ProveIt

How to make proving cryptographic protocols less tedious

Liina Kamm

Computer Science Theory Days at Kubija

28.01.2012

STACC Software Technology and Applications Competence Center
CYBERNETICA
TARTU ÜLIKOOL 1632 UNIVERSITAS TARTUENSIS
ETF9171
European Union Regional Development Fund
Investing in your future

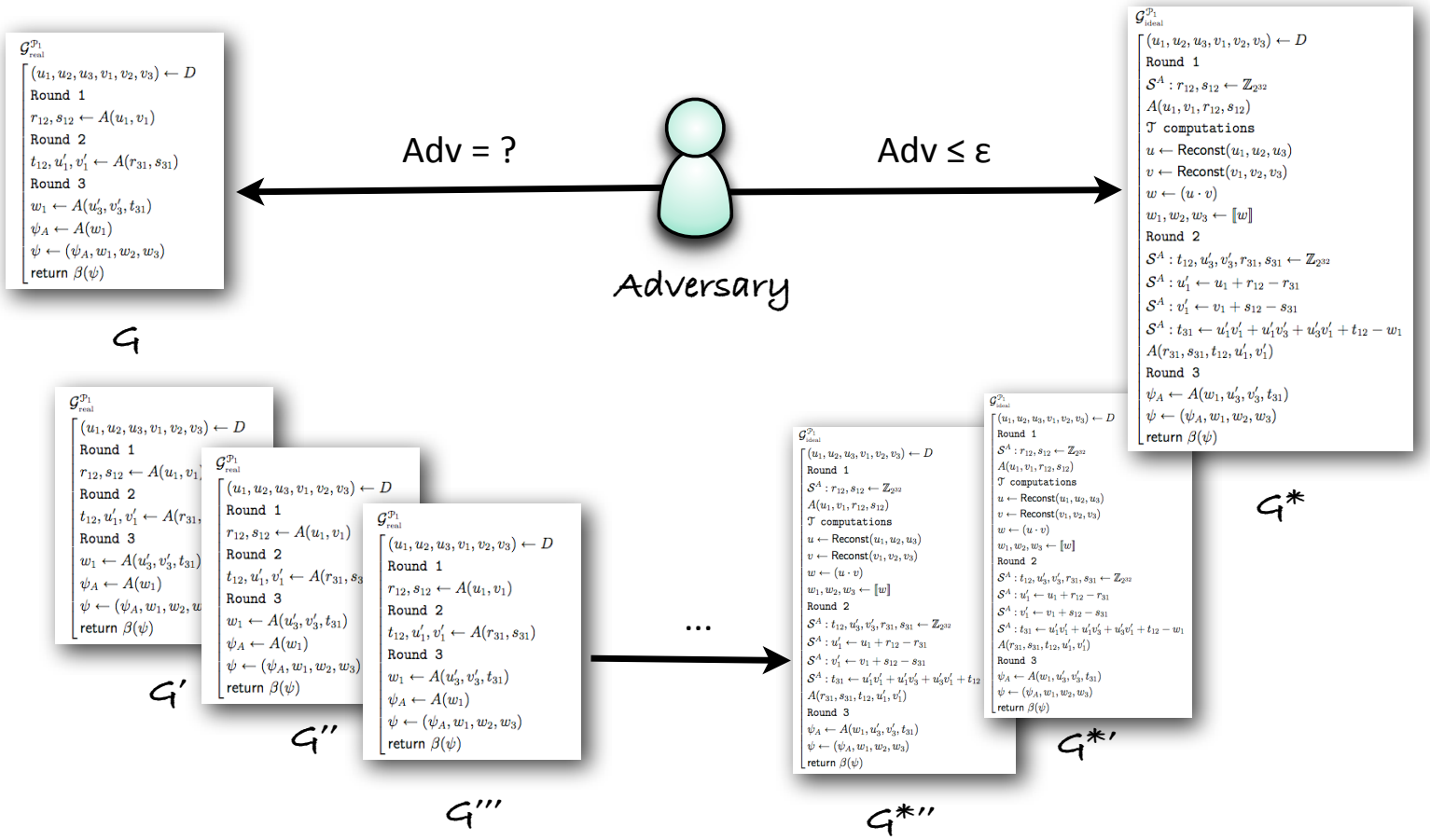
Overview

- Motivation
- Game-based protocol proofs
- ProveIt
- Who is it for?

Motivation

- Proving a security protocol using game rewriting is often
  - Error-prone
  - Time consuming

Game-Based Proofs



[Bellare, Rogaway 04], [Shoup 04]

Transformations

- Consider two games $\mathcal{G}$ and $\mathcal{H}$ with adversaries *A* and *B* and let $p_{\mathcal{G}}$ and $p_{\mathcal{H}}$ be the respective probabilities, that *A* wins $\mathcal{G}$ and *B* wins $\mathcal{H}$
- Transformation is **safe** if $p_{\mathcal{G}} \leq p_{\mathcal{H}}$
- Transformation is **conservative** if $p_{\mathcal{G}} = p_{\mathcal{H}}$
- Transformation is **lossy** if

$$p_{\mathcal{G}} \leq p_{\mathcal{H}} + \varepsilon \quad \text{or} \quad p_{\mathcal{G}} \leq c \cdot p_{\mathcal{H}}$$

for some particular $\varepsilon > 0$ or $c > 1$

Features of ProveIt

- Protocol entered in pseudocode
- Protocol parsed from text to abstract syntax tree
- Transformations:
  - FreeStep, User Defined Step
  - PRP/PRF Switching, Function Rename
  - Dead Code Elimination, Statement Switching

FreeStep, User Defined Step

- Preconditions: none
- Application rules: can be applied to any statement
- Difference between the games: user specified

PRP/PRF Switching

- Preconditions: the secret key used in the pseudorandom permutation *f* must not appear on the right side of other statements
- Application rules: can be applied to the function call of *f*
- Difference between the games?

PRP/PRF Switching Lemma

- Let $n \geq 1$ be an integer. Let A be an adversary that asks at most q oracle queries. Then

$$|\Pr[A^{\pi} \Rightarrow 1] - \Pr[A^{\rho} \Rightarrow 1]| \leq \frac{q(q-1)}{2^{n+1}}$$

- π - randomly sampled from the set of all permutations on $\{0,1\}^n$
- ρ - randomly sampled from the set of all functions from $\{0,1\}^n$ to $\{0,1\}^n$

PRP/PRF Switching

- Preconditions: the secret key used in the pseudorandom permutation *f* must not appear on the right side of other statements
- Application rules: can be applied to the function call of *f*
- Difference between the games:

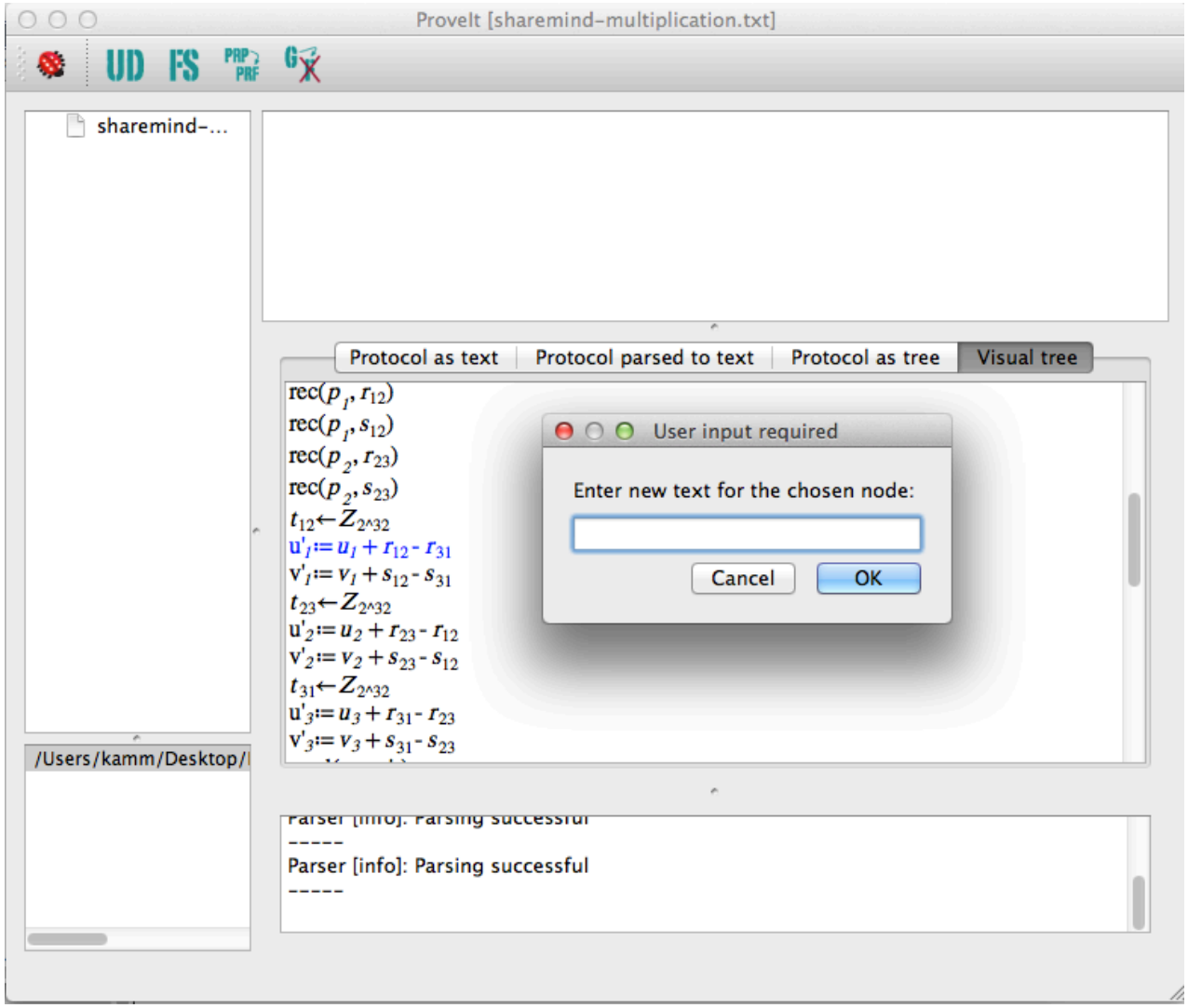
$$\mathsf{sd}(\mathcal{G}_0^A, \mathcal{G}_1^A) \leq \frac{q(q-1)}{2^{n+1}}$$

Protocols

Protocol as text | Protocol parsed to text | Protocol as tree | Visual tree

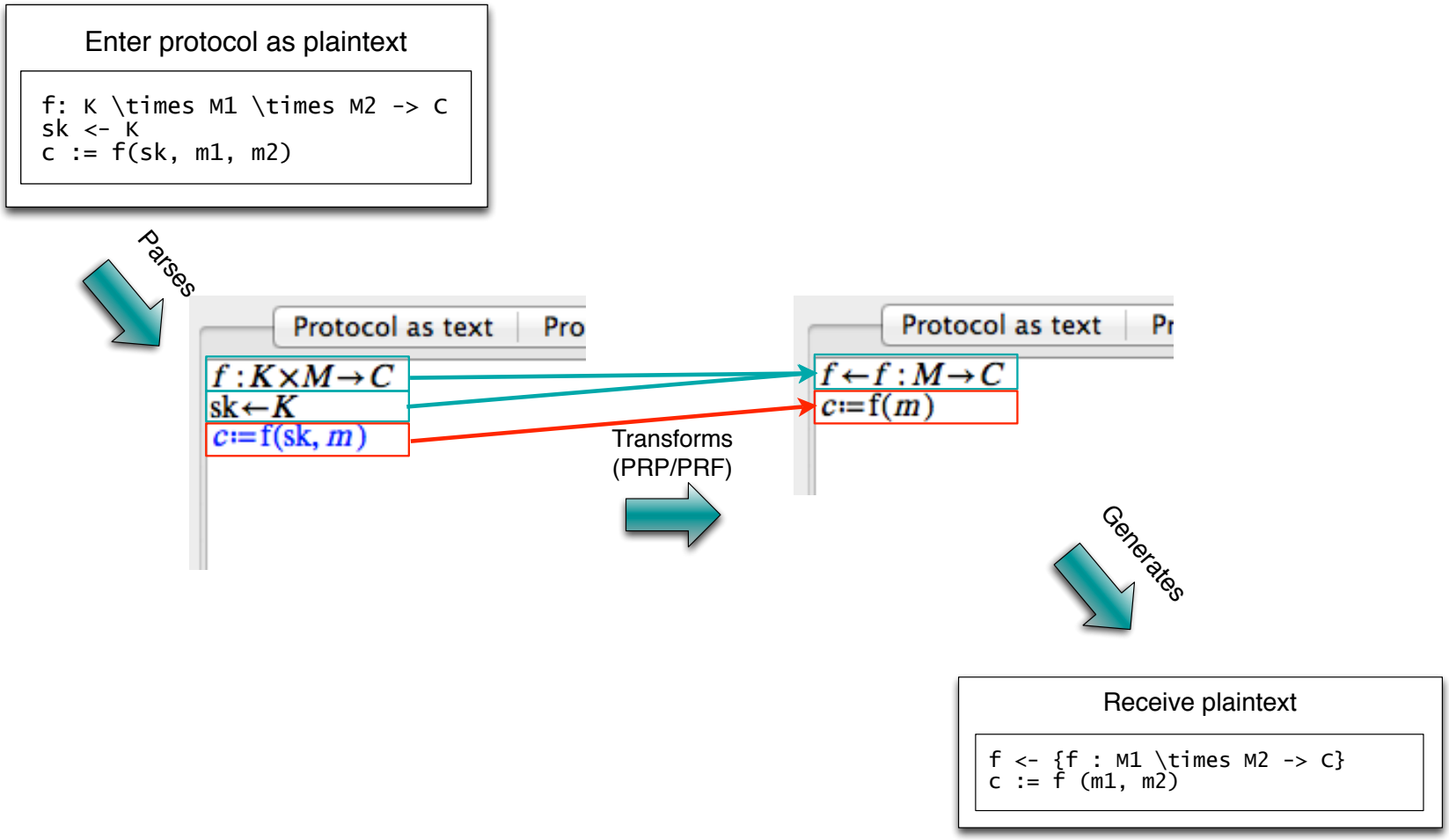
```
r_23 <- Z_2^32
s_23 <- Z_2^32
r_31 <- Z_2^32
s_31 <- Z_2^32
send(p_2, r_12)
send(p_2, s_12)
send(p_3, r_23)
send(p_3, s_23)
send(p_1, r_31)
send(p_1, s_31)

rec(p_3, r_31)
rec(p_3, s_31)
rec(p_1, r_12)
rec(p_1, s_12)
rec(p_2, r_23)
rec(p_2, s_23)
t_12<- Z_2^32
u'_1 := u_1 + r_12 - r_31
v'_1 := v_1 + s_12 - s_31
t_23 <- Z_2^32
u'_2 := u_2 + r_23 - r_12
v'_2 := v_2 + s_23 - s_12
t_31 <- Z_2^32
u'_3 := u_3 + r_31 - r_23
v'_3 := v_3 + s_31 - s_23
send(p_2, u'_1 )
send(p_2, v'_1)
send(p_2, t_12)
send(p_3, u'_2 )
send(p_3, v'_2)
```

Protocols


ProveIt [sharemind-multiplication.txt]
UD
FS
PRP
PRF
sharemind-...
Protocol as text
Protocol parsed to text
Protocol as tree
Visual tree
rec(p1, r12)
rec(p1, s12)
rec(p2, r23)
rec(p2, s23)
t12←Z2^32
u'1:= u1 + r12 - r31
v'1:= v1 + s12 - s31
t23←Z2^32
u'2:= u2 + r23 - r12
v'2:= v2 + s23 - s12
t31←Z2^32
u'3:= u3 + r31 - r23
v'3:= v3 + s31 - s23
User input required
Enter new text for the chosen node:
Cancel
OK
/Users/kamm/Desktop/
Parser [info]: Parsing successful

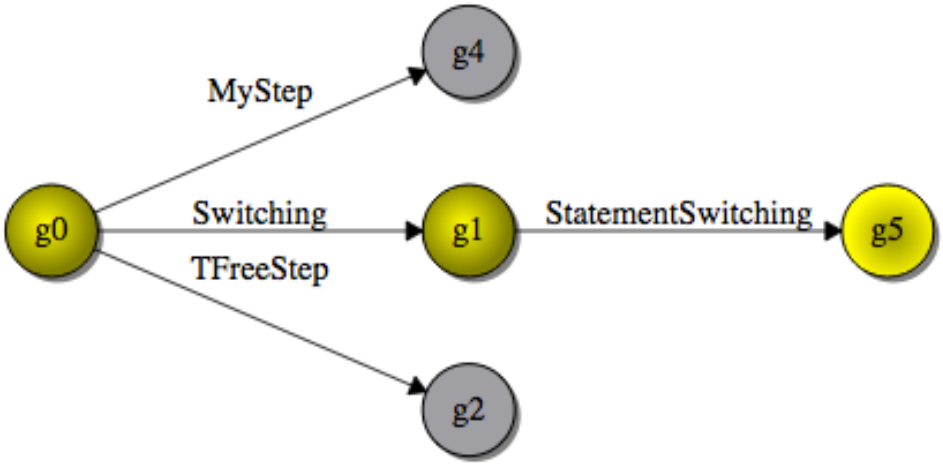
Parser [info]: Parsing successful

Workflow


Enter protocol as plaintext
f: K \times M1 \times M2 -> C
sk <- K
c := f(sk, m1, m2)
Parses
Protocol as text
Pro
$f : K \times M \to C$
$\mathrm{sk} \leftarrow K$
$c := \mathrm{f}(\mathrm{sk}, m)$
Transforms
(PRP/PRF)
Protocol as text
Pr
$f \leftarrow f : M \to C$
$c := \mathrm{f}(m)$
Generates
Receive plaintext
f <- {f : M1 \times M2 -> C}
c := f (m1, m2)

Game Trees


g0
Switching
g1
RFSimulation
g2
DeadDefinition
g3
StatementSwitching
g4


g4
MyStep
g0
Switching
g1
StatementSwitching
g5
TFreeStep
g2

Advantages

- Automatic game-rewriting
  - Reduces the number of rewriting errors
  - Makes the proving process faster
- Checks for rule usage
  - Is it possible to use a certain proof step for the selected statement?
  - What steps can I use for a selected statement?
- Helps researchers, students, teaching assistants

Ongoing Work

- Control flow analysis
- Type inference
- Protocol presentation
- More transformations
- Translation to EasyCrypt and CertiCrypt
- User feedback
  - Students
  - Researchers

Demo?