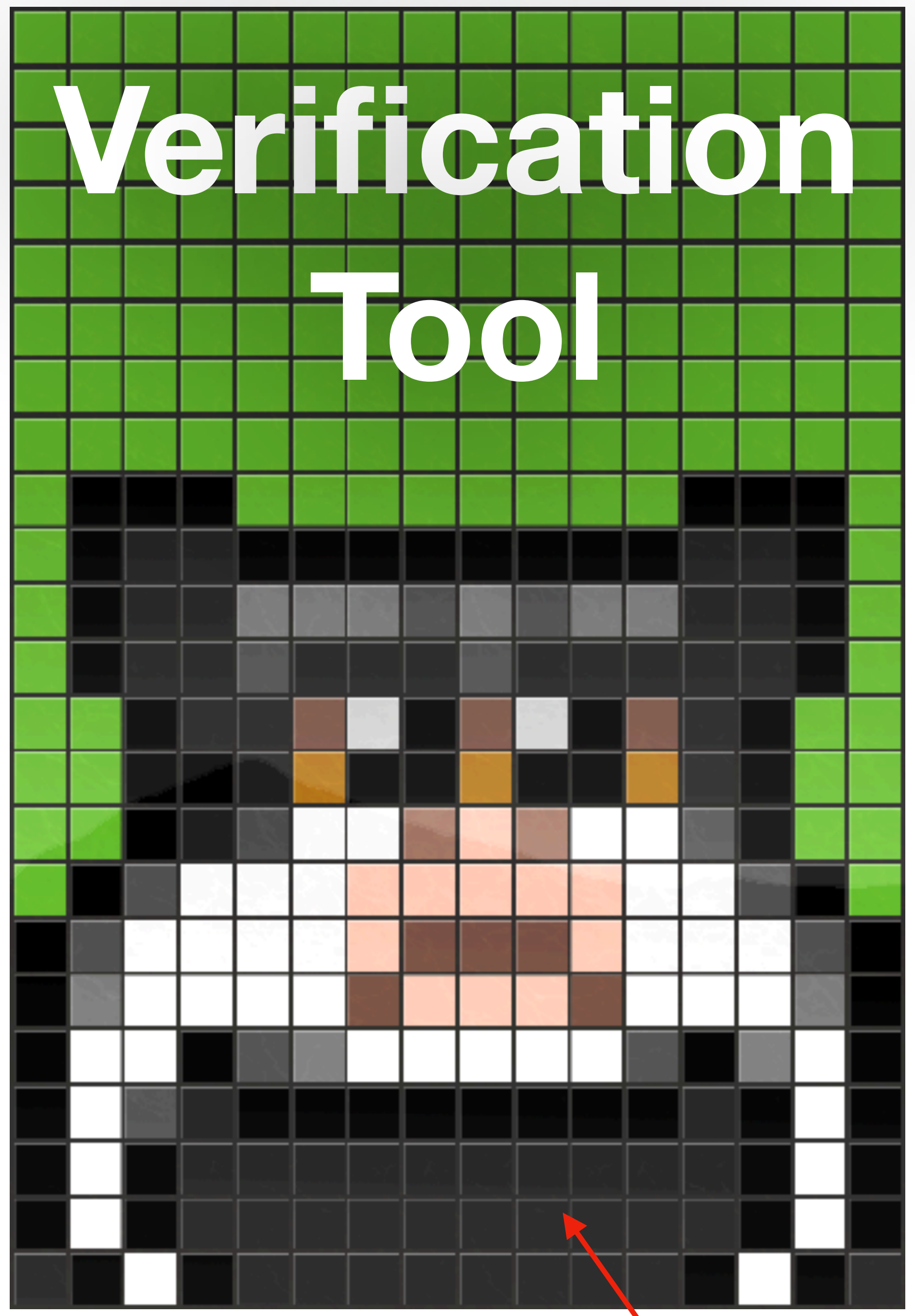
Subterm-based p
techniq
th scope of
s protocol analysis

Subterms for Tamarin

Cas Cremers ⧺ Charlie Jacomme ⧺ **Philip Lukert**
∃t. Cispa ⊏ Saarbrücken ⊏ Germany

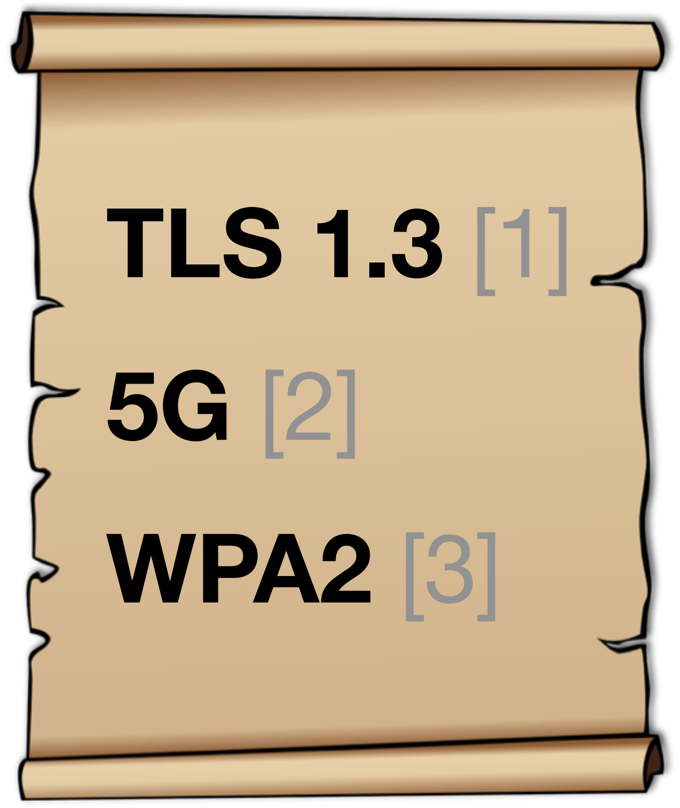
Protocol Analysis → **Data-structures** → **Subterms** → **Great Proofs**

Protocol Analysis

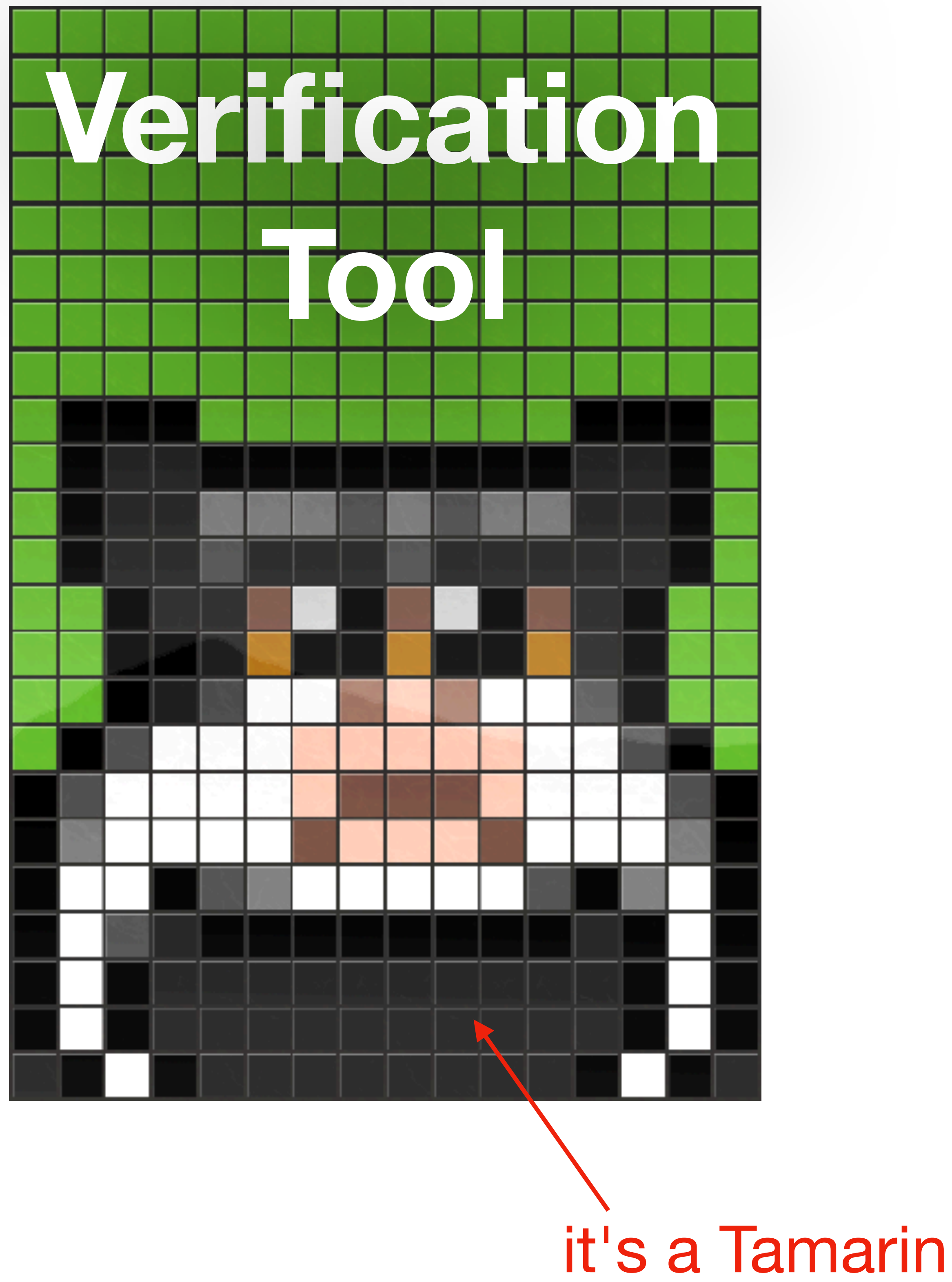


it's a Tamarin

Protocol Analysis

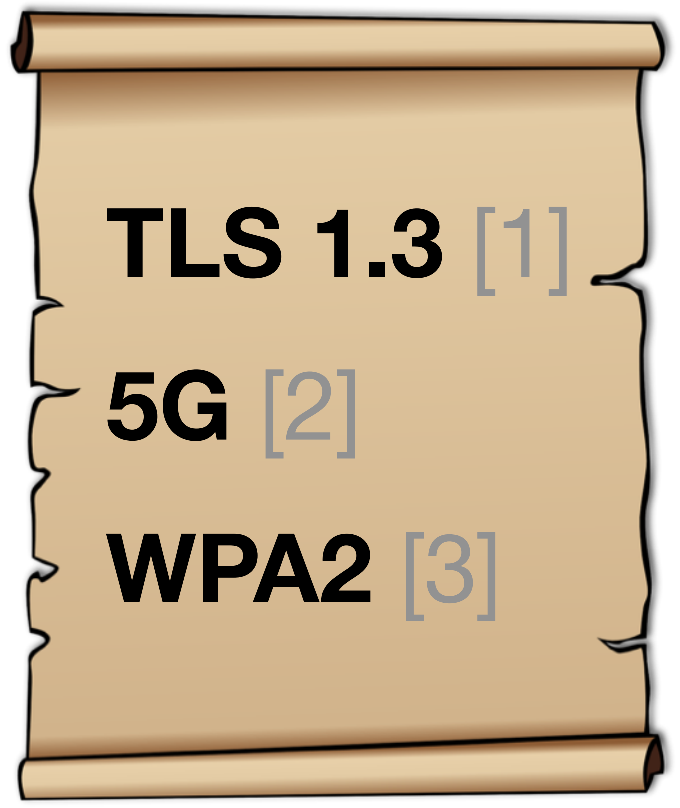


Protocol



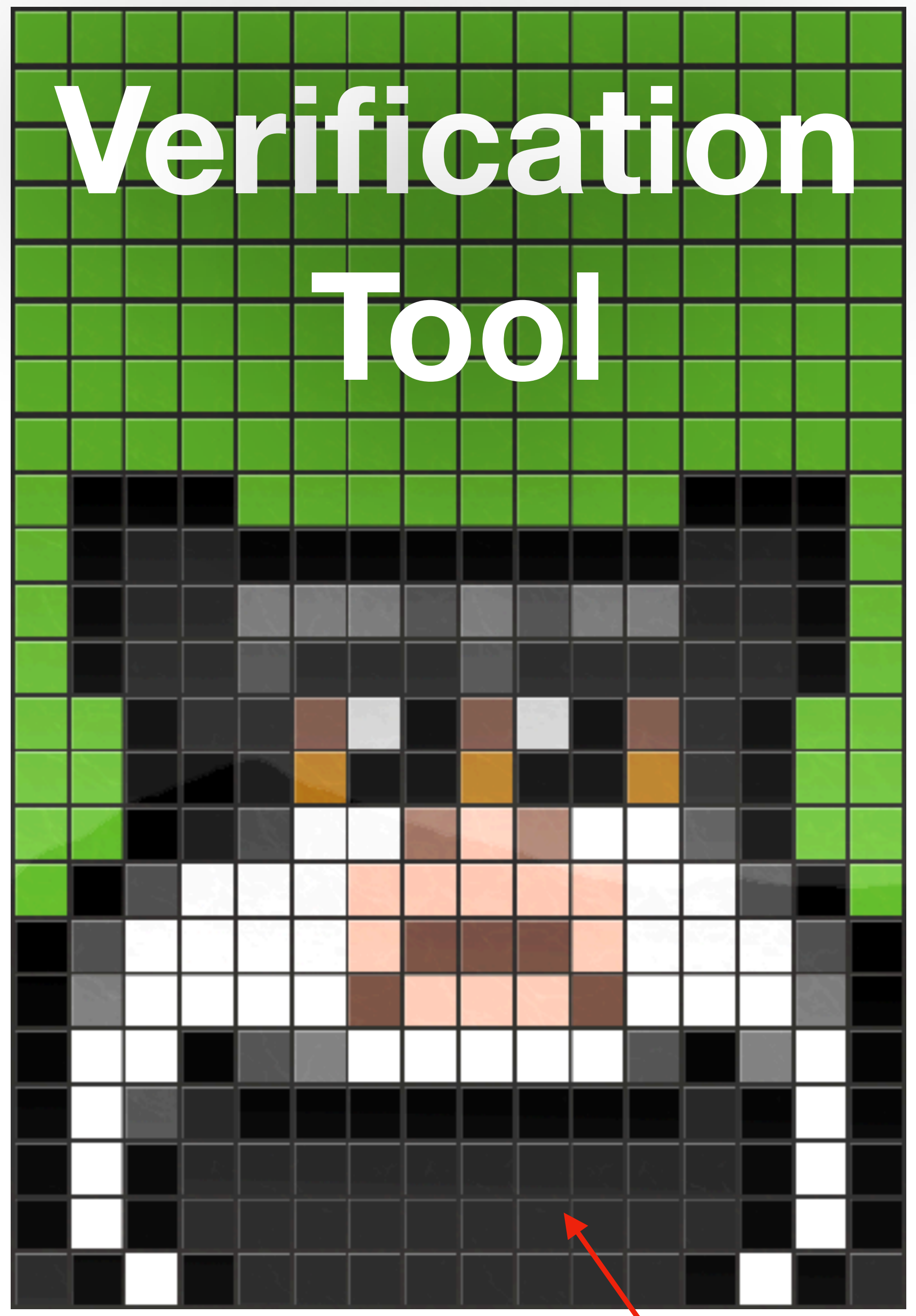
[1] Automated Analysis of TLS 1.3 (Cas Cremers, Marko Horvat, Sam Scott, Thyla van der Merwe)
[2] A Formal Analysis of 5G Authentication (David Basin, Jannik Dreier, Lucca Hirschi, Saša Radomirovic, Ralf Sasse, Vincent Stettler)
[3] A Formal Analysis of IEEE 802.11's WPA2 (Cas Cremers, Benjamin Kiesl, Niklas Medinger)

Protocol Analysis



Protocol

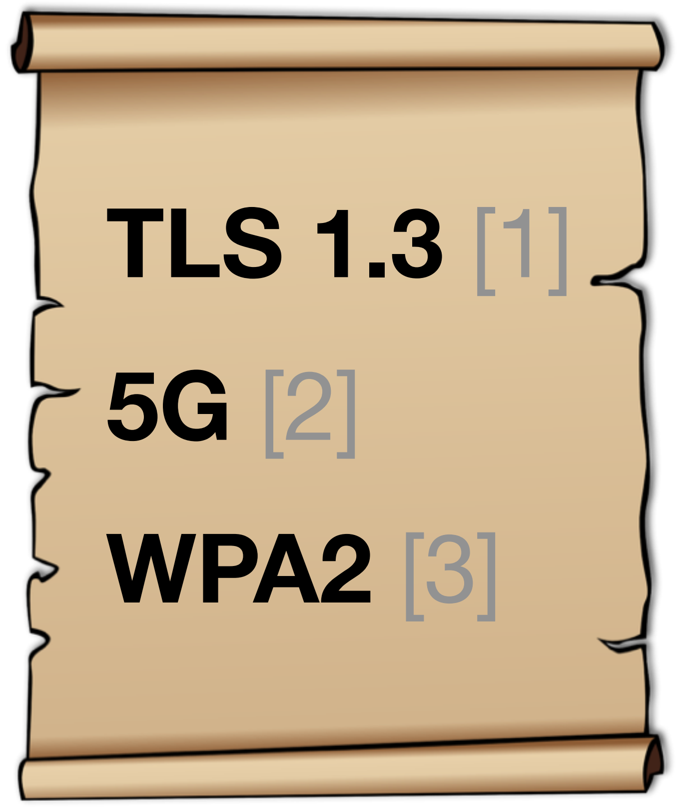
Property



it's a Tamarin

[1] Automated Analysis of TLS 1.3 (Cas Cremers, Marko Horvat, Sam Scott, Thyla van der Merwe)
[2] A Formal Analysis of 5G Authentication (David Basin, Jannik Dreier, Lucca Hirschi, Saša Radomirovic, Ralf Sasse, Vincent Stettler)
[3] A Formal Analysis of IEEE 802.11's WPA2 (Cas Cremers, Benjamin Kiesl, Niklas Medinger)

Protocol Analysis



Protocol

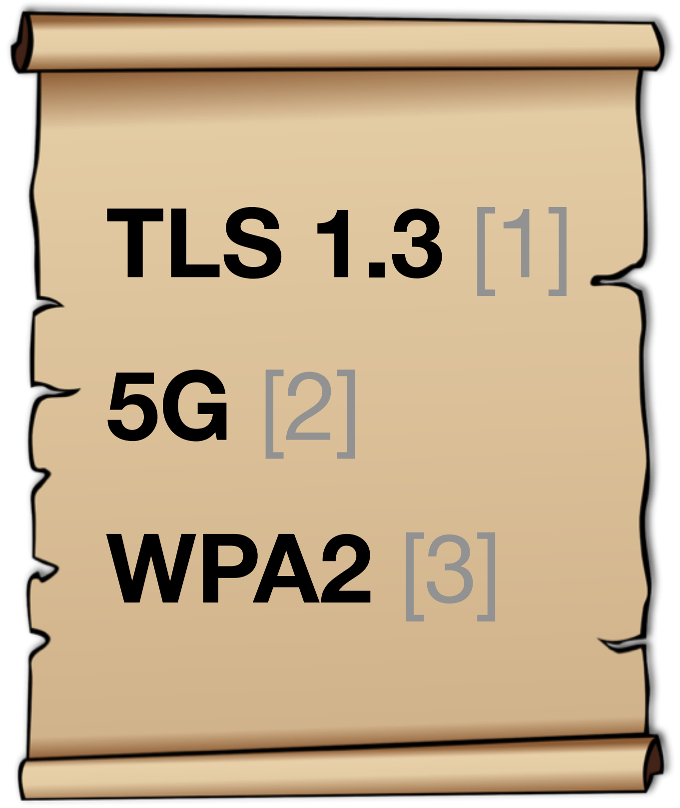
Property

Helper Lemmas

it's a Tamarin

[1] Automated Analysis of TLS 1.3 (Cas Cremers, Marko Horvat, Sam Scott, Thyla van der Merwe)
[2] A Formal Analysis of 5G Authentication (David Basin, Jannik Dreier, Lucca Hirschi, Saša Radomirovic, Ralf Sasse, Vincent Stettler)
[3] A Formal Analysis of IEEE 802.11's WPA2 (Cas Cremers, Benjamin Kiesl, Niklas Medinger)

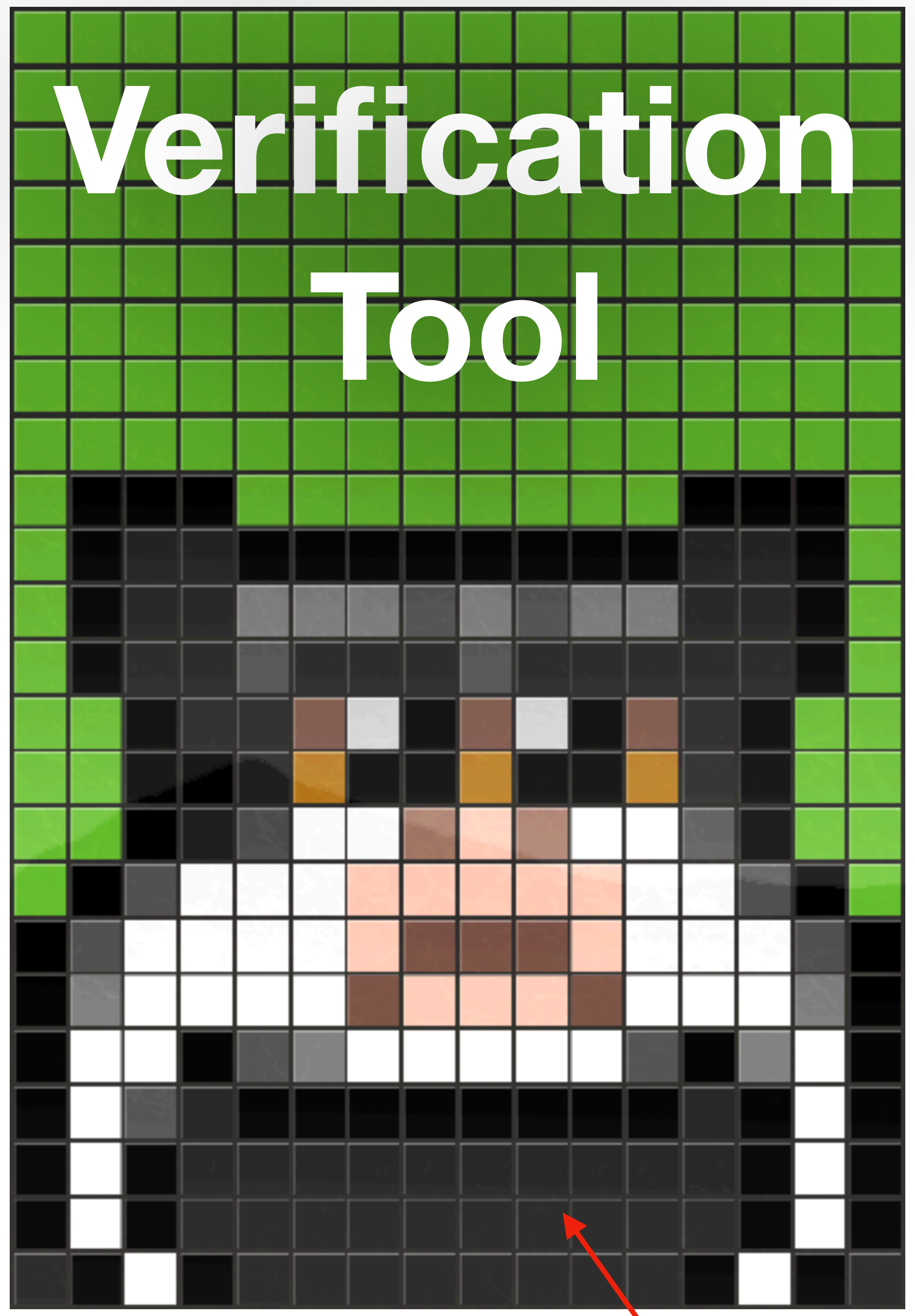
Protocol Analysis



Protocol

Property

Helper Lemmas

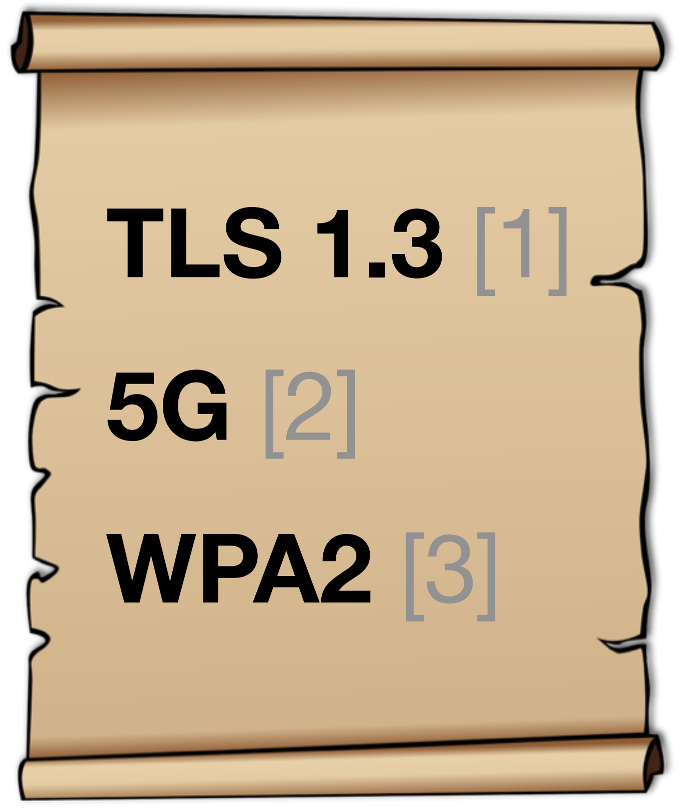


Proof

it's a Tamarin

[1] Automated Analysis of TLS 1.3 (Cas Cremers, Marko Horvat, Sam Scott, Thyla van der Merwe)
[2] A Formal Analysis of 5G Authentication (David Basin, Jannik Dreier, Lucca Hirschi, Saša Radomirovic, Ralf Sasse, Vincent Stettler)
[3] A Formal Analysis of IEEE 802.11's WPA2 (Cas Cremers, Benjamin Kiesl, Niklas Medinger)

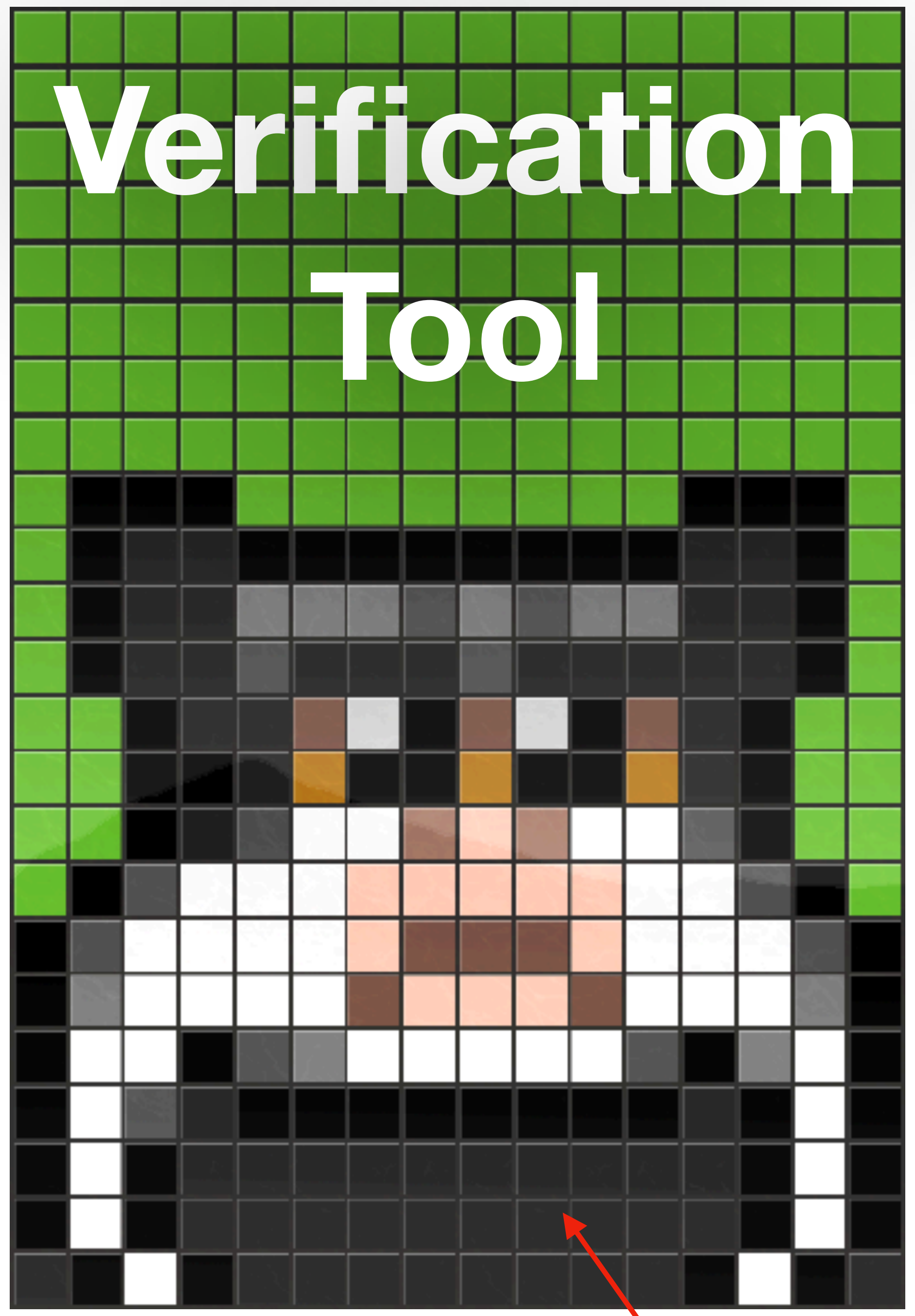
Protocol Analysis



Protocol

Property

Helper Lemmas



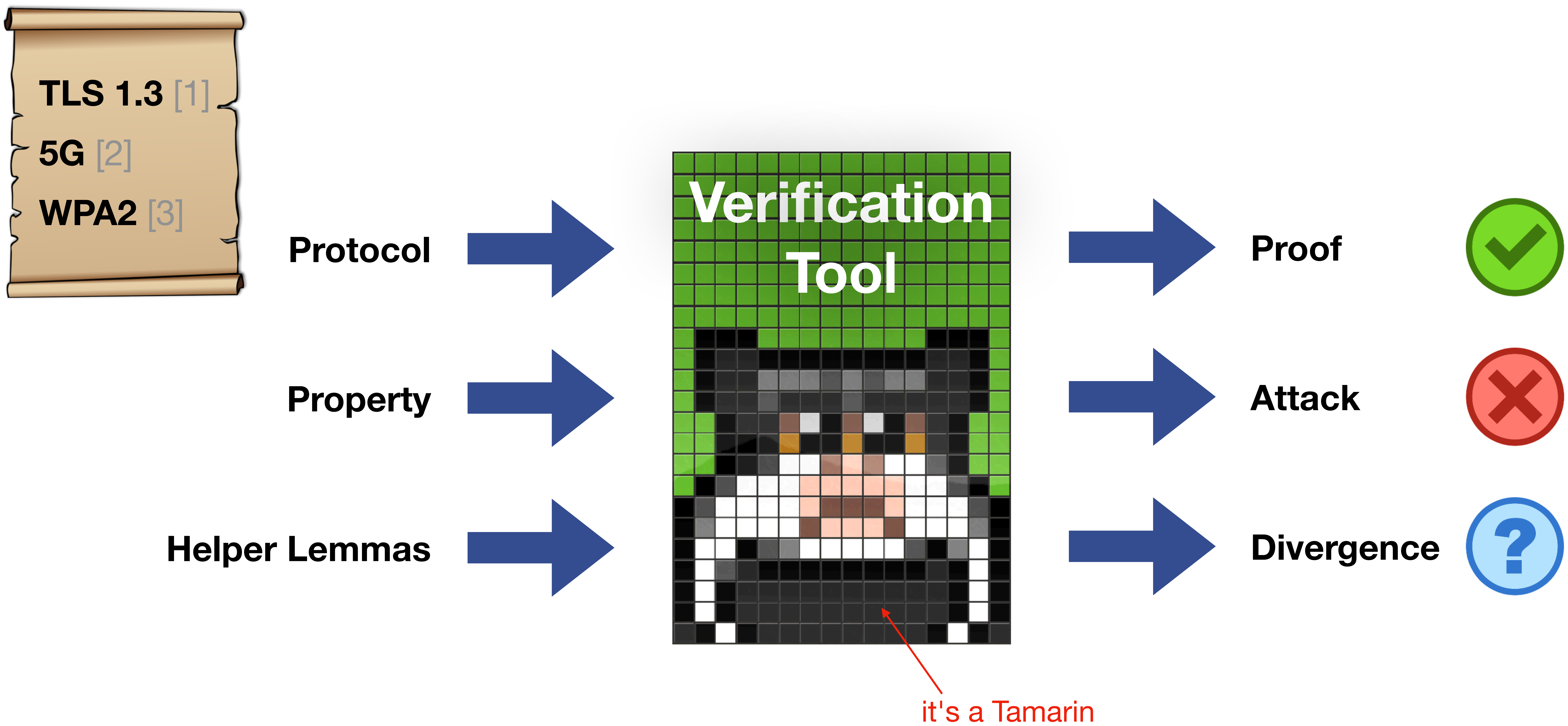
Proof

Attack

it's a Tamarin

[1] Automated Analysis of TLS 1.3 (Cas Cremers, Marko Horvat, Sam Scott, Thyla van der Merwe)
[2] A Formal Analysis of 5G Authentication (David Basin, Jannik Dreier, Lucca Hirschi, Saša Radomirovic, Ralf Sasse, Vincent Stettler)
[3] A Formal Analysis of IEEE 802.11's WPA2 (Cas Cremers, Benjamin Kiesl, Niklas Medinger)

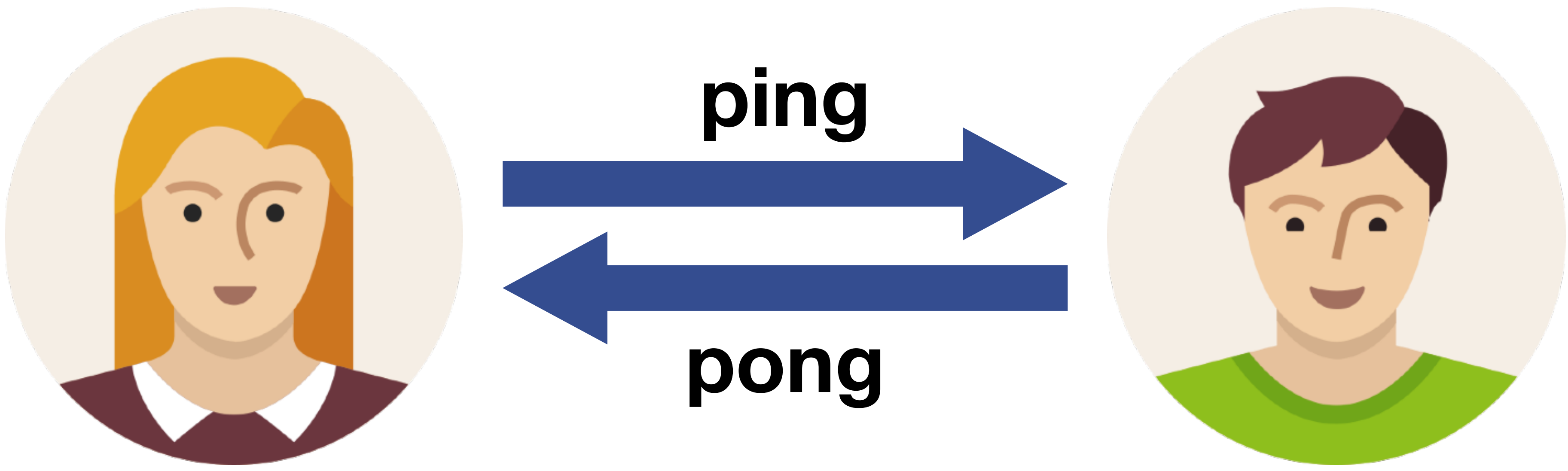
Protocol Analysis



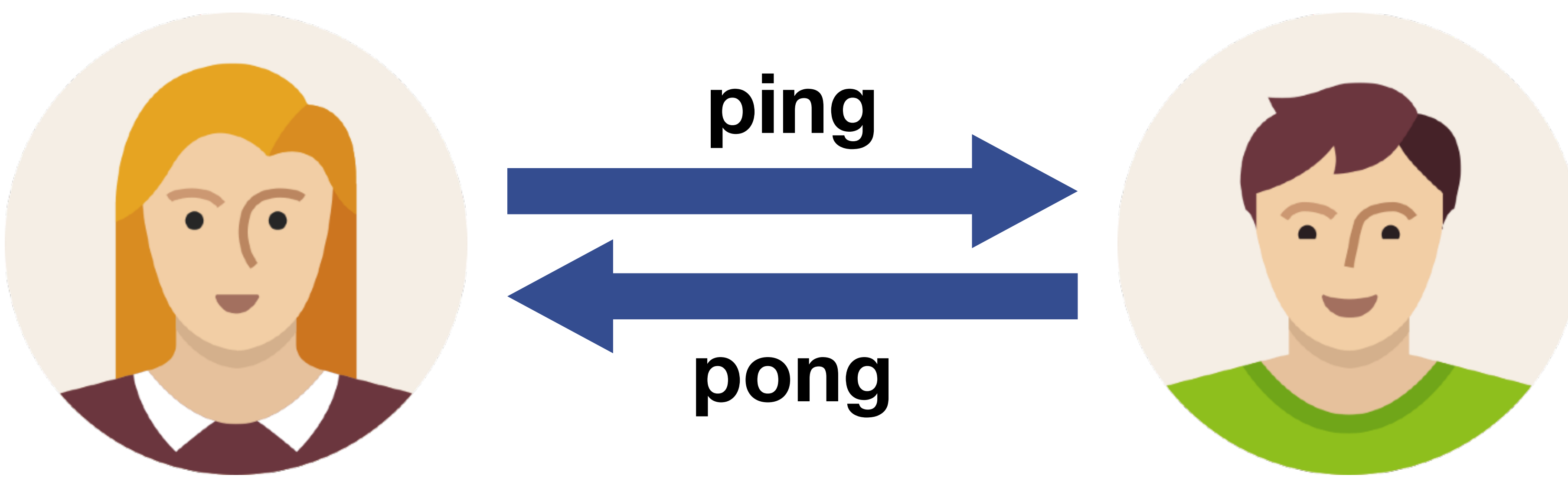
[1] Automated Analysis of TLS 1.3 (Cas Cremers, Marko Horvat, Sam Scott, Thyla van der Merwe)
[2] A Formal Analysis of 5G Authentication (David Basin, Jannik Dreier, Lucca Hirschi, Saša Radomirovic, Ralf Sasse, Vincent Stettler)
[3] A Formal Analysis of IEEE 802.11's WPA2 (Cas Cremers, Benjamin Kiesl, Niklas Medinger)

Tamarin

Tamarin



Tamarin



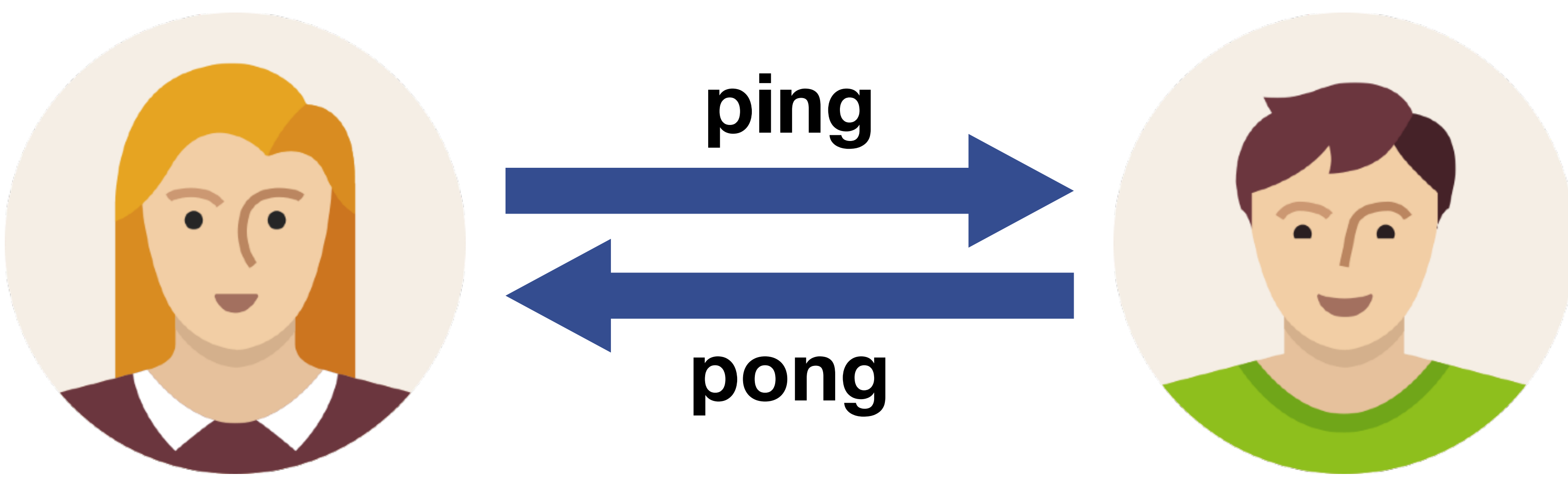
required state facts

action

produced state facts

[ ] --[Start]--> [Out("ping"), State()]

Tamarin



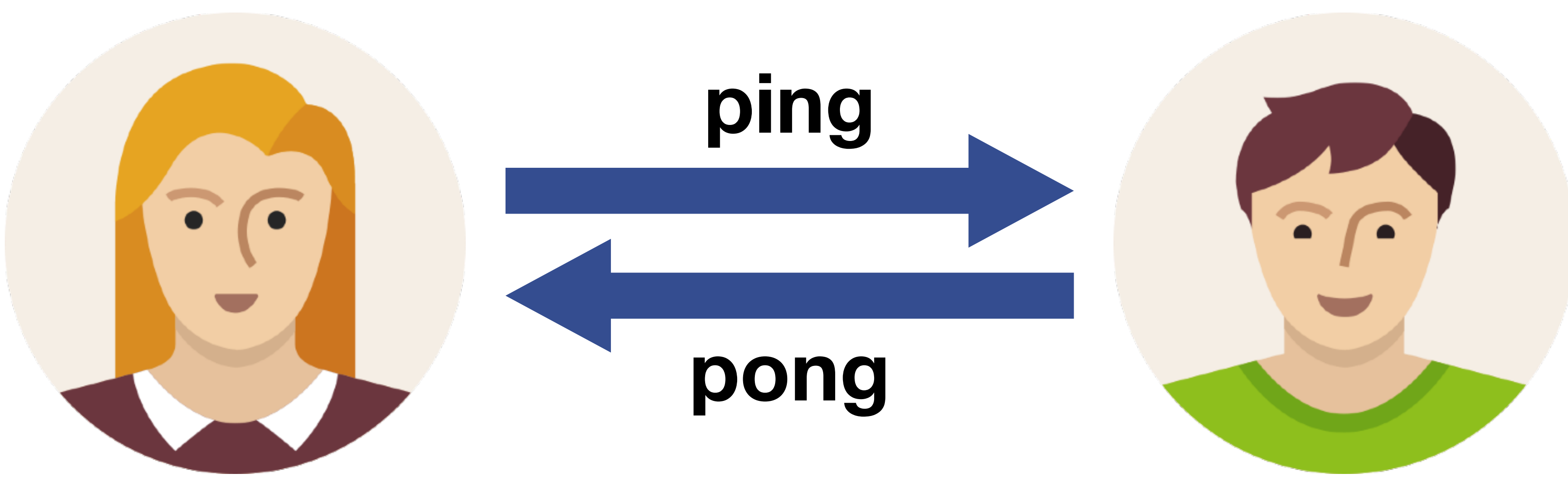
required state facts

action

produced state facts

```
          [ ]   --[   Start   ]-->  [ Out("ping"), State() ]
[ In("ping") ]  --[ Answer ]-->  [ Out("pong") ]
```

Tamarin



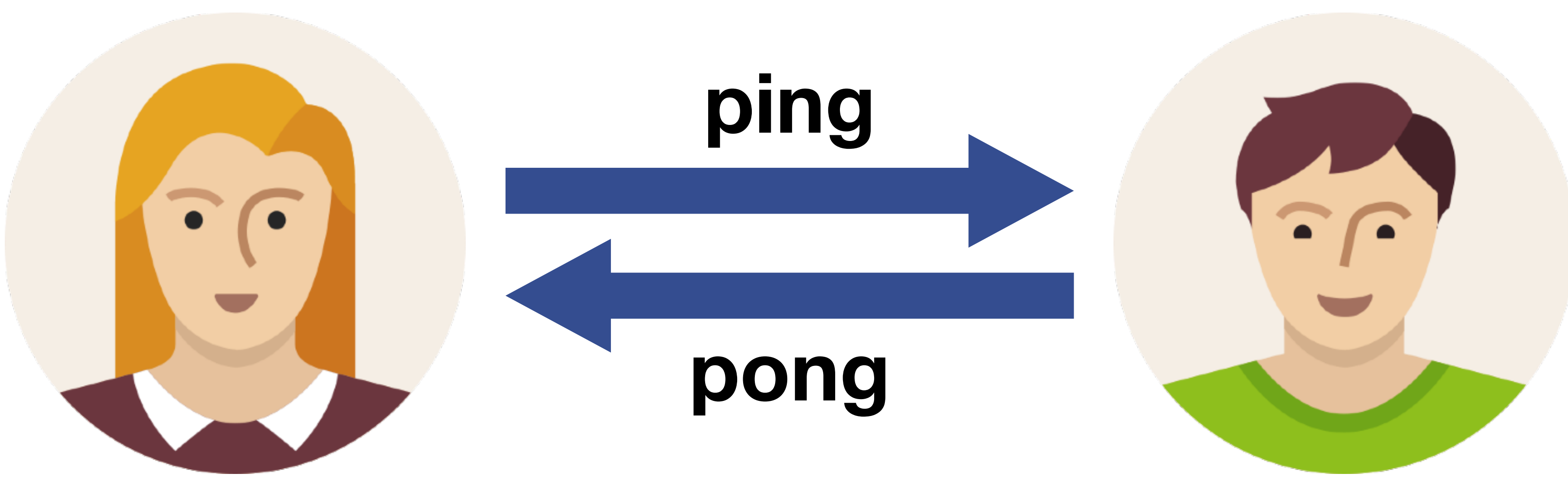
required state facts

action

produced state facts

```
                  [ ]  --[  Start   ]-->  [ Out("ping"), State() ]
        [ In("ping") ]  --[ Answer ]-->  [ Out("pong") ]
[ In("pong"), State() ]  --[  Finish  ]-->  [ ]
```

Tamarin



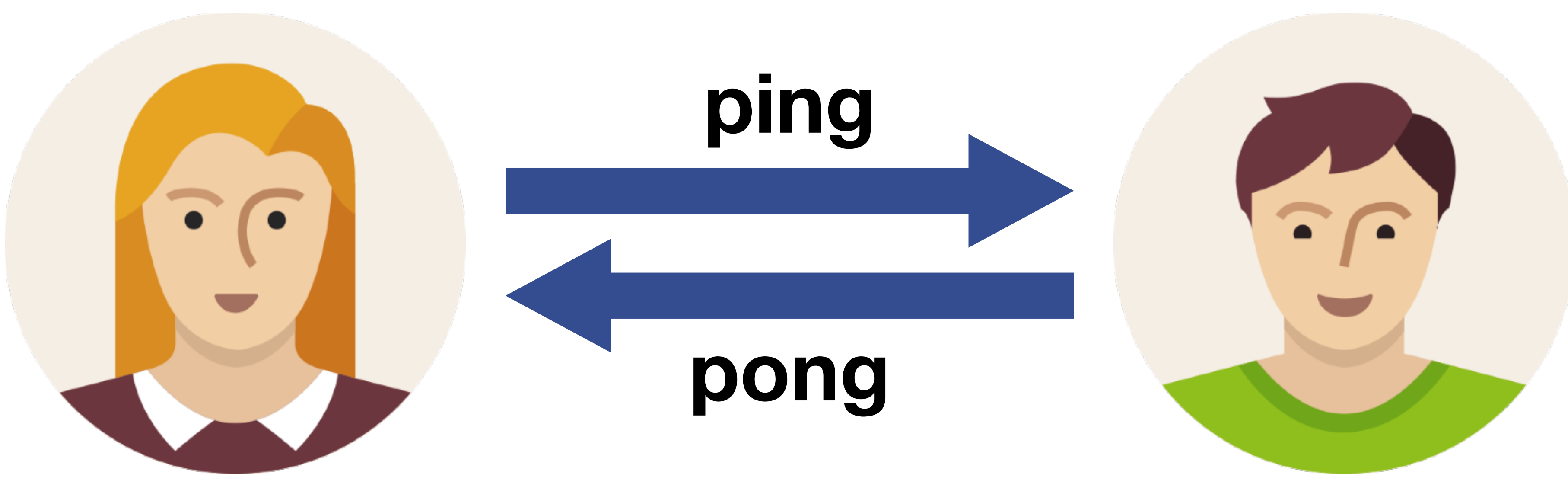
required state facts

action

produced state facts

```
                 [ ] --[  Start  ]--> [ Out("ping"), State() ]
        [ In("ping") ] --[ Answer ]--> [ Out("pong") ]
[ In("pong"), State() ] --[ Finish ]--> [ ]
```

Tamarin



required state facts

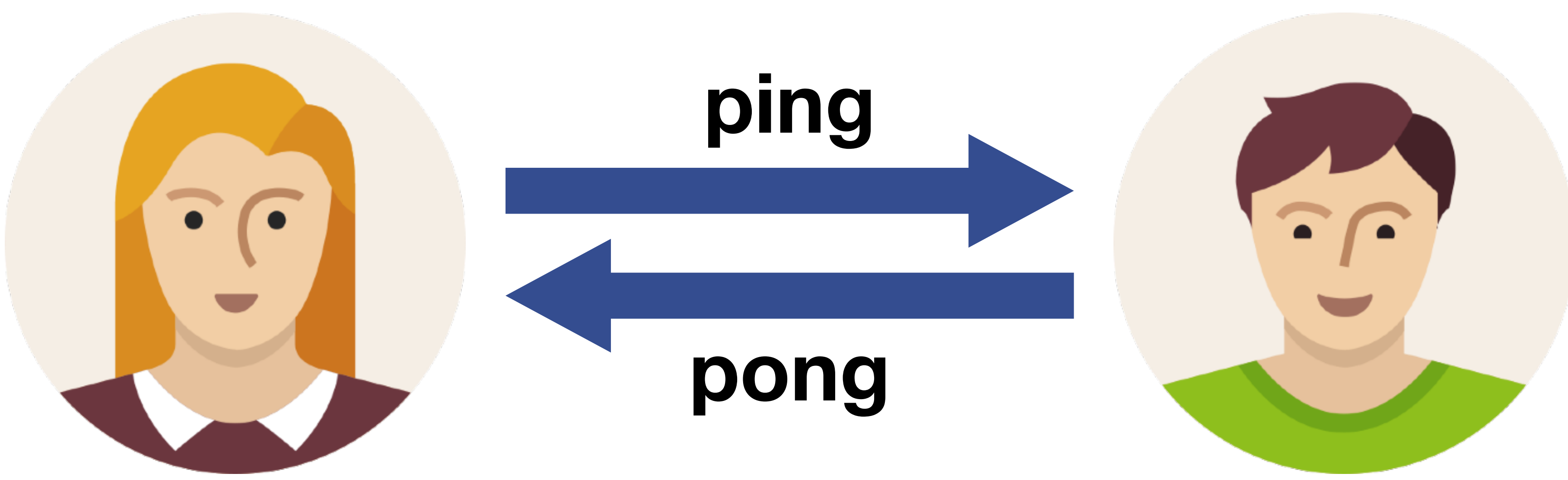
action

produced state facts

```
[ ]                         --[  Start  ]--> [ Out("ping"), State() ]
[ In("ping") ]              --[ Answer ]-->  [ Out("pong") ]
[ In("pong"), State() ]     --[  Finish ]--> [ ]
```

Start → Answer → Finish

Tamarin



required state facts

action

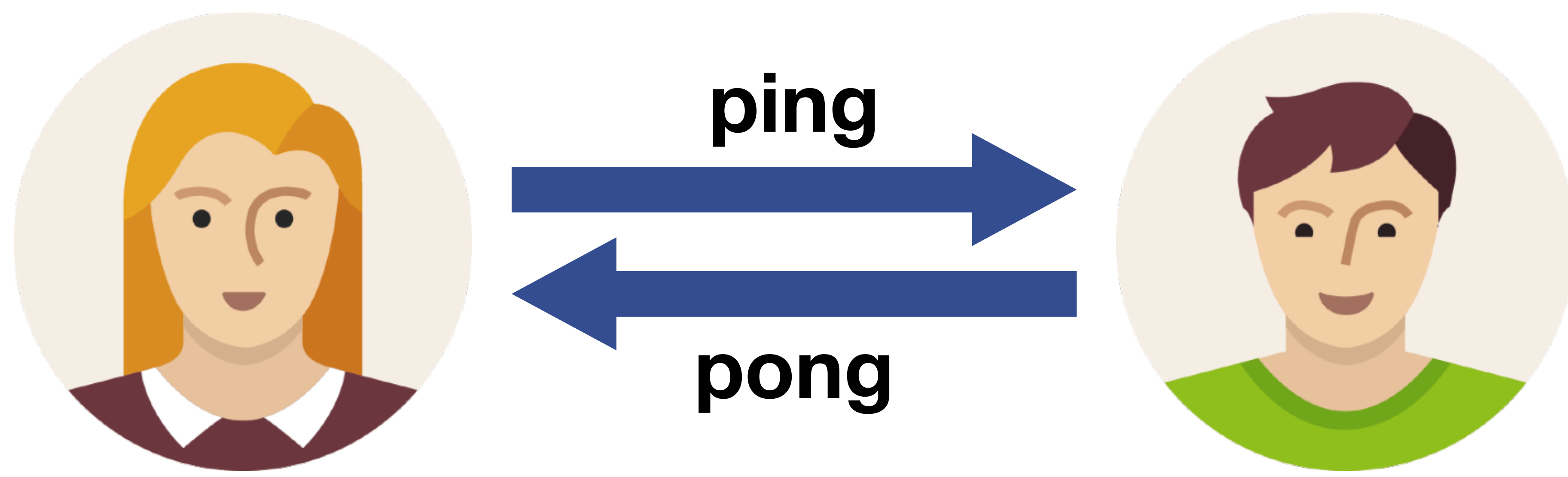
produced state facts

required state facts	action	produced state facts
[ ]	--[Start]-->	[Out("ping"), State()]
[In("ping")]	--[Answer]-->	[Out("pong")]
[In("pong"), State()]	--[Finish]-->	[ ]

Start → Answer → Finish

Start → Start → Answer → Start → Finish

Tamarin



required state facts

action

produced state facts

required state facts	action	produced state facts
[ ]	--[Start]-->	[Out("ping"), State()]
[In("ping")]	--[Answer]-->	[Out("pong")]
[In("pong"), State()]	--[Finish]-->	[ ]

$\forall$ Finish $\Rightarrow$ $\exists$ Answer

Start → Answer → Finish

Start → Start → Answer → Start → Finish

Tamarin

required state facts

action

produced state facts

```
[ ]                        --[  Start   ]-->  [ Out("ping"), State() ]
[ In("ping") ]             --[ Answer  ]-->  [ Out("pong") ]
[ In("pong"), State() ]    --[  Finish  ]-->  [ ]
```

$\forall$ Finish $\Rightarrow$ $\exists$ Answer

Start → Answer → Finish

Start → Start → Answer → Start → Finish

Tamarin

Symbolic Attacker

required state facts

action

produced state facts

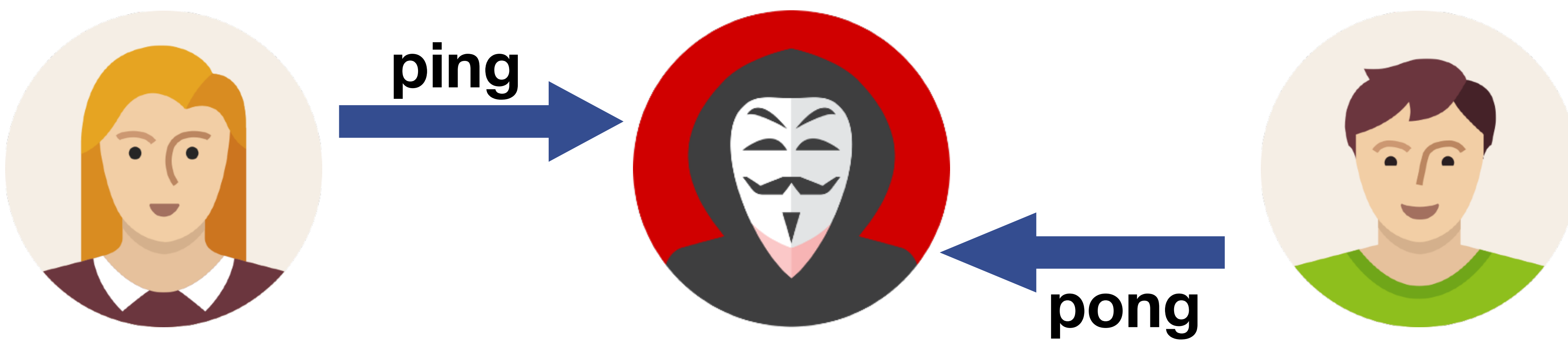
```
                   [ ] --[  Start  ]--> [ Out("ping"), State() ]
           [ In("ping") ] --[ Answer ]--> [ Out("pong") ]
[ In("pong"), State() ] --[  Finish ]--> [ ]
```

∀ Finish ⇒ ∃ Answer

Start → Answer → Finish

Start → Start → Answer → Start → Finish

Tamarin



Symbolic Attacker
- observe all Out(...)

required state facts | action | produced state facts

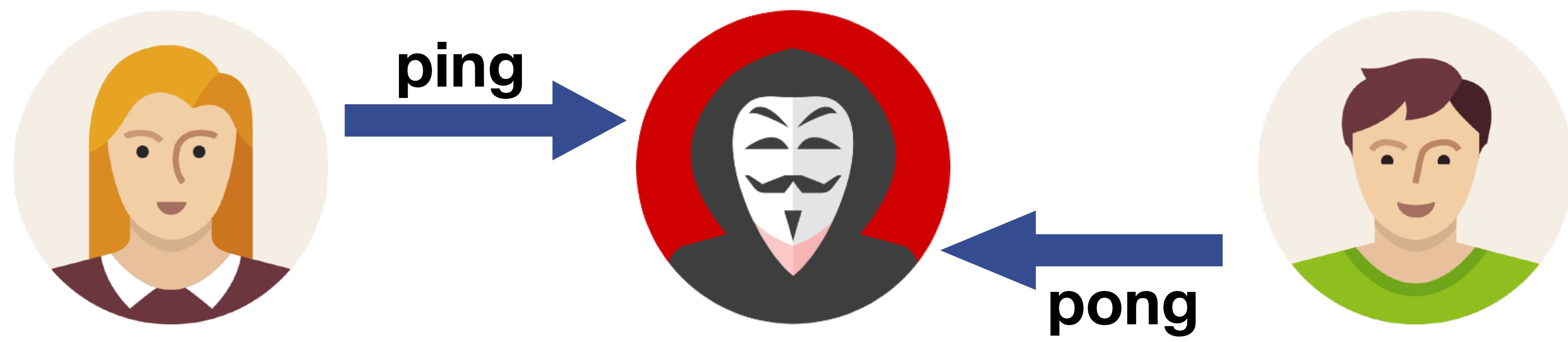
```
                   [ ] --[  Start  ]--> [ Out("ping"), State() ]
          [ In("ping") ] --[ Answer ]--> [ Out("pong") ]
[ In("pong"), State() ] --[  Finish ]--> [ ]
```

$\forall$ Finish $\Rightarrow$ $\exists$ Answer

Start → Answer → Finish

Start → Start → Answer → Start → Finish

Tamarin



Symbolic Attacker
- observe all Out(...)
- controls all In(...)

required state facts

action

produced state facts

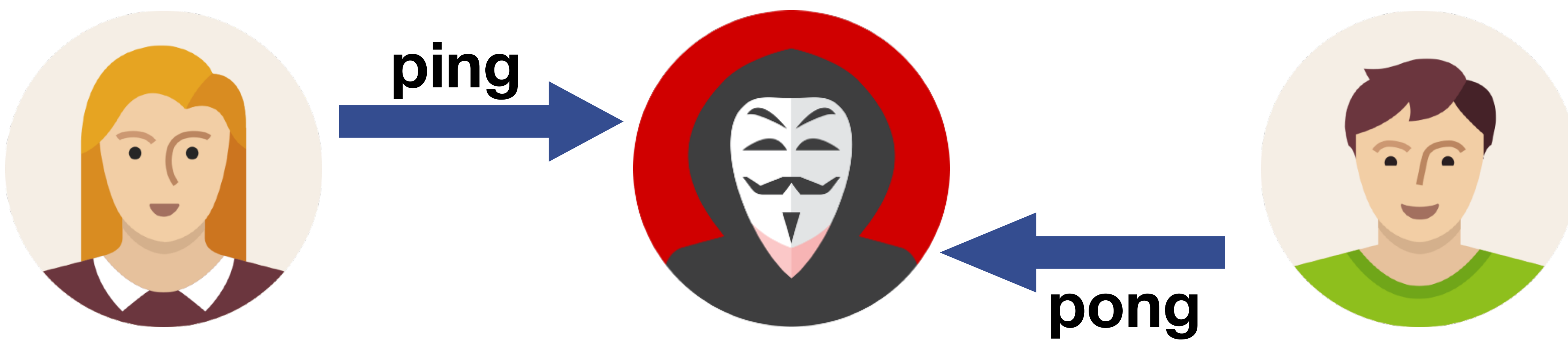
[ ]	--[Start]-->	[Out("ping"), State()]
[In("ping")]	--[Answer]-->	[Out("pong")]
[In("pong"), State()]	--[Finish]-->	[ ]

$\forall$ Finish $\Rightarrow$ $\exists$ Answer

Start → Answer → Finish

Start → Start → Answer → Start → Finish

Tamarin



Symbolic Attacker
- observe all Out(...)
- controls all In(...)
  - drop messages

required state facts

action

produced state facts

```
[ ]                        --[  Start   ]--> [ Out("ping"), State() ]
[ In("ping") ]             --[ Answer  ]--> [ Out("pong") ]
[ In("pong"), State() ]    --[  Finish  ]--> [ ]
```

$\forall$ Finish $\Rightarrow$ $\exists$ Answer

Start → Answer → Finish

Start → Start → Answer → Start → Finish

Tamarin

required state facts

action

produced state facts

```
[ ]                     --[ Start  ]--> [ Out("ping"), State() ]
[ In("ping") ]          --[ Answer ]--> [ Out("pong") ]
[ In("pong"), State() ] --[ Finish ]--> [ ]
```

Start → Answer → Finish

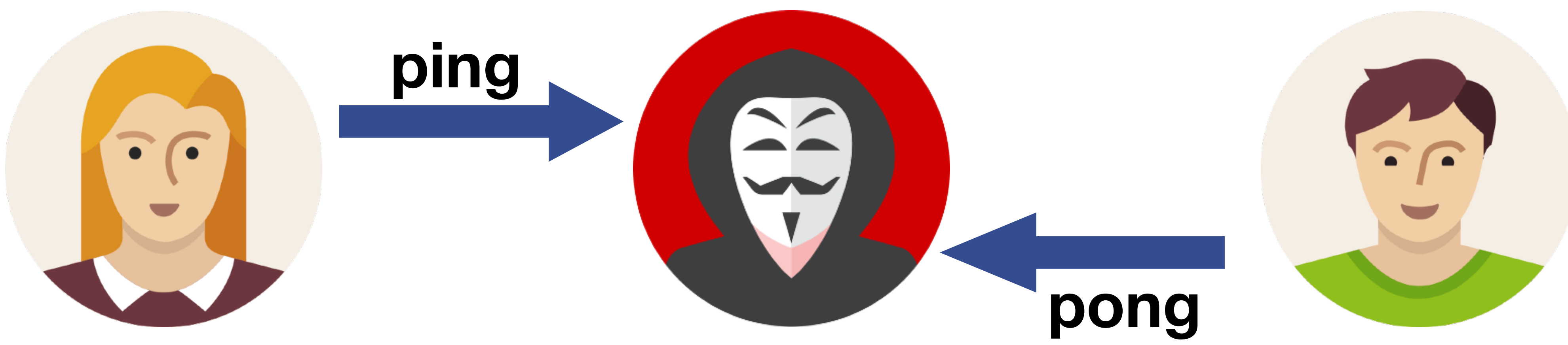
Start → Start → Answer → Start → Finish

Symbolic Attacker

- observe all Out(...)
- controls all In(...)
  - drop messages
  - send a "pong"

$\forall$ Finish $\Rightarrow$ $\exists$ Answer

Tamarin



required state facts | action | produced state facts

```
                [ ]  --[  Start  ]-->  [ Out("ping"), State() ]
      [ In("ping") ]  --[ Answer ]-->  [ Out("pong") ]
[ In("pong"), State() ]  --[ Finish ]-->  [ ]
```

Start → Answer → Finish

Start → Start → Answer → Start → Finish

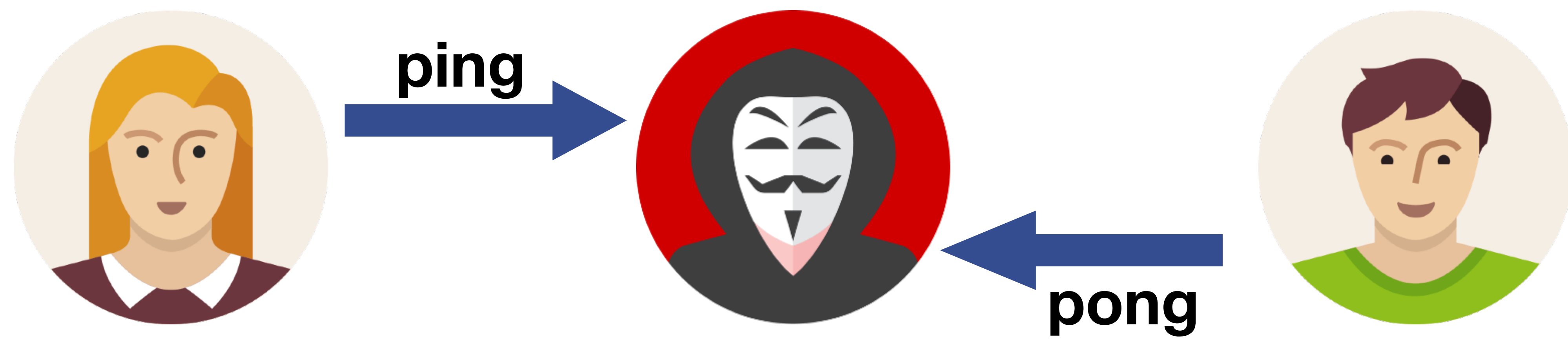
Symbolic Attacker

- observe all Out(...)
- controls all In(...)
  - drop messages
  - send a "pong"

$\forall$ Finish $\Rightarrow$ $\exists$ Answer

Start → Finish

attacker sends "pong"



required state facts

action

produced state facts

```
                  [ ]  --[  Start  ]-->  [ Out("ping"), State() ]
         [ In("ping") ] --[ Answer ]-->  [ Out("pong") ]
[ In("pong"), State() ] --[  Finish ]-->  [ ]
```

Start → Answer → Finish

Start → Start → Answer → Start → Finish

Symbolic Attacker

- observe all Out(...)
- controls all In(...)
  - drop messages
  - send a "pong"

$\forall$ Finish $\Rightarrow$ $\exists$ Answer

Start → Finish

attacker sends "pong"

(Unbounded) Data Structures

(Unbounded) Data Structures

- Trees

(Unbounded) Data Structures

- Trees
  - TreeKEM, Merkle-Trees, ...

(Unbounded) Data Structures

- Trees
  - TreeKEM, Merkle-Trees, ...
  - prove invariant over all sub-trees

(Unbounded) Data Structures

- Trees
  - TreeKEM, Merkle-Trees, ...
  - prove invariant over all sub-trees
- Chains

(Unbounded) Data Structures

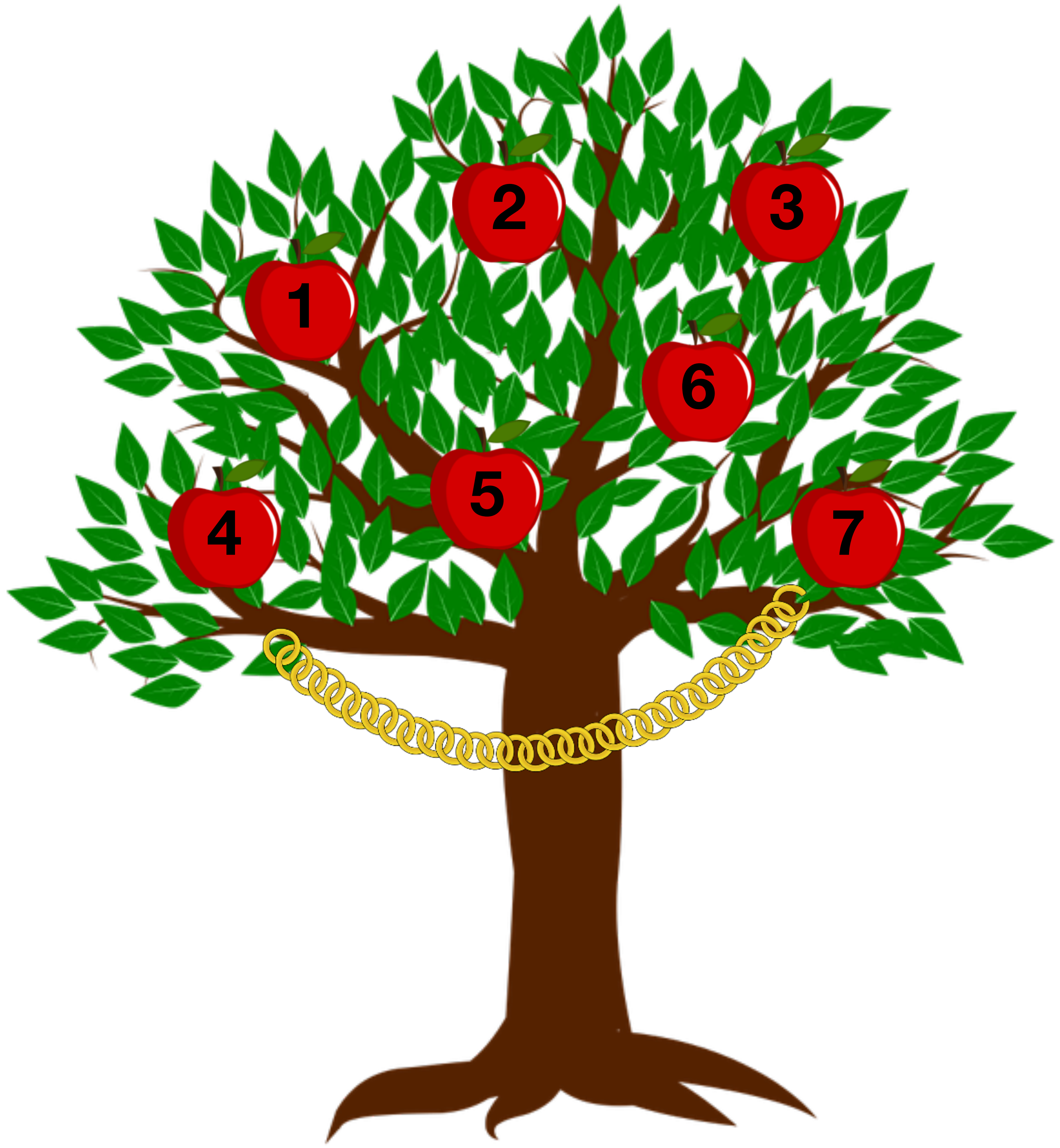
- Trees
  - TreeKEM, Merkle-Trees, ...
  - prove invariant over all sub-trees
- Chains
  - Hash-Chains, Blockchains, ...

(Unbounded) Data Structures

- Trees
  - TreeKEM, Merkle-Trees, ...
  - prove invariant over all sub-trees
- Chains
  - Hash-Chains, Blockchains, ...
  - blockchain: is block X in the chain?

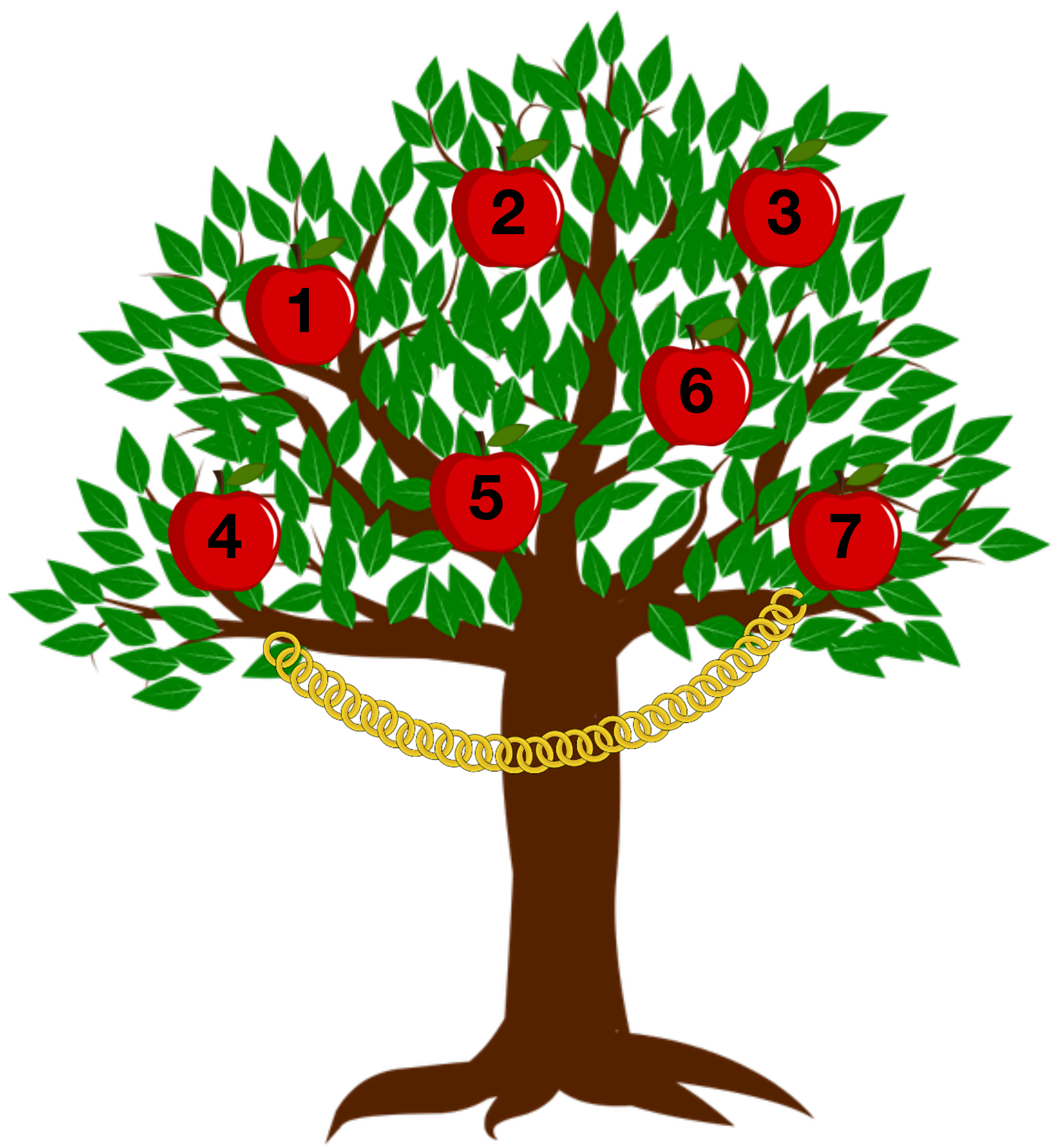
(Unbounded) Data Structures

- Trees
  - TreeKEM, Merkle-Trees, ...
  - prove invariant over all sub-trees
- Chains
  - Hash-Chains, Blockchains, ...
  - blockchain: is block X in the chain?
- Counters



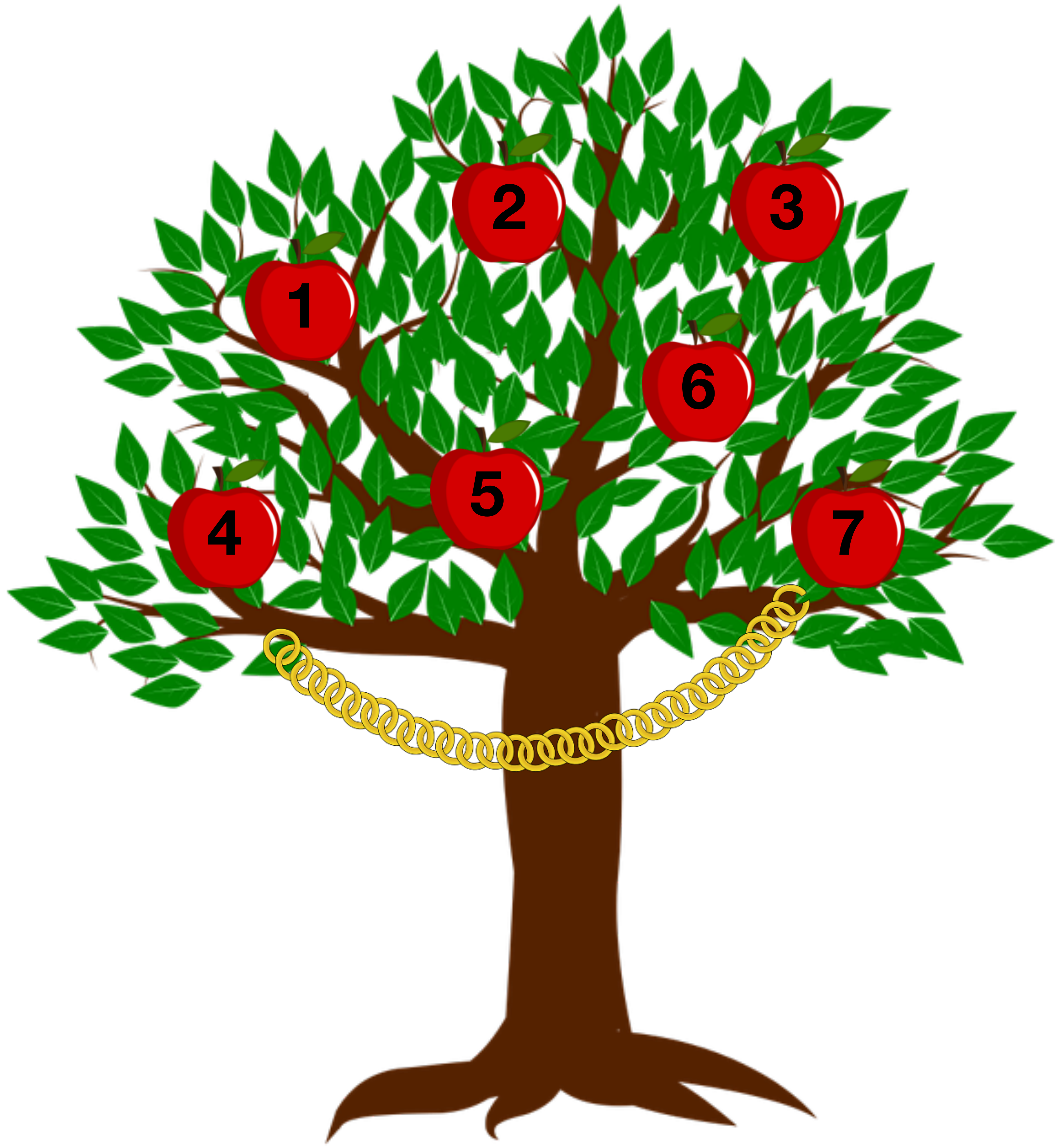
(Unbounded) Data Structures

- Trees
  - TreeKEM, Merkle-Trees, ...
  - prove invariant over all sub-trees
- Chains
  - Hash-Chains, Blockchains, ...
  - blockchain: is block X in the chain?
- Counters
  - WPA-2, 5G, ...



(Unbounded) Data Structures

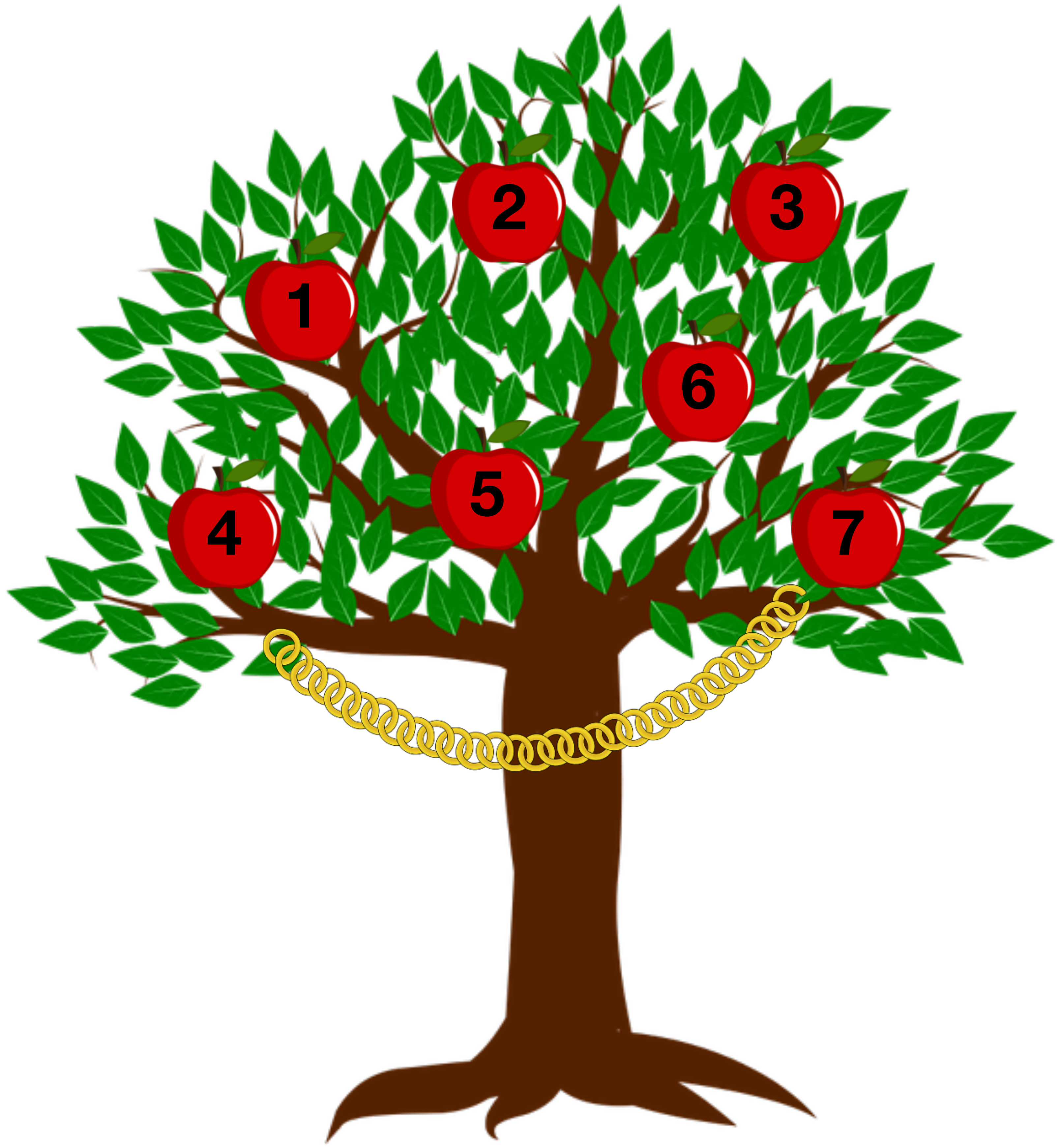
- Trees
  - TreeKEM, Merkle-Trees, ...
  - prove invariant over all sub-trees
- Chains
  - Hash-Chains, Blockchains, ...
  - blockchain: is block X in the chain?
- Counters
  - WPA-2, 5G, ...
  - is a < b ?



(Unbounded) Data Structures

- Trees
  - TreeKEM, Merkle-Trees, ...
  - prove invariant over all sub-trees
- Chains
  - Hash-Chains, Blockchains, ...
  - blockchain: is block X in the chain?
- Counters
  - WPA-2, 5G, ...
  - is a < b ?

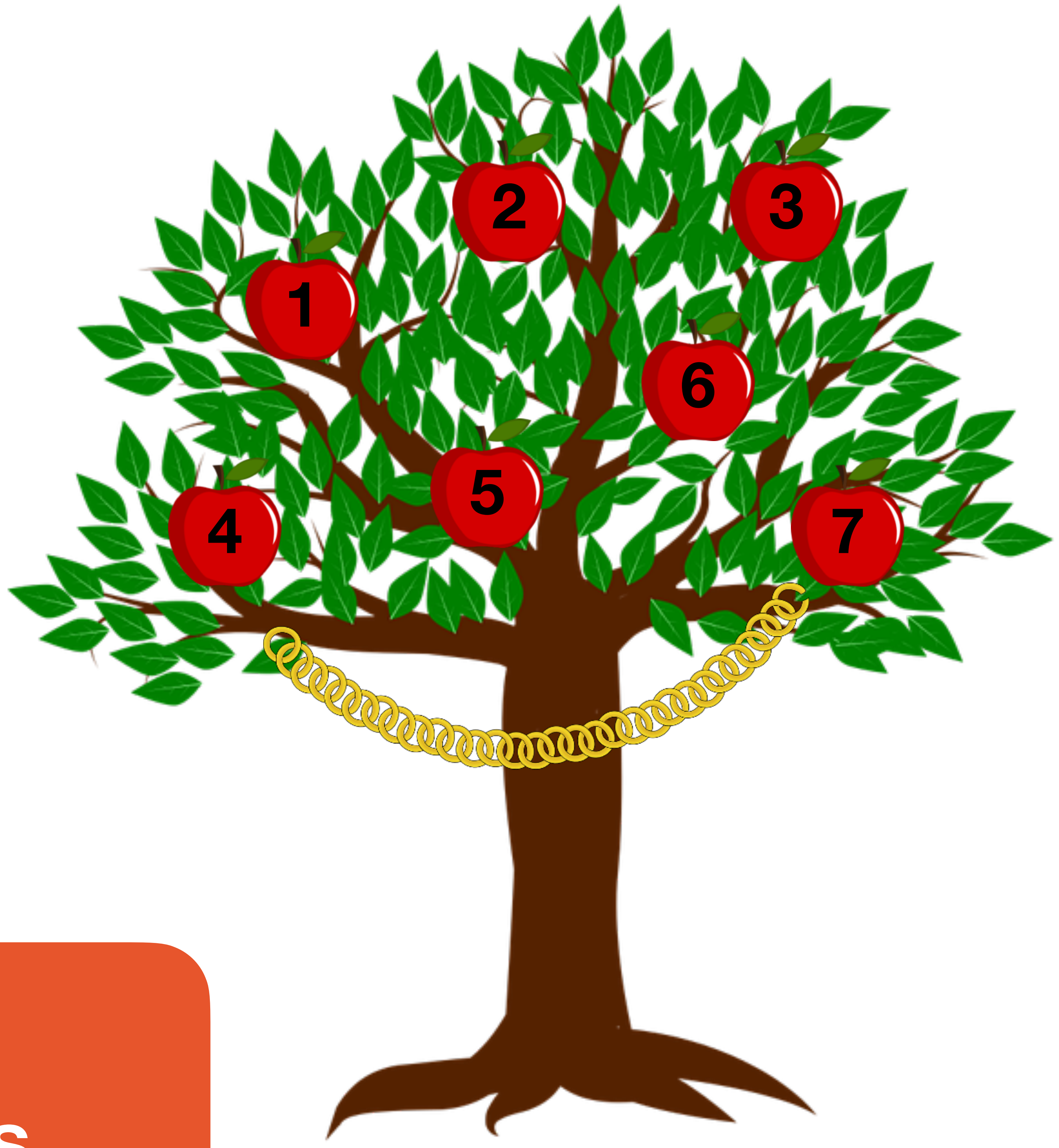
→ **Divergence**



(Unbounded) Data Structures

- Trees
    - TreeKEM, Merkle-Trees, ...
    - prove invariant over all sub-trees
- Chains
    - Hash-Chains, Blockchains, ...
    - blockchain: is block X in the chain?
- Counters
    - WPA-2, 5G, ...
    - is a < b ?

→ **Divergence**



Solution: Subterms

Our Contribution

Our Contribution

Modeling

Our Contribution

Modeling

$\sqsubset$

- Subterm-Predicate "$\sqsubset$"
  - we can now state $x \sqsubset h(h(x))$

Our Contribution

Modeling

- Subterm-Predicate "$\sqsubset$"
  - we can now state $x \sqsubset h(h(x))$
- New Tamarin-Result-Type
  - "we don't know"
  - does $x \sqsubset x \oplus y$ hold ?

Our Contribution

Modeling

- Subterm-Predicate "$\sqsubset$"
  - we can now state $x \sqsubset h(h(x))$
- New Tamarin-Result-Type
  - "we don't know"
  - does $x \sqsubset x \oplus y$ hold ?
- Natural Numbers
  - adding a "+"-operator

Our Contribution

Modeling

- Subterm-Predicate "$\sqsubset$"
  - we can now state $x \sqsubset h(h(x))$
- New Tamarin-Result-Type
  - "we don't know"
  - does $x \sqsubset x \oplus y$ hold ?
- Natural Numbers
  - adding a "+"-operator

Proof Techniques

"under the hood"

- Algorithm for Numbers
- Monotonicity
- Fresh Ordering

Our Contribution

Modeling

- Subterm-Predicate "$\sqsubset$"
  - we can now state $x \sqsubset h(h(x))$
- New Tamarin-Result-Type
  - "we don't know"
  - does $x \sqsubset x \oplus y$ hold ?
- Natural Numbers
  - adding a "+"-operator

Proof Techniques

"under the hood"

- Algorithm for Numbers
- Monotonicity
- Fresh Ordering

Case Studies

- New Proofs
- Application to Old Proofs

(small) Numbers

(small) Numbers

$(a \mathbin{+\!\!\!+} b) \mathbin{+\!\!\!+} c$
$= a \mathbin{+\!\!\!+} (b \mathbin{+\!\!\!+} c)$

$a \mathbin{+\!\!\!+} b$
$= b \mathbin{+\!\!\!+} a$

- well studied: associative and commutative (AC) operator $\mathbin{+\!\!\!+}$

(small) Numbers

$(a \mathbin{+\!\!\!+} b) \mathbin{+\!\!\!+} c$
$= a \mathbin{+\!\!\!+} (b \mathbin{+\!\!\!+} c)$

$a \mathbin{+\!\!\!+} b$
$= b \mathbin{+\!\!\!+} a$

- well studied: associative and commutative (AC) operator $\mathbin{+\!\!\!+}$
- used as multiset: $a \mathbin{+\!\!\!+} b \mathbin{+\!\!\!+} b = \{a, b, b\}$

(small) Numbers

$(a \mathbin{+\!\!\!+} b) \mathbin{+\!\!\!+} c$
$= a \mathbin{+\!\!\!+} (b \mathbin{+\!\!\!+} c)$

$a \mathbin{+\!\!\!+} b$
$= b \mathbin{+\!\!\!+} a$

- well studied: associative and commutative (AC) operator $\mathbin{+\!\!\!+}$
- used as multiset: $a \mathbin{+\!\!\!+} b \mathbin{+\!\!\!+} b = \{a, b, b\}$
- used for counting: one $\mathbin{+\!\!\!+}$ one $\mathbin{+\!\!\!+}$ one = 3

(small) Numbers

$(a \mathbin{+\!\!+} b) \mathbin{+\!\!+} c$
$= a \mathbin{+\!\!+} (b \mathbin{+\!\!+} c)$

$a \mathbin{+\!\!+} b$
$= b \mathbin{+\!\!+} a$

- well studied: associative and commutative (AC) operator $\mathbin{+\!\!+}$
- used as multiset: $a \mathbin{+\!\!+} b \mathbin{+\!\!+} b = \{a, b, b\}$
- used for counting: one $\mathbin{+\!\!+}$ one $\mathbin{+\!\!+}$ one = 3
- our improvement:
  - type system, dedicated operator +

(small) Numbers

$(a \mathbin{+\!\!+} b) \mathbin{+\!\!+} c$
$= a \mathbin{+\!\!+} (b \mathbin{+\!\!+} c)$

$a \mathbin{+\!\!+} b$
$= b \mathbin{+\!\!+} a$

- well studied: associative and commutative (AC) operator $\mathbin{+\!\!+}$
- used as multiset: $a \mathbin{+\!\!+} b \mathbin{+\!\!+} b = \{a, b, b\}$
- used for counting: one $\mathbin{+\!\!+}$ one $\mathbin{+\!\!+}$ one = 3
- our improvement:
  - type system, dedicated operator +
  - comperator: $a < b \quad \Leftrightarrow \quad \exists\, x.\ a + x = b$

(small) Numbers

$(a \mathbin{+\!\!+} b) \mathbin{+\!\!+} c$
$= a \mathbin{+\!\!+} (b \mathbin{+\!\!+} c)$

$a \mathbin{+\!\!+} b$
$= b \mathbin{+\!\!+} a$

- well studied: associative and commutative (AC) operator $\mathbin{+\!\!+}$
- used as multiset: $a \mathbin{+\!\!+} b \mathbin{+\!\!+} b = \{a, b, b\}$
- used for counting: $\text{one} \mathbin{+\!\!+} \text{one} \mathbin{+\!\!+} \text{one} = 3$
- our improvement:
  - type system, dedicated operator $+$
  - comperator: $a < b \quad \Leftrightarrow \quad \exists\, x.\ a + x = b$
  - dedicated algorithm: $a < b < a{+}2 \quad \Rightarrow \quad b = a{+}1$

Dedicated Proof Techniques

Dedicated Proof Techniques

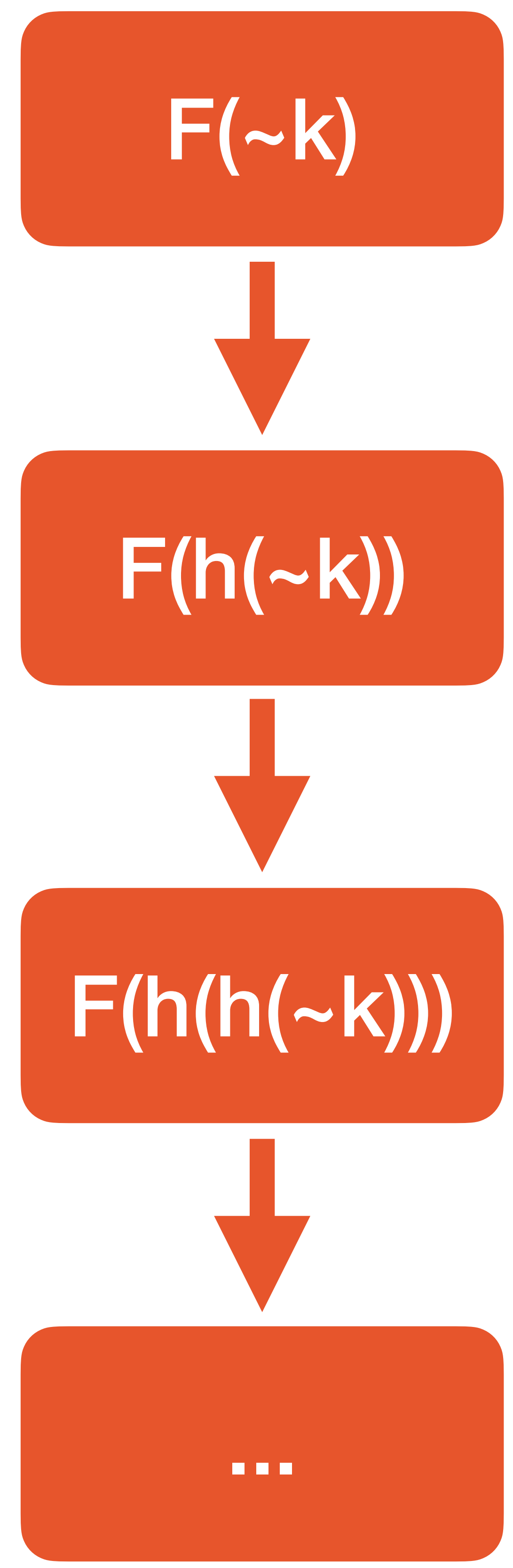
Monotonicity

F(~k) → F(h(~k)) → F(h(h(~k))) → ...

Dedicated Proof Techniques

Monotonicity

- F(s)@i,   F(t)@j



Dedicated Proof Techniques

Monotonicity

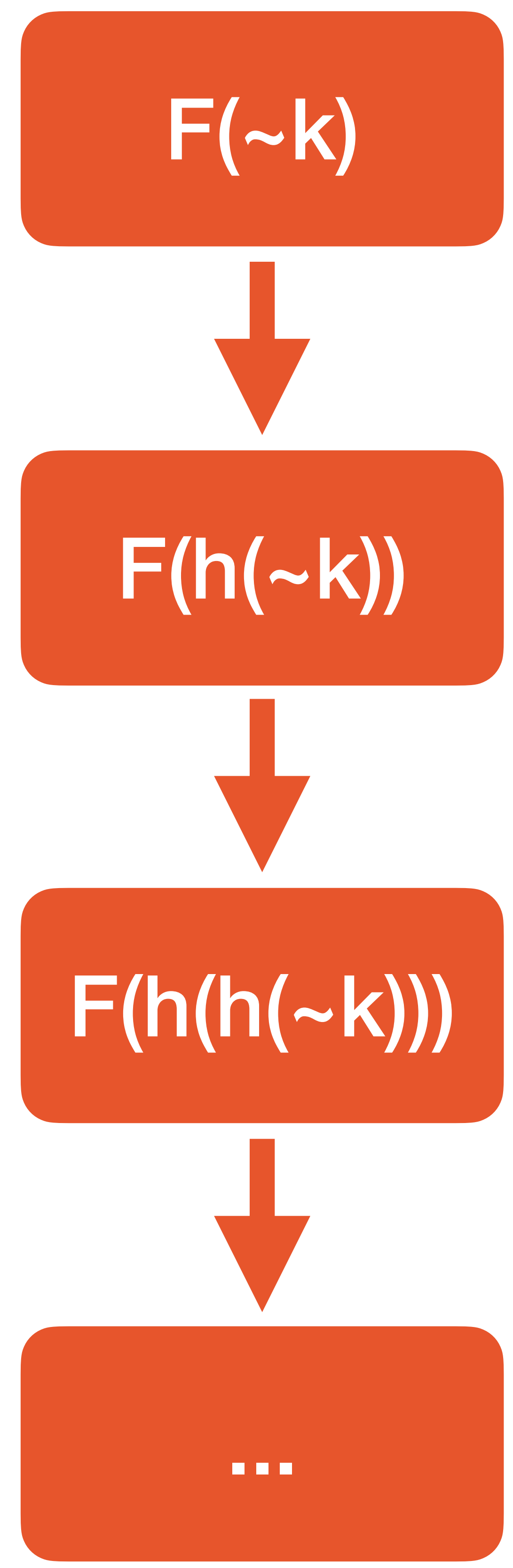
- F(s)@i,   F(t)@j
  - $s \sqsubset t \Rightarrow i < j$

$$F(\sim k) \rightarrow F(h(\sim k)) \rightarrow F(h(h(\sim k))) \rightarrow \ldots$$

Dedicated Proof Techniques

Monotonicity

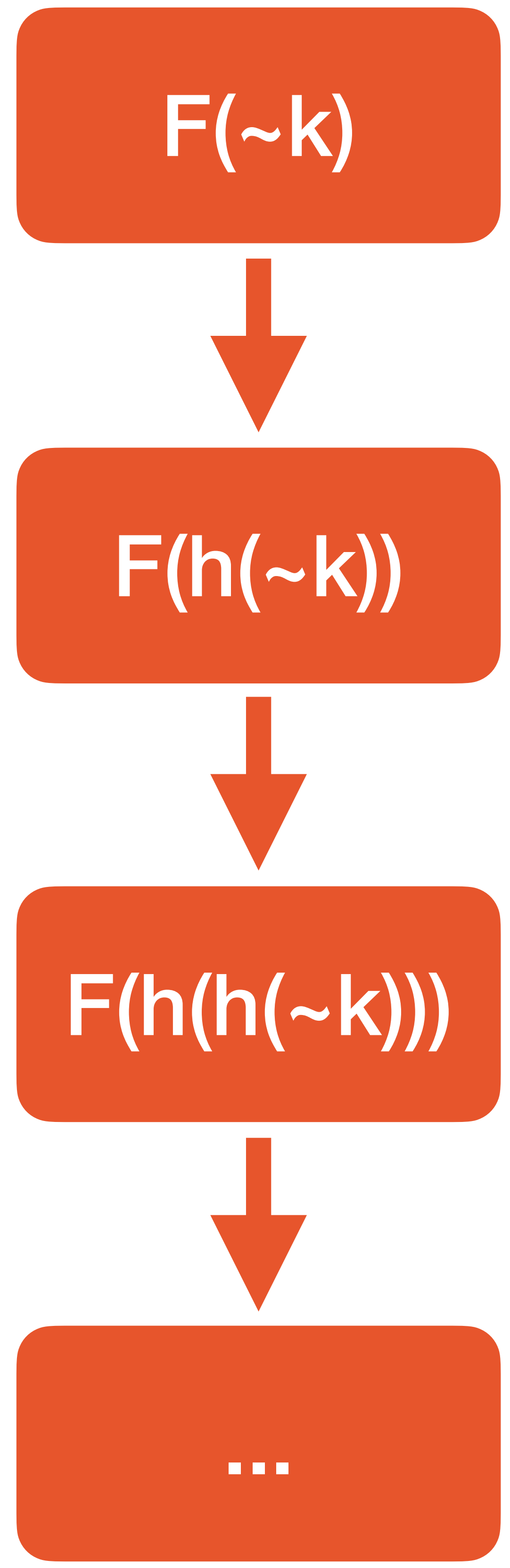
- $F(\ s\ )@i, \quad F(\ t\ )@j$
  - $s \sqsubset t \Rightarrow i < j$
  - $s = t \Rightarrow i = j$



Dedicated Proof Techniques

Monotonicity

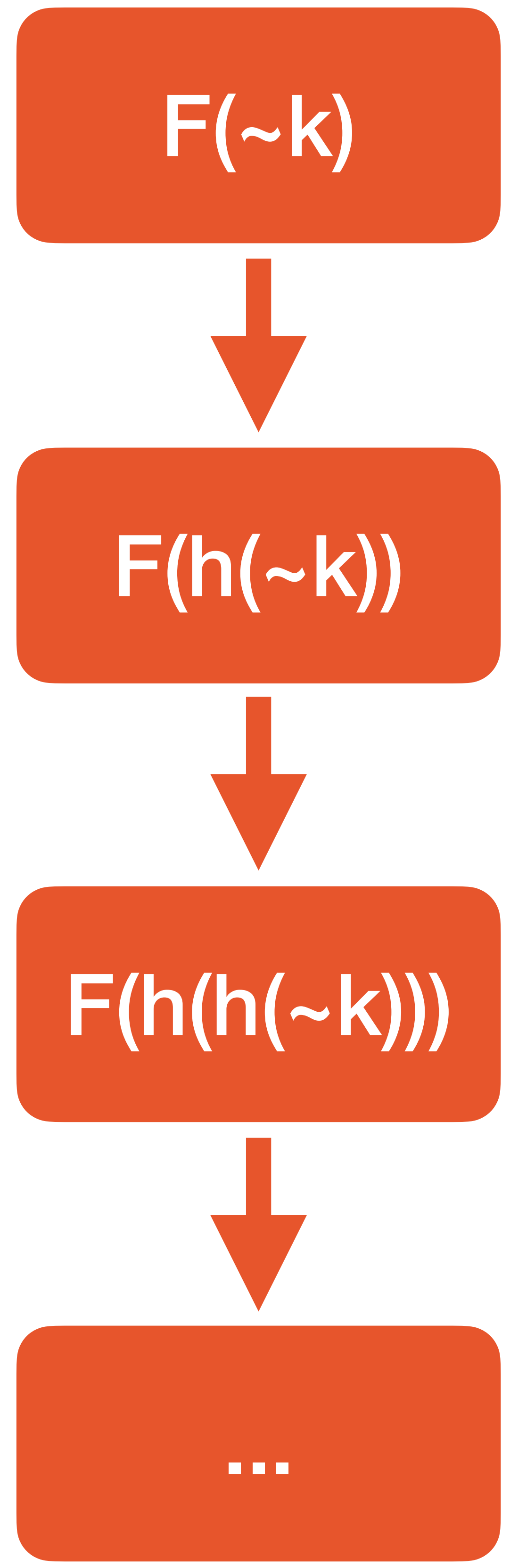
- $F(\ s\ )@i, \quad F(\ t\ )@j$
  - $s \sqsubset t \Rightarrow i < j$
  - $s = t \Rightarrow i = j$
  - $i \neq j \Rightarrow s \neq t$



Dedicated Proof Techniques

Monotonicity

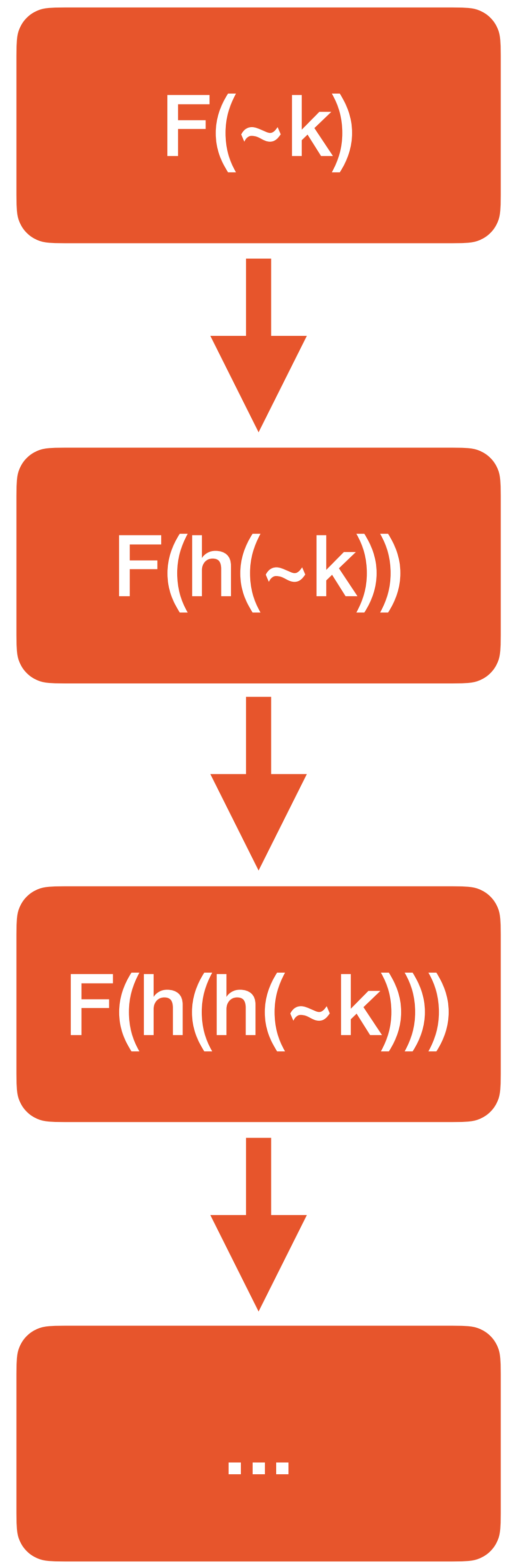
- F(s)@i,   F(t)@j
  - $s \sqsubset t \Rightarrow i < j$
  - $s = t \Rightarrow i = j$
  - $i \neq j \Rightarrow s \neq t$
  - some more ...



Dedicated Proof Techniques

Monotonicity

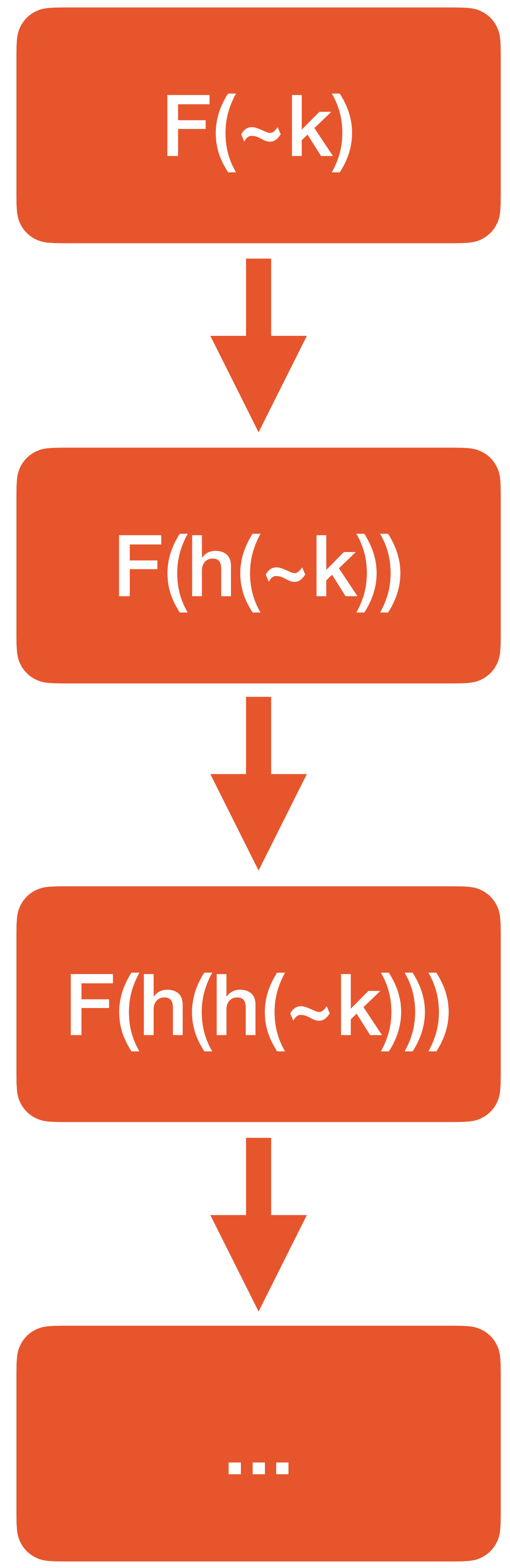
- F(s)@i,   F(t)@j
  - $s \sqsubset t \Rightarrow i < j$
  - $s = t \Rightarrow i = j$
  - $i \neq j \Rightarrow s \neq t$
  - some more ...



- 10x speed-up of WPA-2 proof

Dedicated Proof Techniques

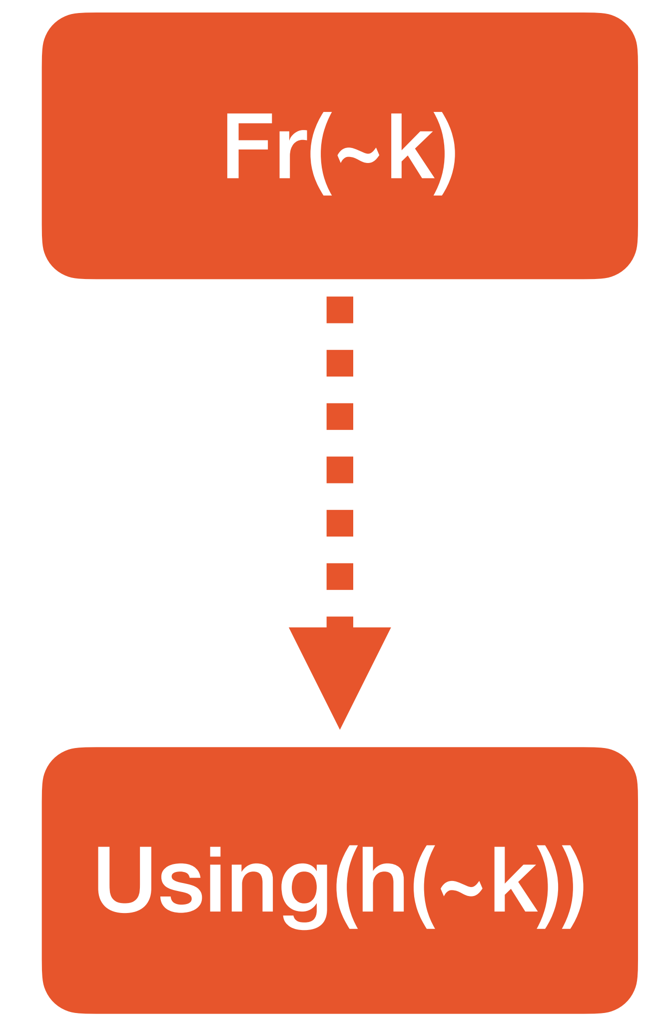
Monotonicity



- F(s)@i,   F(t)@j
  - $s \sqsubset t \Rightarrow i < j$
  - $s = t \Rightarrow i = j$
  - $i \neq j \Rightarrow s \neq t$
  - some more ...

- 10x speed-up of WPA-2 proof

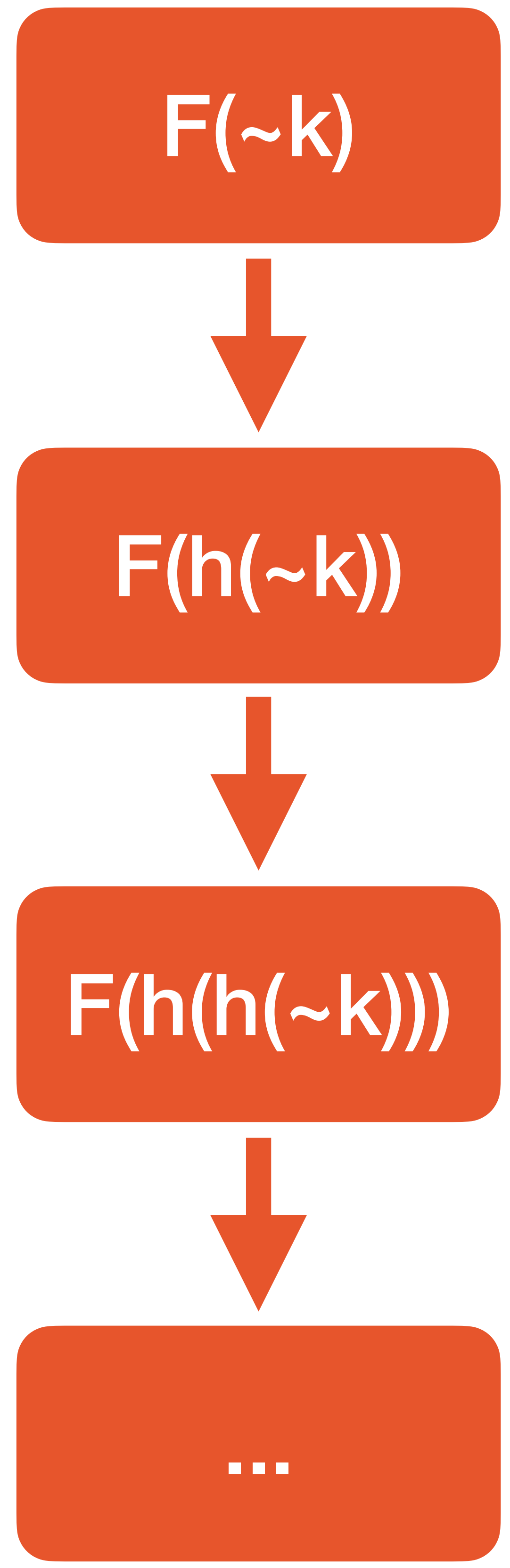
Fresh Order



Dedicated Proof Techniques

Monotonicity

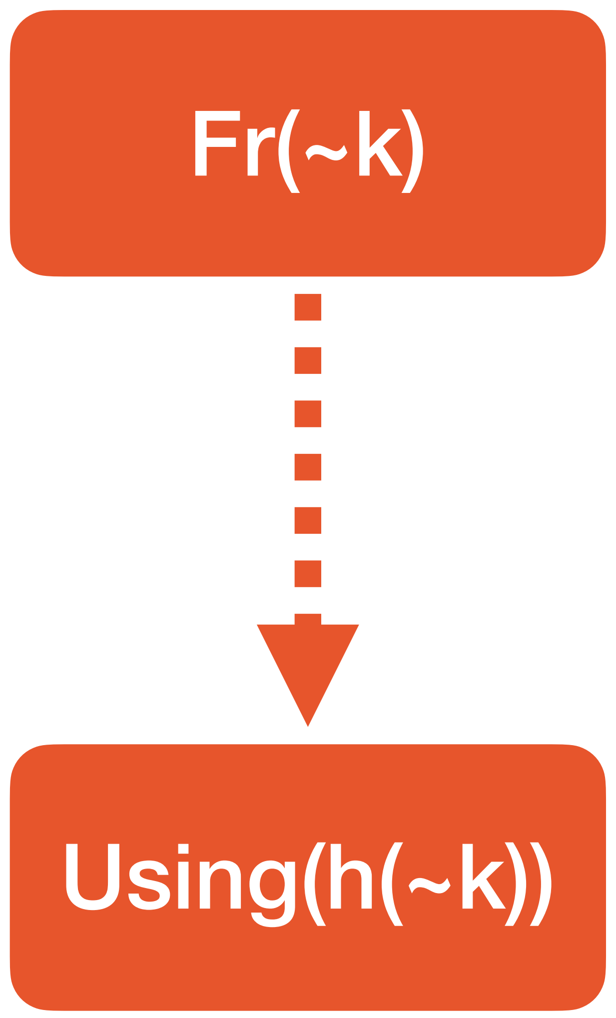
- F(s)@i,   F(t)@j
  - $s \sqsubset t \Rightarrow i < j$
  - $s = t \Rightarrow i = j$
  - $i \neq j \Rightarrow s \neq t$
  - some more ...



- 10x speed-up of WPA-2 proof

Fresh Order

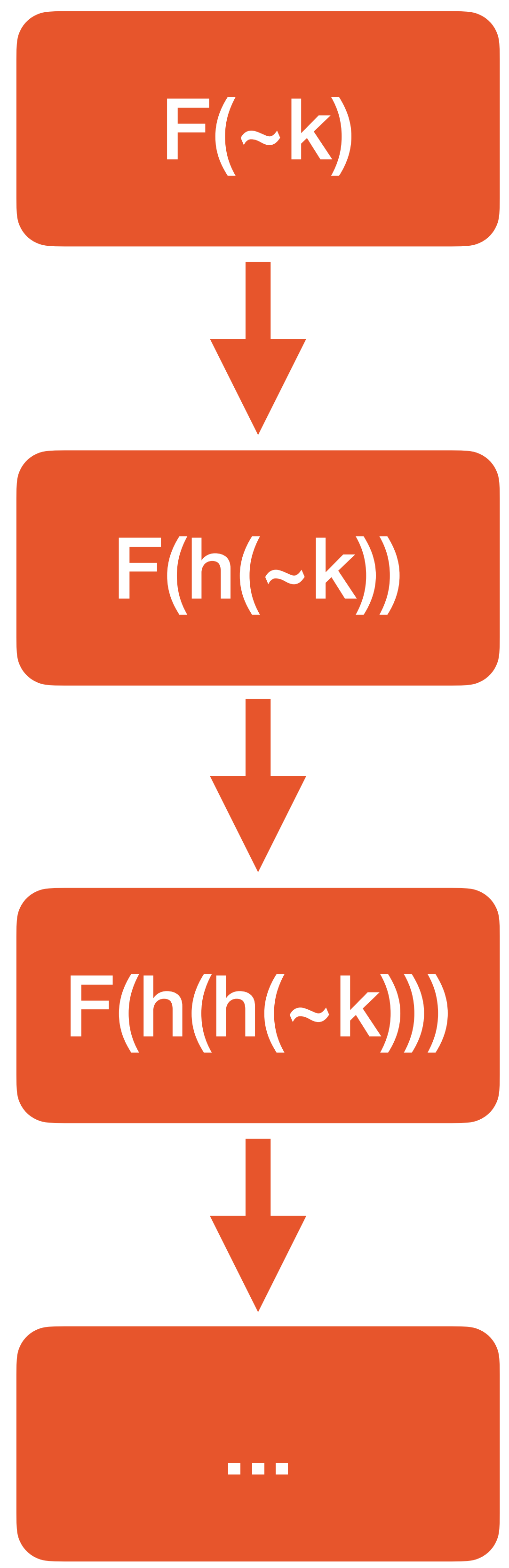
- time-ordering
  Fr(~k) < Using(~k)



Dedicated Proof Techniques

Monotonicity

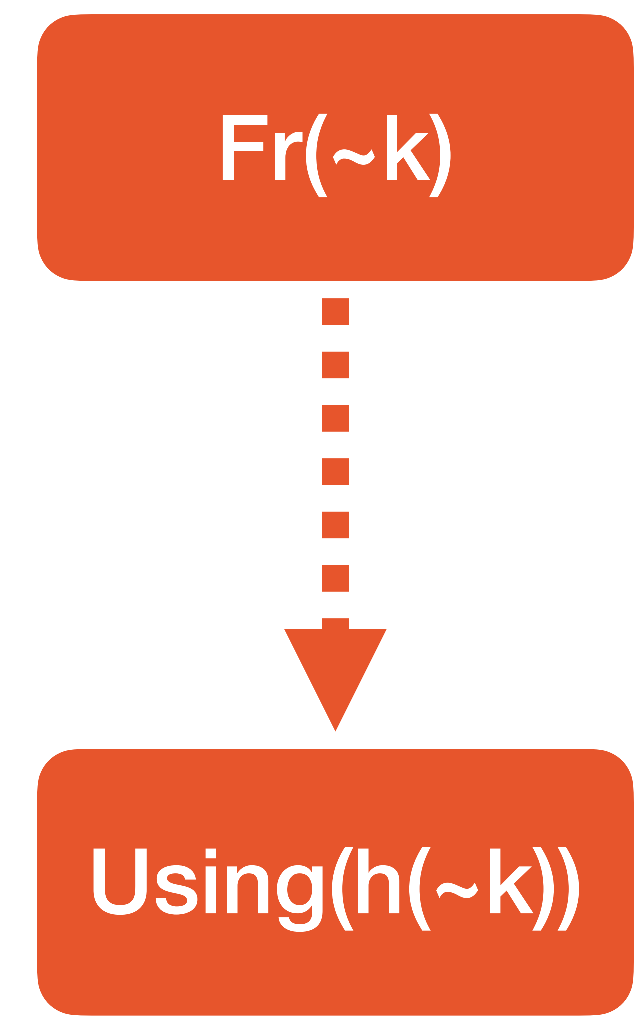
- F(s)@i,   F(t)@j
  - $s \sqsubset t \Rightarrow i < j$
  - $s = t \Rightarrow i = j$
  - $i \neq j \Rightarrow s \neq t$
  - some more ...



- 10x speed-up of WPA-2 proof

Fresh Order

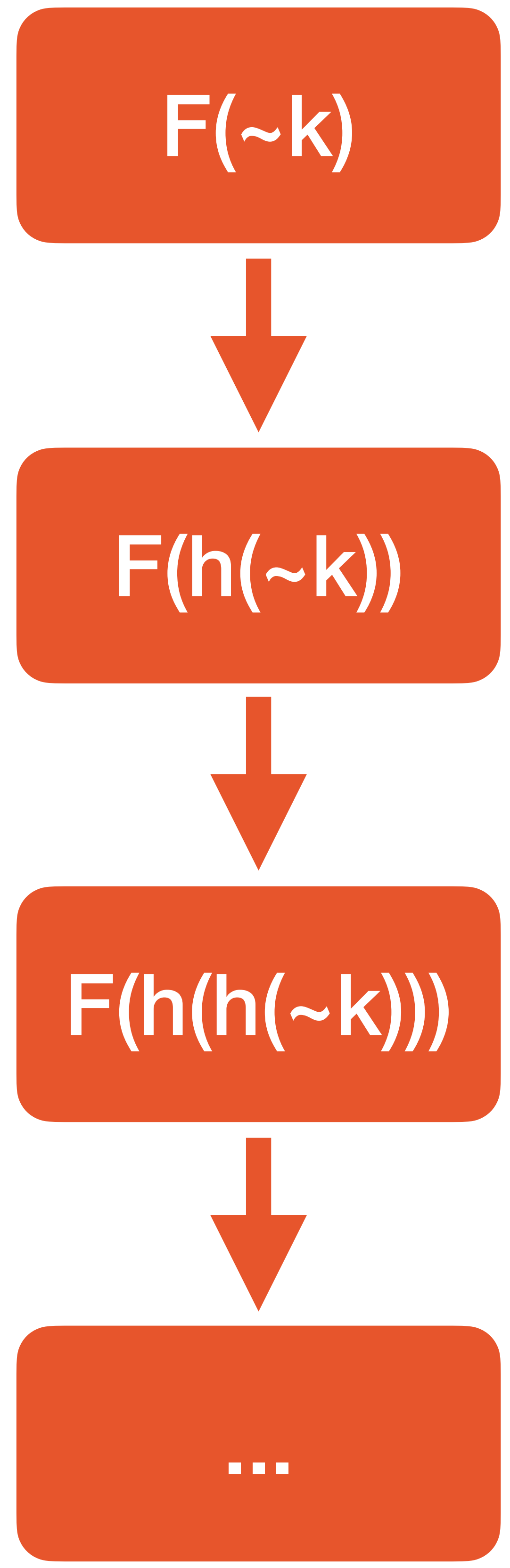
- time-ordering
  Fr(~k) < Using(~k)
- great with $\sqsubset$



Dedicated Proof Techniques

Monotonicity

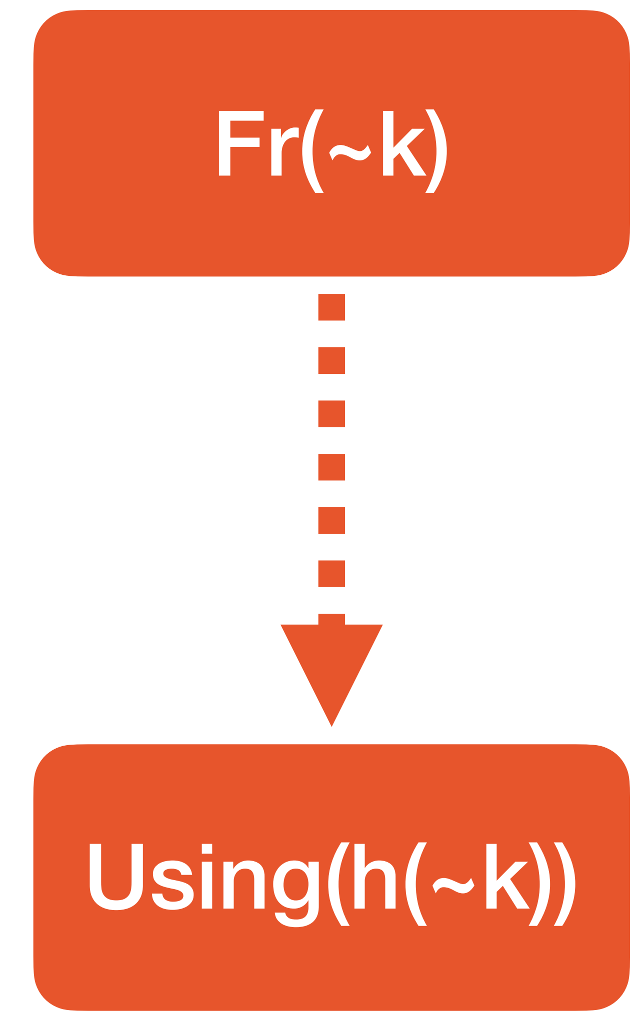
- F(s)@i,   F(t)@j
  - $s \sqsubset t \Rightarrow i < j$
  - $s = t \Rightarrow i = j$
  - $i \neq j \Rightarrow s \neq t$
  - some more ...



- 10x speed-up of WPA-2 proof

Fresh Order

- time-ordering
  Fr(~k) < Using(~k)
- great with $\sqsubset$



- 30x speed-up of CH'07 RFID proof

Applied to Existing Models

Protocol	Runtime	Helper-Lemmas	Why is it faster?
WPA-2	1:20h → 7min	74 → 73	monotonicity (of counters)
5G	8min → 2min	7 → 6	our number system
YubiKey	20s → 1s	4 → 3	our number system
PKCS#11	1min → 10s	4 → 0	each single improvement
CH'07 RFID	50min → 2min	0 → 0	fresh order

Applied to Existing Models

Protocol	Runtime	Helper-Lemmas	Why is it faster?
WPA-2	1:20h → 7min	74 → 73	monotonicity (of counters)
5G	8min → 2min	7 → 6	our number system
YubiKey	20s → 1s	4 → 3	our number system
PKCS#11	1min → 10s	4 → 0	each single improvement
CH'07 RFID	50min → 2min	0 → 0	fresh order

Applied to Existing Models

Protocol	Runtime	Helper-Lemmas	Why is it faster?
WPA-2	1:20h → 7min	74 → 73	monotonicity (of counters)
5G	8min → 2min	7 → 6	our number system
YubiKey	20s → 1s	4 → 3	our number system
PKCS#11	1min → 10s	4 → 0	each single improvement
CH'07 RFID	50min → 2min	0 → 0	fresh order

Applied to Existing Models

Protocol	Runtime	Helper-Lemmas	Why is it faster?
WPA-2	1:20h → 7min	74 → 73	monotonicity (of counters)
5G	8min → 2min	7 → 6	our number system
YubiKey	20s → 1s	4 → 3	our number system
PKCS#11	1min → 10s	4 → 0	each single improvement
CH'07 RFID	50min → 2min	0 → 0	fresh order

Applied to Existing Models

Protocol	Runtime	Helper-Lemmas	Why is it faster?
WPA-2	1:20h → 7min	74 → 73	monotonicity (of counters)
5G	8min → 2min	7 → 6	our number system
YubiKey	20s → 1s	4 → 3	our number system
PKCS#11	1min → 10s	4 → 0	each single improvement
CH'07 RFID	50min → 2min	0 → 0	fresh order

Our Proofs

Our Proofs

- TreeKEM
  - distributed tree
  - forward-secrecy

Our Proofs

- TreeKEM
  - distributed tree
  - forward-secrecy
- S/Key
  - hash-chain
  - authentication

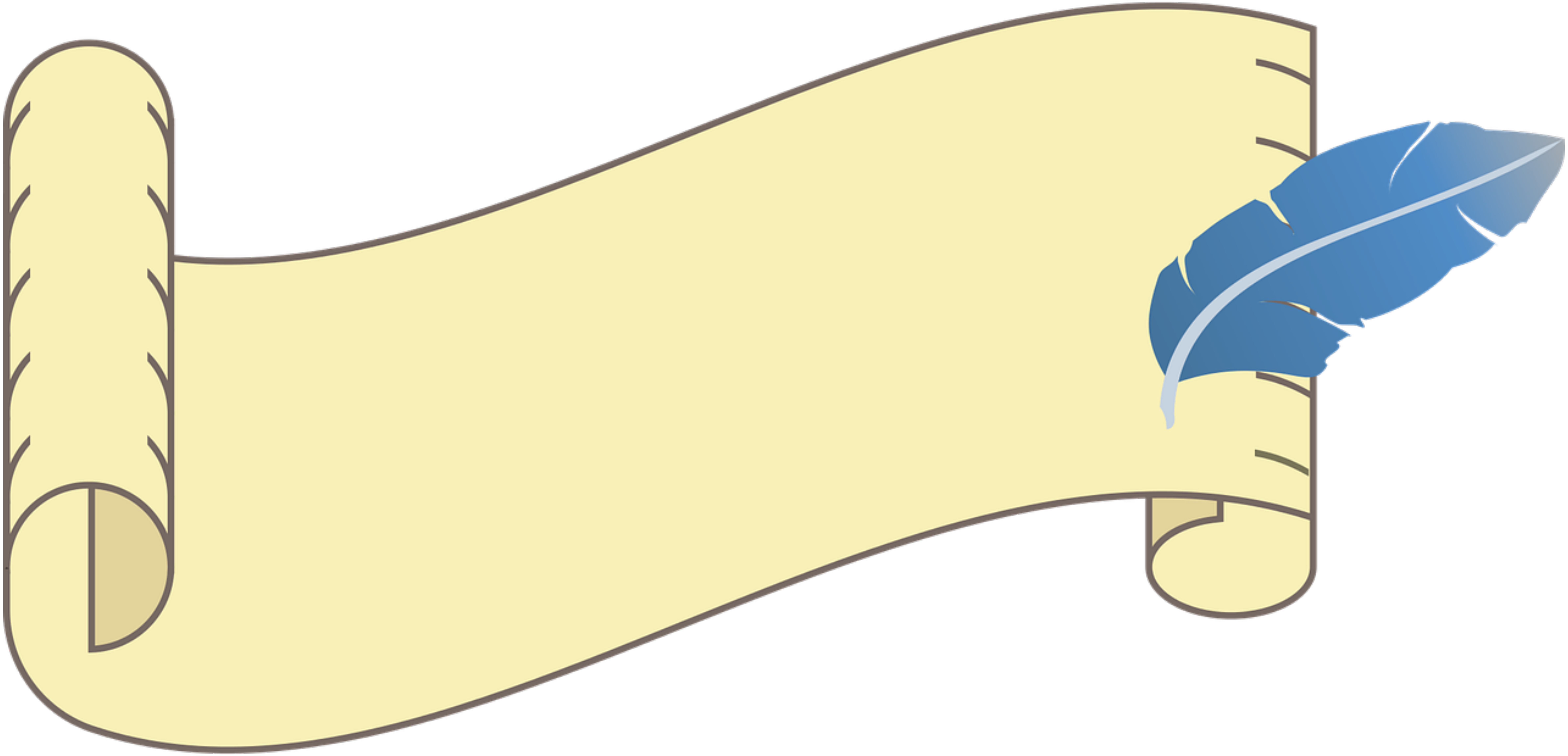
https://www.ftsafe.com/products/OTP/Single_Button_OTP

Our Proofs

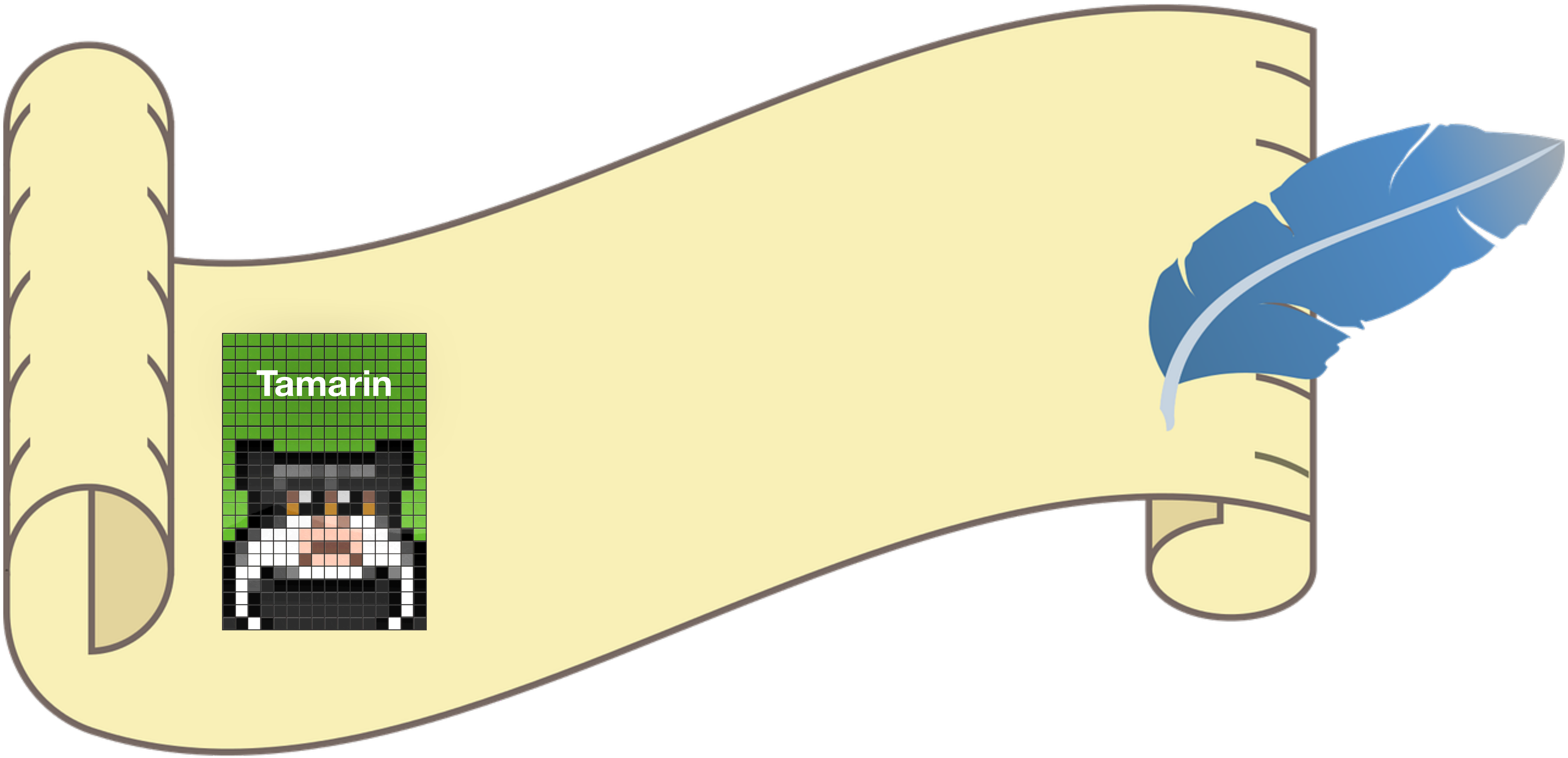
- TreeKEM
  - distributed tree
  - forward-secrecy
- S/Key
  - hash-chain
  - authentication
- Tesla Scheme 2
  - hash-chain like S/Key
  - authentication, secrecy
  - prev. example of Tamarins limits

https://www.ftsafe.com/products/OTP/Single_Button_OTP

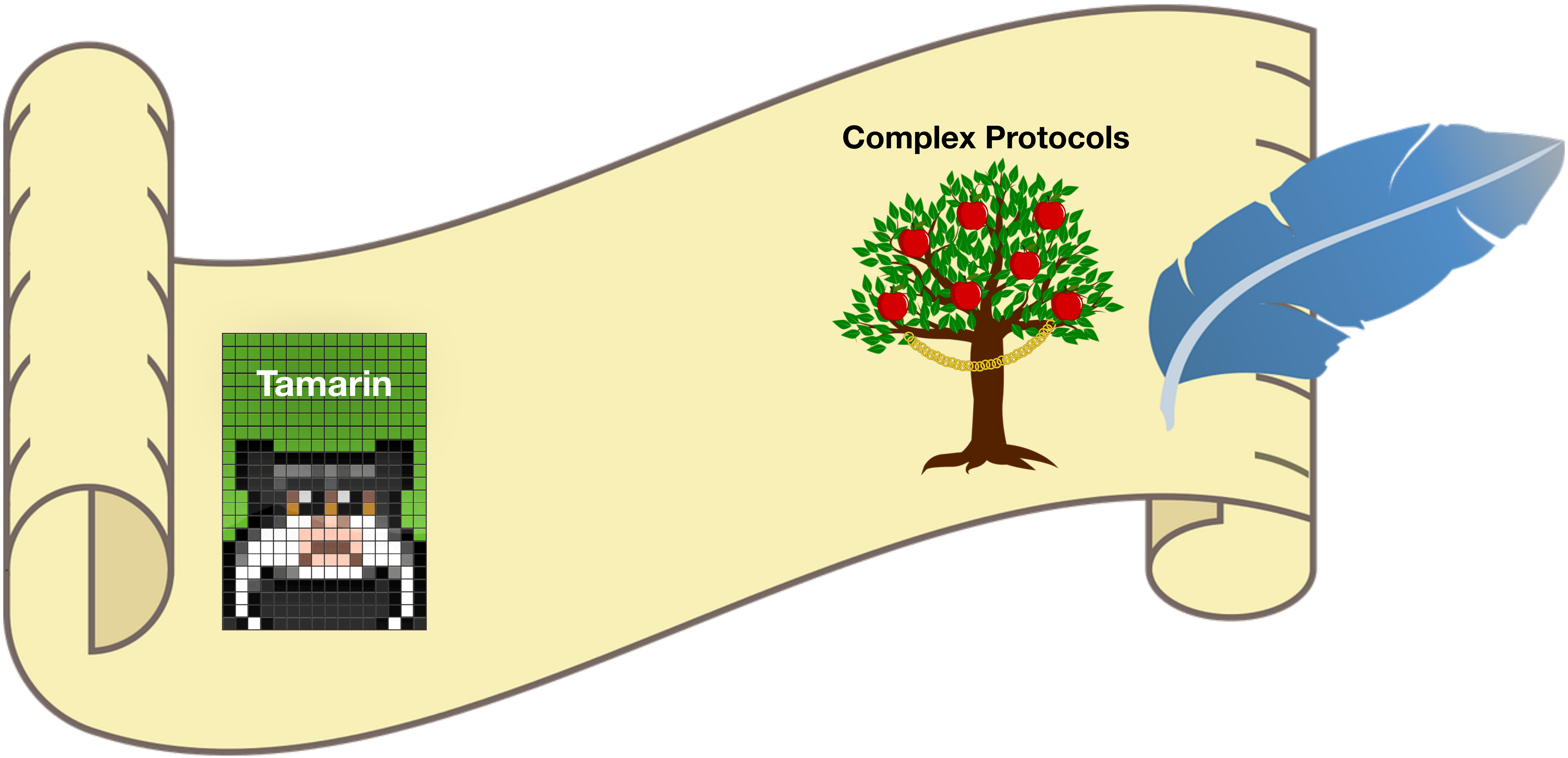
Paper Summary



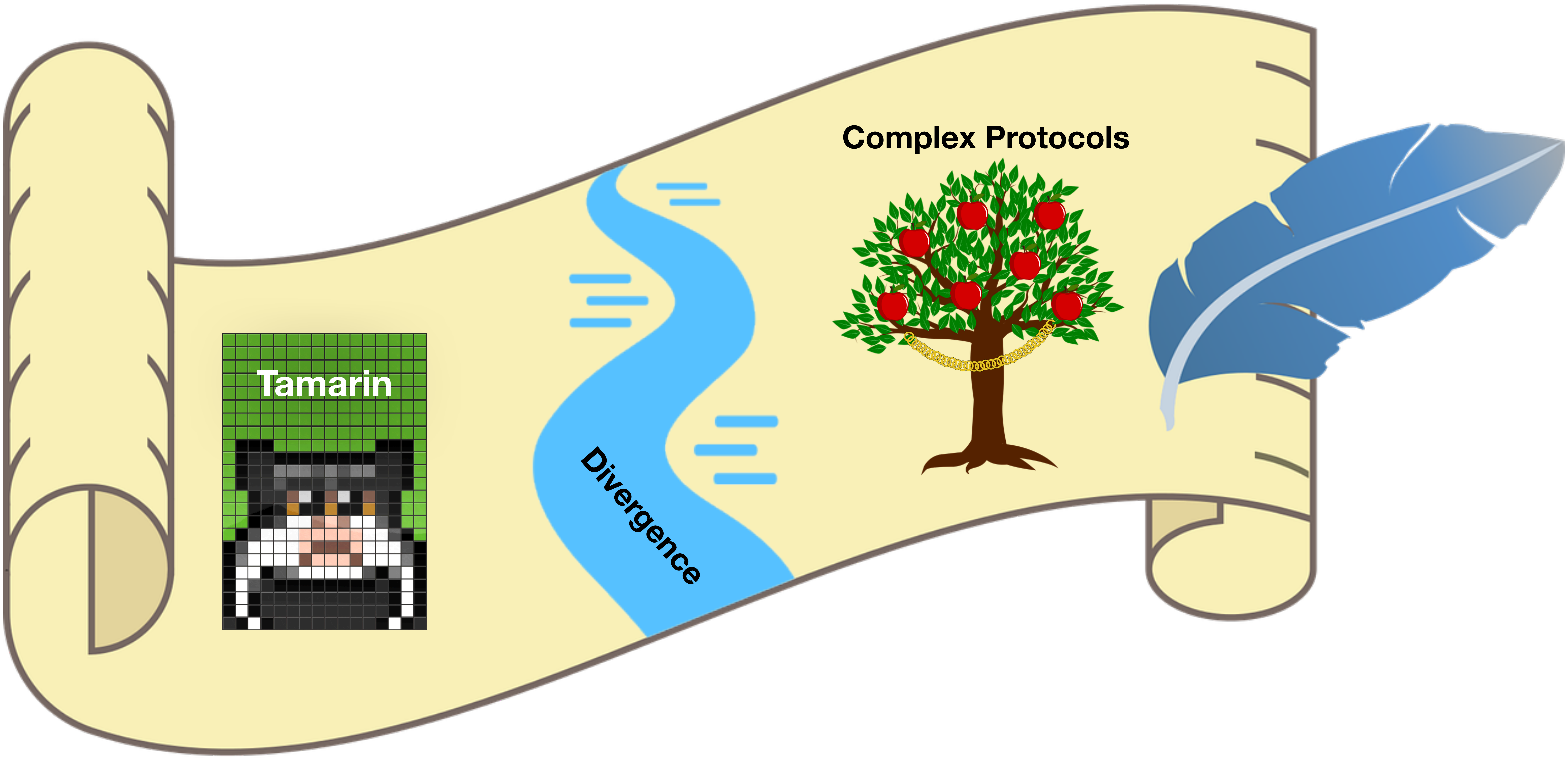
Paper Summary



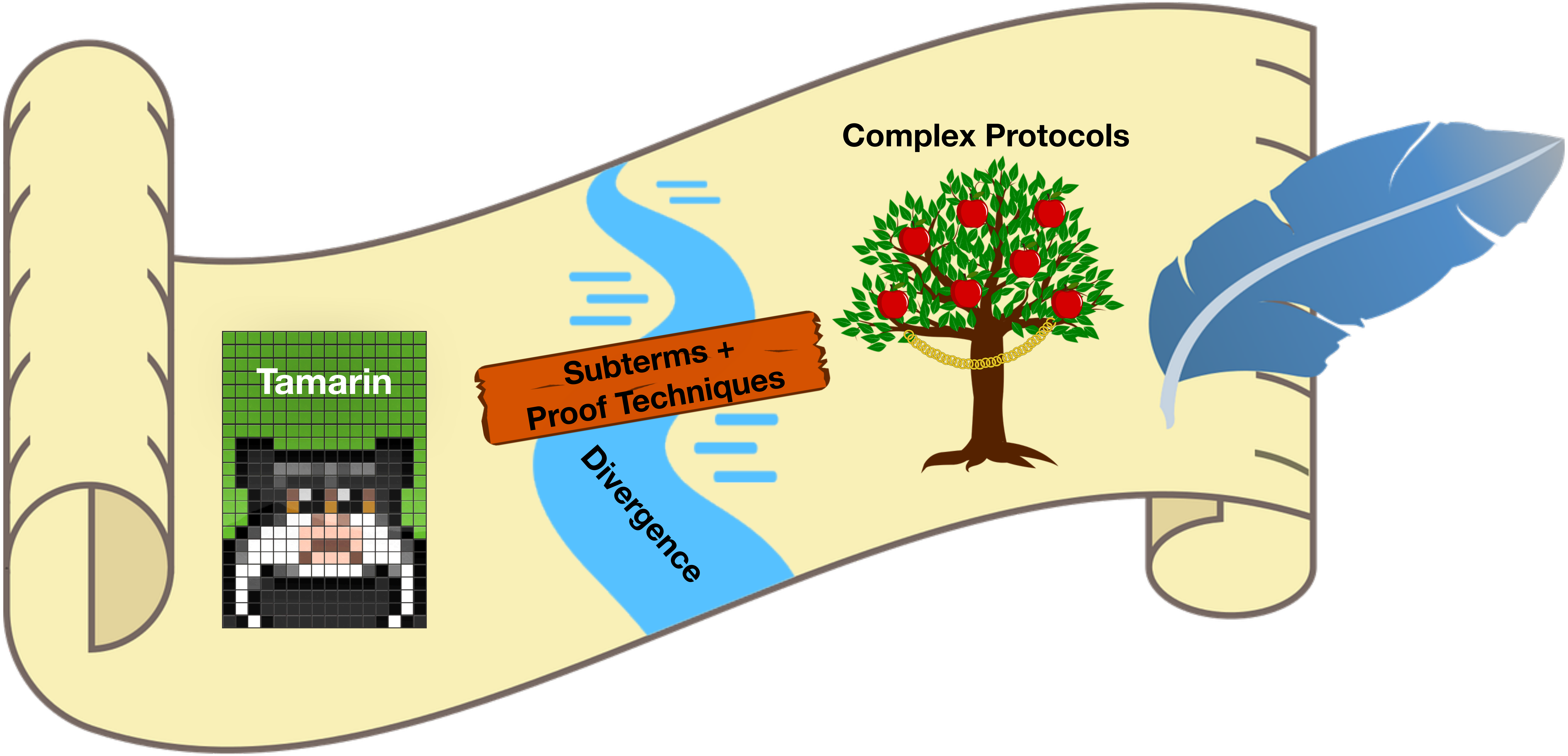
Paper Summary



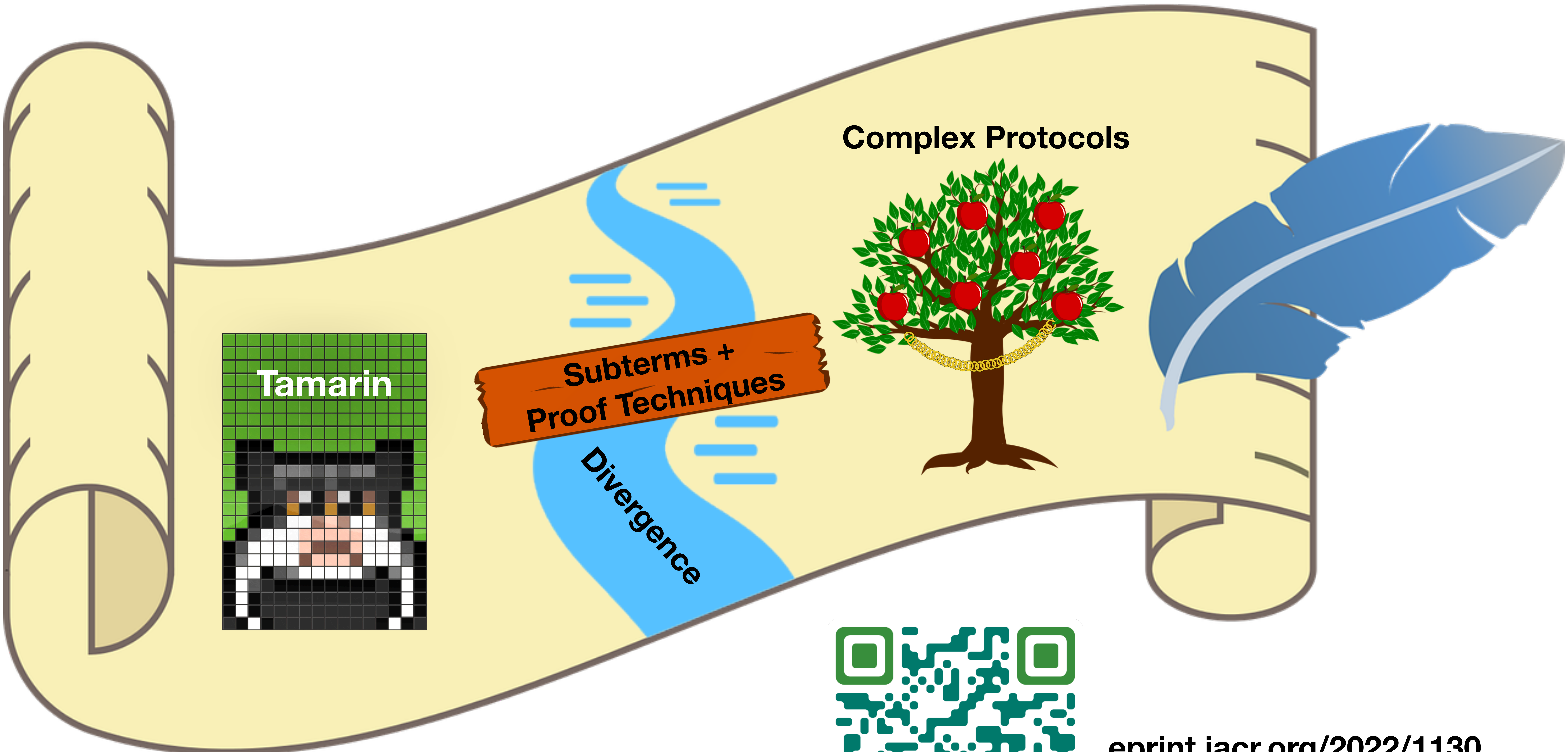
Paper Summary



Paper Summary



Paper Summary



eprint.iacr.org/2022/1130