Advances on Protocol Indistinguishability Analysis in Maude-NPA

José Meseguer[1] **(PI)**
Catherine Meadows[2]
Santiago Escobar[1,3]
Sonia Santiago [1,3]

[1]University of Illinois at Urbana-Champaign (USA)
[2]Naval Research Laboratory (USA)
[3]Universitat Politècnica de València (Spain)

NSA SoS Lablet Meeting
CMU, September 2013

Outline

Outline

What is Indistinguishability?

- Concept used to reason about security in cryptographic algorithms and protocols
  - Example: Chosen plaintext security
  - Attacker chooses between messages m_1 and m_2
  - Receives encrypted message that could be $e(k, m_1)$ or $e(k, m_2)$
  - Performs tests on message, then tries to guess which one it is
- Indistinguishability particularly useful for reasoning about protocols that protect low-entropy data
  - Defense against password guessing attacks
  - Voting
  - Anonymous routing
  - Privacy-preserving database queries

Research Problems we are Addressing

- Can indistinguishability be formally defined in terms of state reachability in general and easy to understand terms?
- Are there sound and complete methods for verifying indistinguishability modulo algebraic properties?
- How general can we make an approach based on state reachability?
- Can such methods be integrated into the Maude-NPA tool?

Outline

Our Approach

- Like ProVerif, run two instances of the protocol in lockstep
- We introduce a pairing operation on protocols
- In Maude-NPA, a “bad”state (i.e. a state that is not indistinguishable) is:
  - A state in which one component of the pair is reachable, but the other one is not, and/or
  - A state in which performing different actions, the intuder has learnt the same message in one component of the pair but two different messages in the other component.

Protocol Pairing

Given two protocols $\mathcal{P}_1$ and $\mathcal{P}_2$, we define a protocol pairing $\mathcal{P}_1$,$\mathcal{P}_2$ in terms of the strand representation iff

1. $\mathcal{P}_1$ and $\mathcal{P}_2$ share the same equational theory of messages and message functions
2. $\mathcal{P}_1$ and $\mathcal{P}_2$ share the same intruder strands
3. Correspondence between protocol strands of $\mathcal{P}_1$ and $\mathcal{P}_2$ such that two corresponding strands
    - Have the same length
    - Have output and input messages in the same order
    - Can differ in the actual messages

Synchronous Product of Protocols

Given a protocol pairing $\mathcal{P}_1$,$\mathcal{P}_2$, s.t.

- $\mathcal{P}_1 = (\Sigma_{\mathcal{P}}, E_{\mathcal{P}}, T_{\mathcal{P}_1})$
- $\mathcal{P}_2 = (\Sigma_{\mathcal{P}}, E_{\mathcal{P}}, T_{\mathcal{P}_2})$

A synchronous product of $\mathcal{P}_1$ and $\mathcal{P}_2$,

$$\mathcal{P}_1 \otimes \mathcal{P}_2 = (\Sigma_{\mathcal{P}} \cup \{\otimes\}, E_{\mathcal{P}}, T_{\mathcal{P}_1} \otimes T_{\mathcal{P}_2})$$

is a new protocol where the strands of $\mathcal{P}_1 \otimes \mathcal{P}_2$ are obtained by "zipping together" each strand of $\mathcal{P}_1$ and its corresponding strand from $\mathcal{P}_2$

Example of Synchronous Product of Protocols

Protocol $\mathcal{P}_1$

$$\begin{array}{lll}(\mathrm{Alice}) :: r_1, r_2 :: [\ +(k(A,B,r_1)), +(e(k(A,B,r_1),n(A,r_2)))\ ] \\ (\mathrm{Bob}) :: nil :: [\ -(Key), -(e(Key,N_A))\ ]\end{array}$$

Protocol $\mathcal{P}_2$

$$\begin{array}{lll}(\mathrm{Alice}) :: r'_1, r'_2, r_3 :: [\ +(k(A,B,r_3)), +(e(k(A,B,r'_1),n(A,r'_2)))\ ] \\ (\mathrm{Bob}) :: nil :: [\ -(Key_1), -(e(Key_2,N_A))\ ]\end{array}$$

Synchronous product $\mathcal{P}_1 \otimes \mathcal{P}_2$

$$\begin{array}{l}(\mathrm{Alice}) :: r_1, r_2, r_3 :: [\ +(k(A,B,r_1) \otimes k(A,B,r_3)), \\ \qquad +(e(k(A,B,r_1),n(A,r_2)) \otimes e(k(A,B,r_1),n(A,r_2)))\ ]\ \& \\ (\mathrm{Bob}) :: nil :: [\ -(Key \otimes Key_1), \\ \qquad -(e(Key,N_A) \otimes e(Key_2,N_A))\ ]\end{array}$$

Properties of Synchronous Product Operator

- Introduce a new type SingleMessage.

  $\otimes : \mathit{SingleMessage} \times \mathit{SingleMessage} \rightarrow \mathit{Message}$

  - This means that $\otimes$ can never be applied twice
- For any function f on messages add equation

  $f(x_1 \otimes y_1, \ldots, x_k \otimes y_k) = f(x_1, \ldots, x_k) \otimes f(y_1, \ldots y_k)$

  - Allows principals in synchronous product to apply functions to messages of the form $s \otimes t$ and produce another message of the form $s' \otimes t'$
- Result: Adding synchronous product does not affect equational semantics of Maude-NPA; it only extends the signature and equational theory preserving the finite variant property!

Projections on Synchronous Product of Protocols

Given a synchronous product of protocols $\mathcal{P}_1 \otimes \mathcal{P}_2$, there are projections

- $\pi_1 : \mathcal{P}_1 \otimes \mathcal{P}_2 \rightarrow \mathcal{P}_1$
- $\pi_2 : \mathcal{P}_1 \otimes \mathcal{P}_2 \rightarrow \mathcal{P}_2$

from the states of the synchronous product to the states of each of these protocols such that π_1 and π_2 are simulation maps

Example:

$$\pi_1 \; : \{[\, (m_1 \otimes m'_1)^{\pm}, \ldots, (m_n \otimes m'_n)^{\pm}] \;\&\; \ldots\} \rightarrow \{[m_1^{\pm}, \ldots, m_n^{\pm}] \;\&\; \ldots\}$$

Protocol Indistinguishability

- We propose a formal definition of indistinguishability of two protocols as the conjunction of two more basic properties over a protocol pairing $\mathcal{P}_1$,$\mathcal{P}_2$
- First, some preliminaries are required.

Attacker Event Sequences

- The attacker has complete control over the network
- Therefore, behavior of a protocol for an attacker can be reduced to a sequence of attacker events that are either
  (i) Message sent/received events, or
  (ii) Message manipulation actions
- Transitions from an initial state to any reachable state are performed via a sequence of attacker events of category (i) or (ii), called attacker event sequence (AES)
- In Maude-NPA this corresponds to the fact any state transition corresponds to an attacker event of either category (i) or (ii)

Indistinguishability notion in Maude-NPA

Two protocols $\mathcal{P}_1$ and $\mathcal{P}_2$ are indistinguishable iff:

1. $\mathcal{P}_1$ and $\mathcal{P}_2$ have *indistinguishable attacker event sequences* (IAES), and
2. $\mathcal{P}_1$ and $\mathcal{P}_2$ have *indistinguishable messages* (IM)

Intuitive Definitions

- **Indistinguishable Attacker Event Sequences (IAES)**

  $\mathcal{P}_1$ and $\mathcal{P}_2$ have indistinguishable AESs (IAES) if from any initial state the attacker is able to perform exactly the same type of event sequences for each protocol

  Transitions of the same type if involve same actions appearing in synchronous products of strands at same point of execution

- **Indistinguishable Messages (IM)**

  The intruder can never perform two different AESs, say, α and β, so that as a result of α and β the intruder either learns

  (i) the *same* message from $\mathcal{P}_1$ but different messages from $\mathcal{P}_2$, or
  (ii) different messages from $\mathcal{P}_1$ but the *same* message from $\mathcal{P}_2$

Implementing Indistinguishability in Maude-NPA

- No need to change the rewrite rule semantics
- The completeness of the Maude-NPA indistinguishability analysis is based on the satisfiability of IM and IAES
    - The satisfiability of IM and IAES can be proved in Maude-NPA as the unreachability of “bad” states (denoted by attack states) from an initial state

Outline

Encrypted Key Exchange (EKE) protocol (*)

- Protocol intended secure against passive guessing attacks.
- Alice encrypts an ephemeral public key using a shared password, and sends it to Bob; then both challenge each other with encrypted nonces.

1. $A \longrightarrow B: \quad enc(pw(A, B), pkey(A, r_1)))$
2. $B \longrightarrow A: \quad enc(pw(A, B), penc(pkey(A, r_1), skey(A, B, r_3)))$
3. $A \longrightarrow B: \quad senc(skey(A, B, r_3), n(A, r_2))$
4. $B \longrightarrow A: \quad senc(skey(A, B, r_3), n(A, r_2); n(B, r_4))$
5. $A \longrightarrow B: \quad senc(skey(A, B, r_3), n(B, r_4))$

(*) S. M. Bellovin; M. Merritt. Encrypted Key Exchange: Password-Based Protocols Secure Against Dictionary Attacks. Proceedings of the IEEE Symposium on Research in Security and Privacy, Oakland, 1992

Encrypted Key Exchange (EKE) protocol

In strand notation the protocol can be specifed by Alice and Bob strands. Note the final checking by Bob of his challenge to Alice (Step 6), implicit in the textbook notation.

$$
\begin{aligned}
(\text{Alice}) :: r_1, r_2 :: \ [\ & +(A\ ;\ enc(pw(A,B), pkey(A,r_1))),\\
& -(enc(pw(A,B), penc(pkey(A,r_1), SKB))),\\
& +(senc(SKB, n(A,r_2))), -(senc(SKB, n(A,r_2); NB)),\\
& +(senc(SKB, NB))\ ]\ \&\\
(\text{Bob}) :: r_3, r_4 :: \ [\ & -(A\ ;\ enc(pw(A,B), PKA))),\\
& +(enc(pw(A,B), penc(PKA, skey(A,B,r_3)))),\\
& -(senc(skey(A,B,r_3), NA)),\\
& +(senc(skey(A,B,r_3), NA; n(B,r_4))),\\
& -(senc(skey(A,B,r_3), n(B,r_4)))\ ]
\end{aligned}
$$

Encrypted Key Exchange (EKE) protocol

Equational Properties: public-key and symmetric encryption and decryption with usual cancellation properties:

$$penc(PK, pdec(inv(PK), X)) = X$$
$$pdec(inv(PK), penc(PK, X)) = X$$

$$senc(SK, sdec(SK, X)) = X$$
$$sdec(SK, senc(SK, X)) = X$$

$$enc(PW, dec(P, X)) = X$$
$$dec(PW, enc(P, X)) = X$$

Encrypted Key Exchange (EKE) protocol

- Synchronous product of protocols ($\mathcal{P}_1 \otimes \mathcal{P}_2$):
  - Same honest strands in $\mathcal{P}_1$ and $\mathcal{P}_2$
  - In $\mathcal{P}_1$ intruder generates right password
  - In $\mathcal{P}_2$ intruder generates wrong password

$$:: nil :: [\ +(pw(A,B) \otimes pw(A,i))\ ]$$

- $\mathcal{P}_1 \otimes \mathcal{P}_2$ is **not** indistinguishable (IAES not satisfied)
  - In $\mathcal{P}_1$ intruder can decrypt message
    `enc(pw(A,B),pkey(A,`r_1`))`
  - In $\mathcal{P}_2$ intruder cannot decrypt that message because of wrong generated password

Encrypted Key Exchange (EKE) protocol

Protocol specification of $\mathcal{P}_1 \otimes \mathcal{P}_2$: differences only in intruder actions

$$
\begin{array}{l}
\text{(Alice)} :: r_1, r_2 :: \\
\quad [\ +((A\ ;\ enc(pw(A,B), pkey(A,r_1))) \otimes (A\ ;\ enc(pw(A,B), pkey(A,r_1)))), \\
\quad\ -((enc(pw(A,B), penc(pkey(A,r_1), SKB))) \\
\qquad \otimes(enc(pw(A,B), penc(pkey(A,r_1), SKB)))), \\
\quad\ +((senc(SKB, n(A,r_2))) \otimes (senc(SKB, n(A,r_2)))), \\
\quad\ -((senc(SKB, n(A,r_2); NB)) \otimes (senc(SKB, n(A,r_2); NB))), \\
\quad\ +((senc(SKB, NB)) \otimes (senc(SKB, NB)))\ ]\ \&
\end{array}
$$

$$
\begin{array}{l}
\text{(Bob)} :: r_3, r_4 :: \\
\quad [\ -((A\ ;\ enc(pw(A,B), PKA)) \otimes (A\ ;\ enc(pw(A,B), PKA))), \\
\quad\ +((enc(pw(A,B), penc(PKA, skey(A,B,r_3)))) \\
\qquad \otimes(enc(pw(A,B), penc(PKA, skey(A,B,r_3))))), \\
\quad\ -((senc(skey(A,B,r_3), NA)) \otimes (senc(skey(A,B,r_3), NA))), \\
\quad\ +((senc(skey(A,B,r_3), NA; n(B,r_5))) \\
\qquad \otimes(senc(skey(A,B,r_3), NA; n(B,r_5)))), \\
\quad\ -((senc(skey(A,B,r_3), n(B,r_5))) \otimes (senc(skey(A,B,r_3), n(B,r_5))))\ ]\ \&
\end{array}
$$

$$
\begin{array}{l}
\text{(Intruder)} \\
\quad :: nil :: [\ +(pw(A,B) \otimes pw(A,i))\ ]\ \&\ [\ldots]
\end{array}
$$

Encrypted Key Exchange (EKE) protocol

Equational theory of $\mathcal{P}_1 \otimes \mathcal{P}_2$: original theory and pairing extension

$$
\begin{aligned}
penc(PK, pdec(inv(PK), X)) &= X \\
pdec(inv(PK), penc(PK, X)) &= X \\
senc(SK, sdec(SK, X)) &= X \\
sdec(SK, senc(SK, X)) &= X \\
enc(PW, dec(P, X)) &= X \\
dec(PW, enc(P, X)) &= X
\end{aligned}
$$

$$
\begin{aligned}
penc(PK1 \otimes PK2, M1 \otimes M2)) &= penc(PK1, M1) \otimes penc(PK2, M2) \\
pdec(K1 \otimes K2, M1 \otimes M2)) &= pdec(K1, M1) \otimes penc(K2, M2) \\
senc(SK1 \otimes SK2, M1 \otimes M2)) &= senc(SK1, M1) \otimes senc(SK2, M2) \\
sdec(SK1 \otimes SK2, M1 \otimes M2)) &= sdec(SK1, M1) \otimes penc(SK2, M2) \\
enc(PW1 \otimes PW2, M1 \otimes M2) &= enc(PW1, M1) \otimes enc(PW2, M2) \\
dec(PW1 \otimes PW2, M1 \otimes M2) &= dec(PW1, M1) \otimes dec(PW2, M2) \\
(M1 \otimes M2) ; (M1' \otimes M2') &= (M1 ; M1') \otimes (M2 ; M2') \\
inv(M1 \otimes M2) &= inv(M1) \otimes inv(M2)
\end{aligned}
$$

Encrypted Key Exchange (EKE) protocol

Maude-NPA proves the EKE does not satisfy IAES property

- Attack state:

```
eq ATTACK-STATE(0)
= :: nil ::
  [ nil | -(pair(Z,pw(a,b))),
          -(pair(enc(pw(a,b),PK),enc(pw(a,b),PK))),
          +(pair(PK,PK)), nil ]
  || pair(Z,pw(a,b)) inI, Z != pw(a,b)
  || nil
  || nil
  || nil
 [nonexec] .
```

- Analysis output:

```
Maude> red summary(0,1) .
reduce in MAUDE-NPA : summary(0, 1) .
rewrites: 8049021 in 12221ms cpu (12252ms real) (658619 rewrites/second)
result Summary: States>> 3 Solutions>> 1
```

Outline

Forwards/Backwards semantics

- Maude-NPA has backwards operational semantics
- Indistinguishability notion based on some form of forwards semantics
- We need to define forwards semantics for Maude-NPA
  - Must be sound and complete w.r.t. backwards semantics
  - Simple intuitive forwards semantics is easy to define
  - But we want a forwards rewriting-based semantics that is executable in Maude and allows standard model checking

Forward semantics

- Forwards oriented rules for executability in Maude (no extra variables in rhs of transition rules)

(1a) $\{SS \,\&\, \{IK\} \,\&\, [u_1^{\pm}, \ldots, u_{j-1}^{\pm} \mid nil]\} \to \{SS \,\&\, \{u_j{\in}\mathcal{I}, IK\} \,\&\, [u_1^{\pm}, \ldots, u_{j-1}^{\pm}, u_j^{+} \mid nil]\}$
for each $[u_1^{\pm}, \ldots, u_{j-1}^{\pm}, u_j^{+}, u_{j+1}^{\pm}, \ldots, u_n^{\pm}] \in \mathcal{P}$, $\;$ *if* $((u_j{\in}\mathcal{I}) \notin IK$ *and* $j \geqslant 1$

(1b) $\{SS \,\&\, \{IK\} \,\&\, [u_1^{\pm}, \ldots, u_{j-1}^{\pm} \mid nil]\} \to \{SS \,\&\, \{IK\} \,\&\, [u_1^{\pm}, \ldots, u_{j-1}^{\pm}, u_j^{+} \mid nil]\}$
for each $[u_1^{\pm}, \ldots, u_{j-1}^{\pm}, u_j^{+}, u_{j+1}^{\pm}, \ldots, u_n^{\pm}] \in \mathcal{P}$, $\;$ *and* $j \geqslant 1$

(2a) $\{SS \,\&\, \{IK\} \to \{SS \,\&\, \{u_1{\in}\mathcal{I}, IK\} \,\&\, [u_1^{+} \mid nil]\}$ for each $[u_1^{\pm}, \ldots, u_n^{\pm}] \in \mathcal{P}$ $\;$ *if* $((u_1{\in}\mathcal{I}) \notin IK$

(2b) $\{SS \,\&\, \{IK\} \to \{SS \,\&\, \{IK\} \,\&\, [u_1^{+} \mid nil]\}$ for each $[u_1^{\pm}, \ldots, u_n^{\pm}] \in \mathcal{P}$

(3) $\{SS \,\&\{u_j{\in}\mathcal{I}, IK\} \,\&\, [u_1^{\pm}, \ldots, u_{j-1}^{\pm} \mid nil]\} \to \{SS \,\&\, \{u_j{\in}\mathcal{I}, IK\} \,\&\, [u_1^{\pm}, \ldots, u_{j-1}^{\pm}, u_j^{-} \mid nil]\}$
for each $[u_1^{\pm}, \ldots, u_{j-1}^{\pm}, u_j^{-}, u_{j+1}^{\pm}, \ldots, u_n^{\pm}] \in \mathcal{P}$ *and* $j \geqslant 1$

(4) $\{SS \,\&\, \{u_1{\in}\mathcal{I}, IK\} \to \{SS \,\&\, [u_1^{-} \mid nil] \,\&\, \{u_1{\in}\mathcal{I}, IK\}$ $\;$ for each $[u_1^{-}, u_2^{\pm}, \ldots, u_n^{\pm}] \in \mathcal{P}$

Backwards semantics

- Forwards oriented transition rules but executed backwards and symbolically

(5) $\{SS\&\ [L \mid M^-, L'] \ \&\ \{M{\in}\mathcal{I}, IK\}\} \rightarrow \{SS\ \&\ [L, M^- \mid L']\ \&\ \{M{\in}\mathcal{I}, IK\}\}$

(6) $\{SS\ \&\ [L \mid M^+, L']\ \&\ \{IK\}\} \rightarrow \{SS\ \&\ [L, M^+ \mid L']\ \&\ \{IK\}\}$

(7) $\{SS\ \&\ [L \mid M^+, L']\ \&\ \{M{\notin}\mathcal{I}, IK\}\} \rightarrow \{SS\ \&\ [L, M^+ \mid L']\ \&\ \{M{\in}\mathcal{I}, IK\}\}$

(8) $\{SS\ \&\ [\, l_1 \mid u^+,\ l_2\,]\ \&\ \{u{\notin}\mathcal{I}, IK\}\} \rightarrow \{SS\ \&\ \{u{\in}\mathcal{I}, IK\}\}$
for each $[l_1, u^+, l_2] \in \mathcal{P}$

Relation between Forwards/Backwards semantics

1. Define a symbolic state

Symbolic P-state

Given a protocol $\mathcal{P}$, a symbolic $\mathcal{P}$-state is a term of the form:

$$S = \{SS \mathbin{\&} [u_1^{\pm},\ldots \mid u_{n_1}^{\pm}]\Theta_1 \mathbin{\&} \ \ldots \mathbin{\&} [u_1^{\pm},\ldots \mid u_{n_k}^{\pm}]\Theta_k \\ \mathbin{\&} \{w_1{\in}\mathcal{I},\ldots,w_m{\in}\mathcal{I},\ w'_1{\notin}\mathcal{I},\ldots,w'_{m'}{\notin}\mathcal{I}, IK\}\}$$

where the $[u_i^{\pm},\ldots \mid u_{n_i}^{\pm}] \ \in \mathcal{P}$

Relation between Forwards/Backwards semantics

2. Define a relation between symbolic (with variables) and concrete states (without variables)

$>^{\Theta}$ relation

Given a symbolic $\mathcal{P}$-state S and a state s we write $S >^{\Theta} s$ iff $\exists \Theta \; : \; Vars(S) - \{SS, IK\} \rightarrow T_{\Sigma}$ s.t.

- $\forall \; [u_1^{\pm}, \ldots, u_j^{\pm} | u_{j+1}, \ldots, u_k^{\pm}] \in \text{strands(S)}$ then $[u_1^{\pm}\Theta, \ldots, u_i^{\pm}\Theta] \in \text{strands(s)}$
- $\forall \; (w{\in}\mathcal{I}) \in knowledge(S)$ then $(w{\in}\mathcal{I}\Theta) \in knowledge(s)$
- $\forall \; (w{\notin}\mathcal{I}) \in knowledge(S)$ then $(w{\in}\mathcal{I}\Theta) \notin knowledge(s)$

and say s is a ground term of forward rewrite theory

Completeness

Theorem

Given a protocol $\mathcal{P}$, two states s, s_0, a symbolic $\mathcal{P}$-state S, a substitution Θ s.t.

- s_0 *is an initial state, and*
- $s_0 \to^n s$, *and*
- $S >^{\Theta} s$

then $\exists$ a symbolic initial $\mathcal{P}$-state S_0, two substitutions μ and Θ', and $k \leqslant n$, s.t.

- $S_0 \overset{k}{\leftsquigarrow}_{\mu} S$, *and*
- $S_0 >^{\Theta'} s_0$

Soundness

Theorem

Given a protocol $\mathcal{P}$, two symbolic $\mathcal{P}$-states S_0, S', an initial state s_0 and a substitution Θ s.t.

- S_0 *is a symbolic initial state,*
- $S_0 >^{\Theta} s_0$, *and*
- $S_0 \overset{*}{\leftsquigarrow} S'$

then $\exists$ a state s' and a substitution Θ', s.t.

- $s_0 \rightarrow^* s'$, *and*
- $S' >^{\Theta'} s'$

Outline

Homomorphic Encryption for Pre-abelian groups

- Formerly unification algorithm only known for homomorphic encryption for a binary operation without axioms, did not have FVP
- Pre-abelian Group = Abelian Group without associativity
- different keys can be used, and for each encryption is homomorphic
- Theory satisfies FVP and, thus, has finitary and complete unification algorithm
- Commonly occurring in protocols with indistinguishability properties

Equational Theory

$$hpke(X, U) + hpke(Y, U) = hpke(X + Y, U)$$
$$-(hpke(X, U)) = hpke(-(X), U)$$
$$hpke(0, U) = 0$$
$$hpke(hpke(X, V), U) = hpke(X, U \ \& \ V)$$
$$hpke(X, U \ \& \ V) + hpke(Y, U) = hpke(hpke(X, V) + Y, U)$$
$$hpke(X, U) + hpke(Y, U \ \& \ V) = hpke(X + hpke(Y, V), U)$$
$$hpke(X, U \ \& \ V) + hpke(Y, U \ \& \ W) = hpke(hpke(V, X) + hpke(W, Y), U)$$

$$hske(0, U) = 0$$
$$hske(hpke(X, U), U) = X$$
$$hske(hpke(X, U \ \& \ V), U) = hpke(X, V)$$
$$hske(hpke(X, U), U \ \& \ W) = hske(X, W)$$
$$hske(hpke(X, U \ \& \ V), U \ \& \ W) = hske(hpke(X, V), W)$$

$$X + Y = Y + X \qquad (\textbf{axiom})$$
$$X + 0 = X$$
$$X + -(X) = 0$$
$$-(-(X)) = X$$
$$-(0) = 0$$

Future Directions on Homomorphic Encryption

- Approximate associativity by adding additional equations
- Investigate new theories for homomorphic encryption on abelian groups
- Apply these to analyze indistinguishability properties of protocols

Outline

Research challenges

- Equational theories:
  - Cancellation of encryption and decryption [Done] (e.g. EKE)
  - Exponentiation [Done] (e.g. Diffie-Hellman)
  - Exclusive-OR [On-going]
  - Homomorphism [On-going]
- After experimentation, we have discovered that state space reduction techniques need to be reformulated to take properties of paired protocols into account in order to reduce the search space
  - Grammars have been already extended with successful results in experiments (Next slide)
  - More optimizations needed to improve state space reduction

Extending grammars for indistinguishability

1. Given a synchronous product $\mathcal{P}_1 \otimes \mathcal{P}_2$, generate grammars for $\mathcal{P}_1$ and $\mathcal{P}_2$ separately.
   - Example: EKE

$$\mathcal{G}_{\mathcal{P}_1} = \left\{ \begin{array}{l} (\ grl\ M\ inL\ => \ penc(PK, M)\ inL.; \\ \ grl\ M\ inL\ => \ pdec(inv(PK), M)\ inL.; \\ \ grl\ M\ inL\ => \ senc(SK, M)\ inL.; \\ \ grl\ M\ inL\ => \ sdec(SK, M)\ inL.; \\ \ grl\ M\ inL\ => \ enc(PW, M)\ inL.; \\ \ grl\ M\ inL\ => \ dec(PW, M)\ inL.; \\ \ grl\ M\ inL\ => \ (M; M')\ inL.; \\ \ grl\ M\ inL\ => \ (M'; M)\ inL.; \\ \ grl\ pw(N1, N2)\ notLeq\ pw(i, N3) \\ \quad => \ pw(N1, N2)\ inL.)\ |\ [\ldots] \end{array} \right.$$

$$\mathcal{G}_{\mathcal{P}_2} = \left\{ \begin{array}{l} (\ grl\ M\ inL\ => \ penc(PK, M)\ inL.; \\ \ grl\ M\ inL\ => \ pdec(inv(PK), M)\ inL.; \\ \ grl\ M\ inL\ => \ senc(SK, M)\ inL.; \\ \ grl\ M\ inL\ => \ sdec(SK, M)\ inL.; \\ \ grl\ M\ inL\ => \ enc(PW, M)\ inL.; \\ \ grl\ M\ inL\ => \ dec(PW, M)\ inL.; \\ \ grl\ M\ inL\ => \ (M; M')\ inL.; \\ \ grl\ M\ inL\ => \ (M'; M)\ inL.; \\ \ grl\ pw(N1, N2)\ notLeq\ pw(a, b) \\ \quad => \ pw(N1, N2)\ inL.)\ |\ [\ldots] \end{array} \right.$$

Extending grammars for indistinguishability

2. The grammars of $\mathcal{P}_1 \otimes \mathcal{P}_2$, will be a "paired" version of the union of grammars generated for $\mathcal{P}_1$ and $\mathcal{P}_2$ separately.

$$
\mathcal{G}_{\mathcal{P}_1 \otimes \mathcal{P}_2} = \left\{
\begin{array}{l}
(\ grl\ (M \otimes X)\ inL\ {=}{>}\ (penc(PK, M) \otimes X)\ inL.;\\
\ grl\ (M \otimes X)\ inL\ {=}{>}\ (pdec(inv(PK), M) \otimes X)\ inL.;\\
\ grl\ (M \otimes X)\ inL\ {=}{>}\ (senc(SK, M) \otimes X)\ inL.;\\
\ grl\ (M \otimes X)\ inL\ {=}{>}\ (sdec(SK, M) \otimes X)\ inL.;\\
\ grl\ (M \otimes X)\ inL\ {=}{>}\ (enc(PW, M) \otimes X)\ inL.;\\
\ grl\ (M \otimes X)\ inL\ {=}{>}\ (dec(PW, M) \otimes X)\ inL.;\\
\ grl\ (M \otimes X)\ inL\ {=}{>}\ ((M\ ;\ M') \otimes X)\ inL.;\\
\ grl\ (M \otimes X)\ inL\ {=}{>}\ ((M'\ ;\ M) \otimes X)\ inL.;\\
\ grl\ (pw(N1, N2) \otimes X)\ notLeq\ (pw(a, b) \otimes X)\\
\quad {=}{>}\ \ ((pw(N1, N2) \otimes Z)\ inL.)\\
\ |\\
(\ grl\ (X \otimes M)\ inL\ {=}{>}\ (X \otimes penc(PK, M))\ inL.;\\
\ grl\ (X \otimes M)\ inL\ {=}{>}\ (X \otimes pdec(inv(PK), M))\ inL.;\\
\ grl\ (X \otimes M)\ inL\ {=}{>}\ (X \otimes senc(SK, M))\ inL.;\\
\ grl\ (X \otimes M)\ inL\ {=}{>}\ (X \otimes sdec(SK, M))\ inL.;\\
\ grl\ (X \otimes M)\ inL\ {=}{>}\ (X \otimes enc(PW, M))\ inL.;\\
\ grl\ (X \otimes M)\ inL\ {=}{>}\ (X \otimes dec(PW, M))\ inL.;\\
\ grl\ (X \otimes M)\ inL\ {=}{>}\ (X \otimes (M\ ;\ M'))\ inL.;\\
\ grl\ (X \otimes M)\ inL\ {=}{>}\ (X \otimes (M'\ ;\ M))\ inL.;\\
\ grl\ (X \otimes pw(N1, N2))\ notLeq\ (Z \otimes pw(a, b))\\
\quad {=}{>}\ \ (X \otimes pw(N1, N2))\ inL.)\ |\ [\ldots]
\end{array}
\right.
$$

Outline

Conclusions

- Have developed new theoretical foundations and new analysis techniques for indistinguishability modulo algebraic properties
- Have used Maude-NPA to experiment with these techniques on simple examples
- Have begun experimenting with more complex examples with development version of Maude-NPA
- In future plan to
  - Further develop the theoretical foundations
    - How does our definition of indistinguishability relate to others?
  - Investigate how to adapt state space reduction techniques to $\otimes$ theory
  - Explore use of Maude-NPA on more privacy-preserving protocols