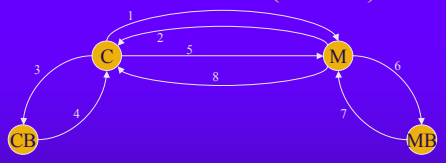
Four Attacks on an Anonymous Fair Exchange E-commerce Protocol

Adam Barth
Andrew Tappert
CS259

Protocol Overview

- Protocol proposed in Ray and Ray 2001
- Five roles
  - Customer and customer's bank
  - Merchant and merchant's bank
  - Trusted third party
- Allows anonymous fair exchange of money for a digital good
- Identities protected by single-transaction public/private key pairs
- Customer assured of obtaining correct product by cross validation (not relevant for our analysis)

Protocol Overview (no TP)



Preamble (on a private channel) M => TP: m K1 Mipub
Preamble (on a private channel) TP => C: [m, K1] Mipub

1) C => M: PO [CC(PO), Ciprv] [Cipub, Mipub]
2) M => C: [CC(PO), Miprv] [m.r, K1xK2] [CC([m.r, K1xK2]), Miprv] [r, K1] [CC([r, K1]), Miprv] [Macct, MBpub] [CC([Macct, MBpub]), Miprv]
3) C => CB: [[MTI, Cprv], CBpub]
4) CB => C: [[P, Bcprv], Cpub]
5) C => M: [[P, Bcprv], Mipub]
6) M => MB: [[P, Bcprv], MBpub]
7) MB => M: [ack, MBprv]
8) M => C: [K2inv, Cipub] [CC(K2inv), Miprv] [rinv, Cipub] [CC(rinv), Miprv]

Attack #1: Malicious Bank

- Neither M nor MB can learn creator of P as such knowledge compromises C's anonymity
- Bcprv is a shared private key among banks
- Thus, any bank can create [[P, Bcprv], Mipub]
- A malicious bank can play the role of customer and obtain the good, but not make good on P
- Neither M nor MB can learn the identity of the malicious bank
- Defense: validity of payment token is a larger issue, not clear how to fix simply

Attack #2: Man in the Middle

- Customer's public/private key pair fresh
- Ciprv/Cipub only occur in messages 1 and 8
  - C => M: po [CC(po), Ciprv] [Cipub, Mipub]
  - M => C: [k2inv, Cipub] [CC(k2inv), Miprv] [rinv, Cipub] [CC(rinv), Miprv]
- Ciprv/Cipub never signed by any role
- Intruder may replace Ciprv/Cipub
- Intruder learns the digital good
- Intruder cannot relay message 8 to C, but C can invoke TP to receive product
- Defense: add [CC(Cipub), Miprv] to message 2

Extended Protocol with TP

- We assume resilient private channels with TP
- Only the customer may invoke the TP
  - C => TP: message 1, message 2, [P, Bcprv]
  - TP => M: "Please send product decryption key for PO"
  - *Option 1 (if M already has [P, Bcprv])*
    - M => TP: k2inv, rinv
    - TP => C: k2inv, rinv
  - *Option 2 (if M does not have [P, Bcprv])*
    - M => TP: "I did not receive payment token"
    - TP => M: [P, Bcprv] resume base protocol with message 6
  - *Option 3 (if timeout occurs)*
    - No response from merchant
    - TP => C: K1inv

Attack #3: Dishonest Merchant

- M can receive payment and not send good
- C may invoke the trusted party
- M can claim payment was not received
- TP forwards P and base protocol resumes
- M can still not send product
- Defense: add state to TP and disallow option 2 after the first time TP invoked

Attack #4: Unbalance for C

- Only C can invoke the trusted party
- After receiving [P, Bcprv] from CB, C can either force the transaction to occur or abort
- C can prove to another party that s/he can force transaction, but cannot prove s/he can force abort
- Once M sends message 2 s/he is committed to the transaction and cannot abort
- Maybe M does not care?

Methods

- We modeled this protocol using MOCHA
- We discovered these attack by hand while creating the formal models
- MOCHA found trace based attacks 1 and 2
- Unable to model TP due to MOCHA bug
- We modeled simplified TP
- Attack 4 should be detectable with ATL
- MOCHA ran for 150 hours with no answer

Questions?