# Logic for Computer Security Protocols

## John Mitchell Stanford University

# Outline

- Example
  Floyd-Hoare logic of programs
- ♦BAN logic
- ◆Current Protocol Logic

# Part I

# Logic of programs

Historical references: Floyd, ... Hoare, ...

# **Before-after assertions**

#### ◆Main idea

- F <P> G
- If F is true before executing P, then G after

#### ♦Two variants

- Total correctness F [P] G
   If F before, then P will halt with G
- Partial correctness F {P} G
- If F before, and *if P halts*, then G

# While programs

#### ♦Programs

P ::= x := e | P;P | if B then P else P while B do P

## where x is any variable

- e is any integer expression
- B is a Boolean expression (true or false)

# Assertion about assignment

Assignment axiom
F(t) { x := t } F(x)

### ♦Examples

7=7	{ x := 7 }	x=
(y+1)>0	{ x := y+1 }	X>
	{ x := x+1 }	

This is not most general case. Need to assume no aliasing...

# Rule of consequence

◆If
• F { P } G
◆ And
• F' → F and G → G'
◆ Then
• F' { P } G'

# Example

y>0 { x	::= y+1 } x>0	
♦Proof		
(y+1)>0 { x	:= y+1 } x>0	(assignment axiom)
y>0 { x	:= y+1 } x>0	(consequence)
Assertion		
x=1 { x :	= x+1 } x=2	
Proof		
x+1=2 {	x := x+1 } x=3	2 (assignment axiom)
x=1 {	x := x+1 } x=	2 (consequence)

# Conditional

F ^ B { P<sub>1</sub> } G F ^¬B { P<sub>2</sub> } G

F { if B then  $P_1$  else  $P_2$  } G

#### ♦Example

true { if  $y \ge 0$  then x := y else x := -y }  $x \ge 0$ 

# Sequence

F {P<sub>1</sub>}G G {P<sub>2</sub>}H

F { P<sub>1</sub>; P<sub>2</sub> } H

◆Example
x=0 {x := x+1; x := x+1} x=2

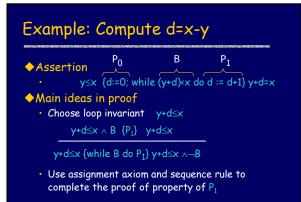
# Loop Invariant

F ∧ B { P } F

F { while B do P } F  $\land \neg B$ 

#### ♦Example

true { while  $x \neq 0$  do  $x \coloneqq x-1$  } x=0



# Facts about Hoare logic

#### Compositional

- Proof follows structure of program
- ♦ Sound
- "Relative completeness"
  - Properties of computation over N provable from properties of N
  - Some technical issues ...
- Important concept: Loop invariant !!!

# Part II

# **BAN Logic**

# There is something called BAN

#### ♦Needham

"The main contribution of BAN logic was to make the study of 3-line protocols intellectually respectable."

#### Paper,

A Logic of Authentication", ACM Transactions on Computer Systems, Vol. 8, No. 1, pp. 18-36, February 1990.

# Using BAN Logic

- Protocol expressed in "idealized" form
- Identify initial assumptions in the language of BAN logic
- •Use postulates and rules of BAN logic to deduce new predicate

## Notation

#### P |≡X: P believes X

- P would be entitled to believe X.
- The principal P may act as though X is true.

#### P ∜X: P sees X

- P can read the contents of X (possibly after decryption, assuming P has the needed keys)
- P can include X in messages to other principals

# **BAN Logic**

#### P |~ X P once said X

- P sent a message including the statement X.
- Possibly in the past or in the current run of the protocol
- P believed that X was true when it send the message

## $P \Rightarrow X$

- P controls X • P has jurisdiction over X
- P is a trusted authority on the truth of X.

#### #(X) X is fresh

- The present begins with the start of the current execution of the current protocol
  X is fresh if it is not contained in any message in the past

# BAN Logic K P ↔ Q: K is a shared key for P and Q. K is a secure key for communication between P and Q. K will never be discovered by any principal except for P or Q, or a principal trusted by either P or Q. K → P K is a public key for P The matching secret key (the inverse of K, denoted by K-1) will never be discovered by any principal except P, or a principals trusted by P