Model Checking

An overview, A comparison, A look ahead, A man a plan....

Covering...

- What it takes to describe a protocol
- High Level Descriptions of two checkers
- Quick comparison of the two
- Our Old friend Needham-Schroeder
- Closing remarks

Description of a checker

- Adversary capabilities
- How messages are treated
- Properties checked

Adversary capabilities

a) listen capability
b) keep track of things they have seen
    0) nonces
    1) keys
    2) encrytped messages
c) ability to send messages (good and fake)

What is a message

In a protocol there is meta information that we need to provide explicitly to the model.

source
dest
key(s)
messageType
nonce(s)

What are we checking for?

- Authentication
- Encryption
- Monitoring the messages and reading them

Descriptions (high level) of each model

- Mur-Phi
- FDR

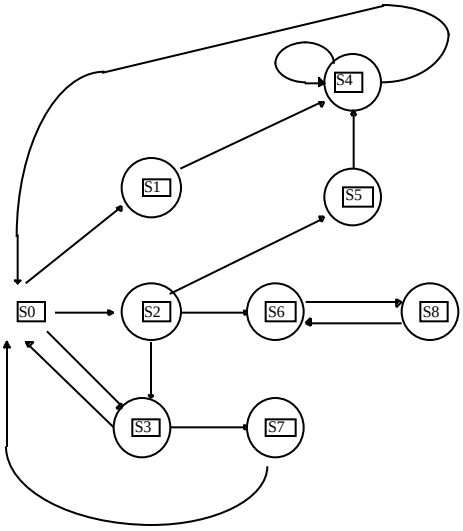
Mur-phi

- Describes the protocol being checked via a FSM
- It “moves” around by changing states (if foo state then bar must be be in such a state)
- Explicit state enumeration (ie brute force)
- Nondeterministic (attacker can choose different messages to reply to)

Steps for mur-phi

- Formulate Protocol
- Add an Adversary the to system
- State the desired correctness condition
- Run the protocol
- Change forumlations and repeat

Example mur-phi FSM



Failures Divergences Refinement Checker (aka FDR)

- Describes the protocol being checked via a process
- It “moves” around by receiving a message that triggers an action
- Basically enumeration again
- Nondeterministic (attacker can choose different messages to reply to)

Steps for FDR

- Modeled in Communicating Sequential Processes (CSP)
- Uses sets to describe the protocol (Initiator, Responder, Key, Nonces)
- Describe the Protocol in CSP

Simililarities

- Enumeration
- Adversary description
- FDR can be turned into an FSM
- Nondeterminstic
- Break on a large amount of particiapants

Differences

- Mur-phi has lots of work to make the FSM (ie no real high level language to do automatically)
- FDR is more abstracted (I can say send a message from foo to bar)
- Could say Low Level Vs High Level

Needham-Schroeder

- A -> B : A.B{Na • A}$_{PK(B)}$
- B -> A : B.A{Na • Nb}$_{PK(A)}$
- A -> B : A.B{Nb}$_{PK(B)}$

Model Checking Sucess

- Both find the following attack
- A -> I : A.I{Na • A}$_{PK(I)}$
- I(A) -> B : A.B{Na • A}$_{PK(B)}$
- B -> I(A) :B.A{Na•Nb}$_{PK(A)}$
- I -> A :I.A{Na•Nb}$_{PK(A)}$
- A -> I : A.I{Nb}$_{PK(I)}$
- I(A) -> B : A.B{Nb}$_{PK(B)}$

More success

- Both the Mur-phi and FDR find all the errors in protocols tested
- Both find it in an acceptable time (10 minutes for the Mur-phi, no times given for FDR tho can assume close)

Not the Catch all be all

- Both only work for small number of particiapants (ie 2)
- Adversary descriptions problems
- Can’t prove that it is “correct” for a bigger number of particiapants
- Has some problems with malleability {M1 • M2}Ki = {M1}ki • {M2}ki

So why use them?

- You can prove that if I have a large particiapants and then take a smaller particiapants and find error that the error will hold

Computers are better enumerators than humans

- Finding replay attacks by enumeration
- Some distributed system looking for a long time might find something