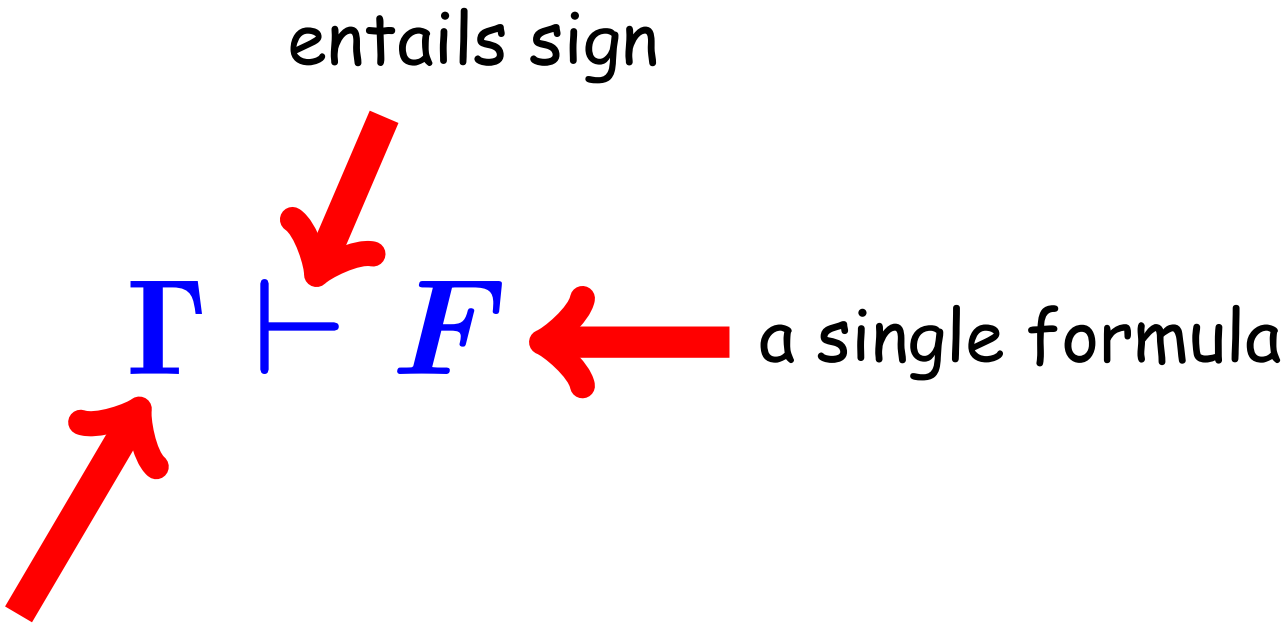
Access Control and Privacy Policies (7)

Email: christian.urban at kcl.ac.uk
Office: S1.27 (1st floor Strand Building)
Slides: KEATS (also homework is there)

Judgements

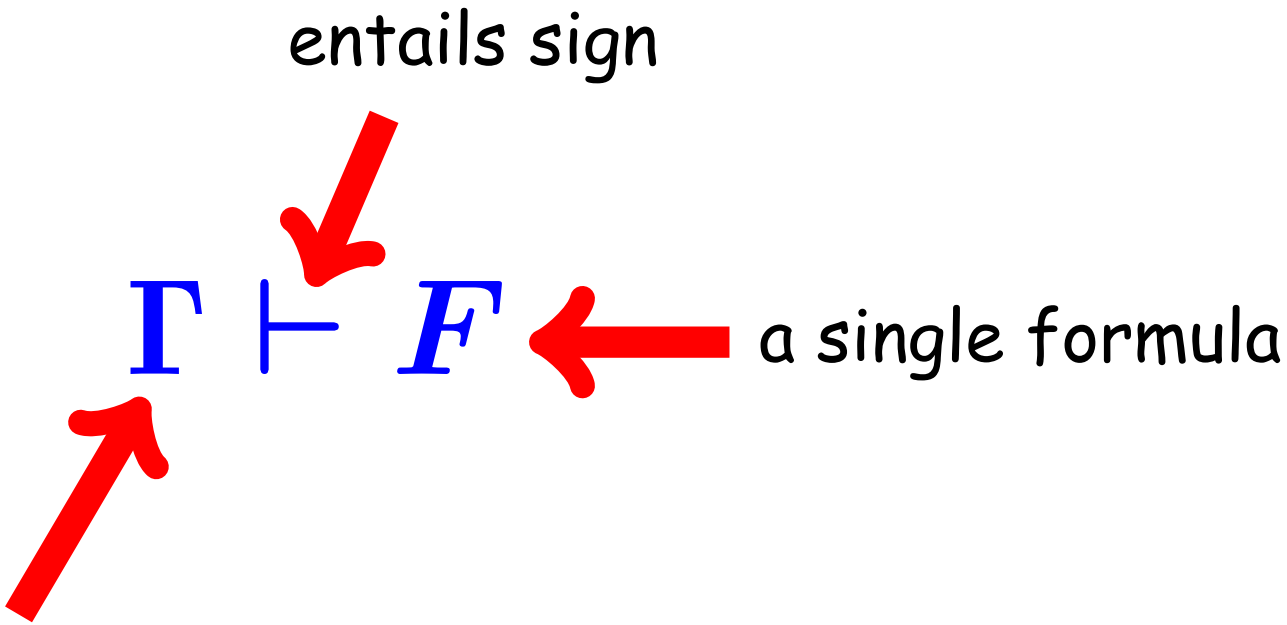
$$\Gamma \vdash F$$

Judgements



Gamma
stands for a collection of formulas
("assumptions")

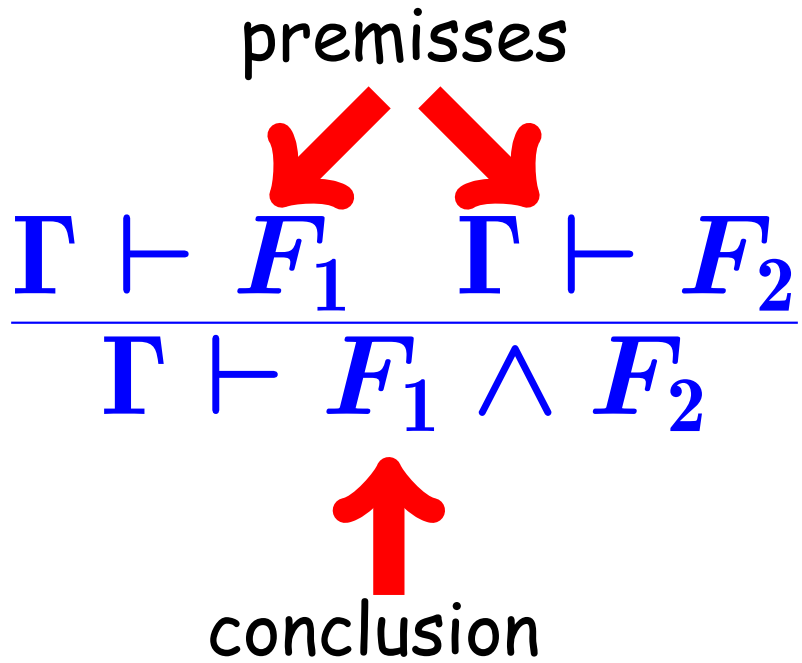
Judgements



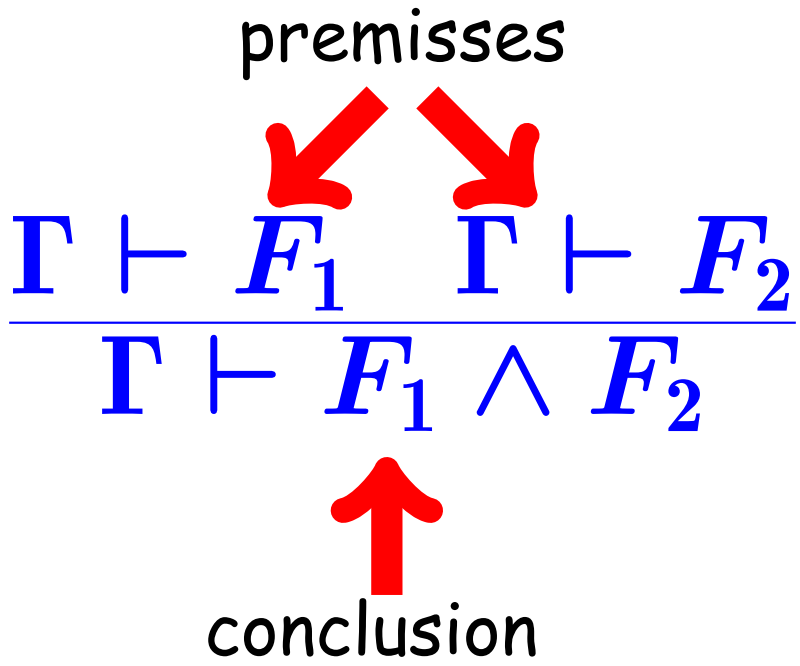
Gamma
stands for a collection of formulas
("assumptions")

Gimel (Phoenician), Gamma (Greek), C and G (Latin), Gim (Arabic),
?? (Indian), Ge (Cyrillic)

Inference Rules

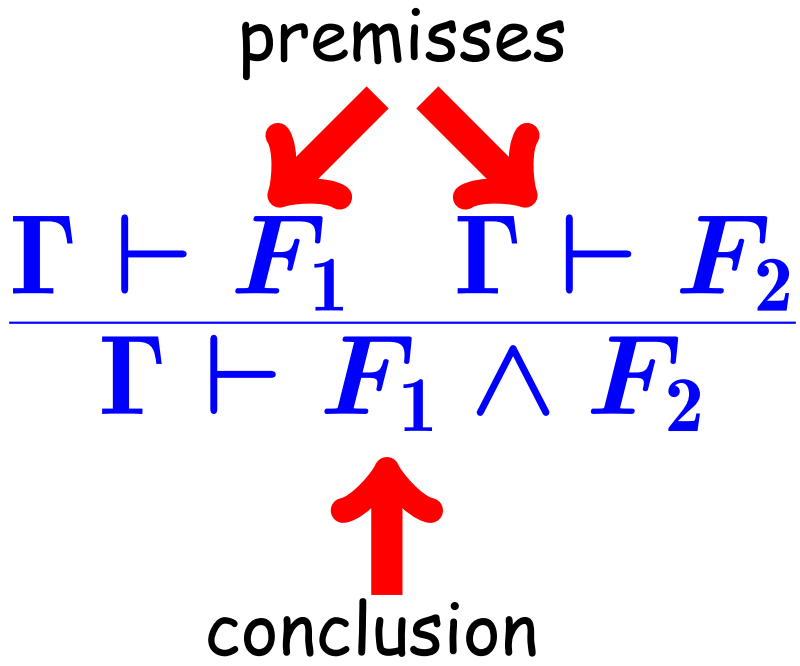


Inference Rules



P says $F \vdash Q$ says $F \wedge P$ says G

Inference Rules



$$\underbrace{P \text{ says } F}_{\Gamma} \vdash \underbrace{Q \text{ says } F}_{F_1} \wedge \underbrace{P \text{ says } G}_{F_2}$$

$$\frac{\Gamma \vdash F_1 \Rightarrow F_2 \quad \Gamma \vdash F_1}{\Gamma \vdash F_2}$$

$$\frac{\Gamma \vdash F}{\Gamma \vdash P \text{ says } F}$$

Digression: Proofs in CS

Formal proofs in CS sound like science fiction?

Digression: Proofs in CS

Formal proofs in CS sound like science fiction?
Completely irrelevant!

Digression: Proofs in CS

Formal proofs in CS sound like science fiction?

Completely irrelevant!

- in 2008, verification of a small C-compiler
- in 2010, verification of a micro-kernel operating system (approximately 8700 loc)
  - 200k loc of proof
  - 25 - 30 person years
  - found 160 bugs in the C code (144 by the proof)

Bob Harper
(CMU)

Frank Pfenning
(CMU)

published a proof about a specification in a journal (2005), ∼31pages

Bob Harper
(CMU)

Frank Pfenning
(CMU)

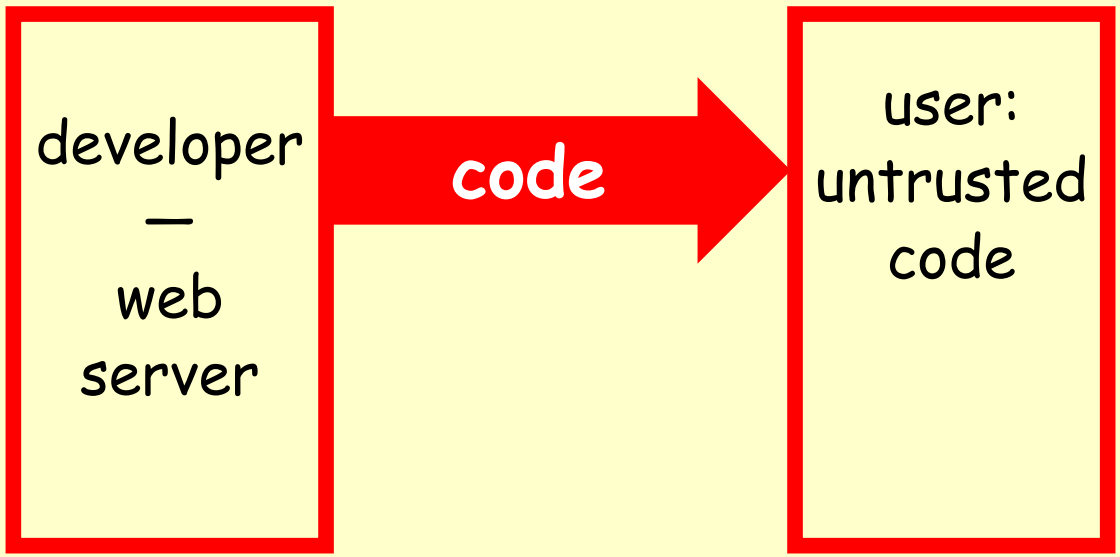
published a proof about a specification in a journal (2005), ∼31pages

Andrew Appel
(Princeton)

relied on their proof in a **security** critical application

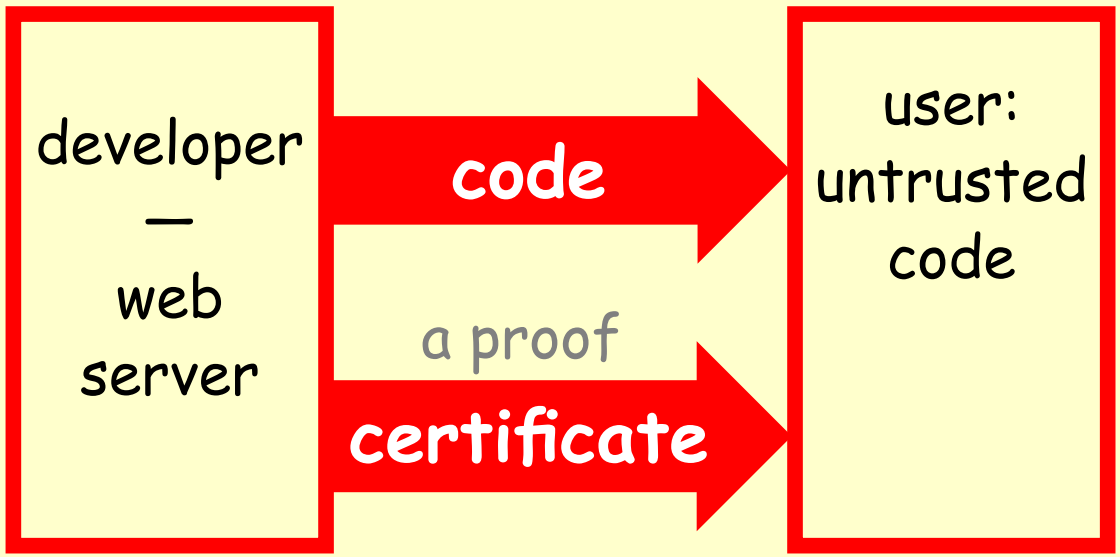
Proof-Carrying Code

Idea:



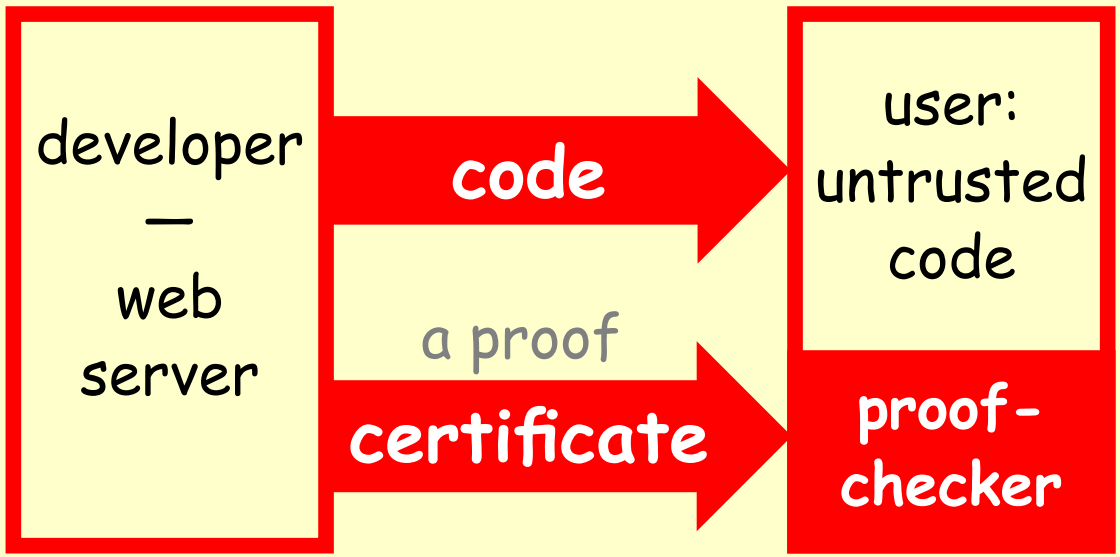
Proof-Carrying Code

Idea:



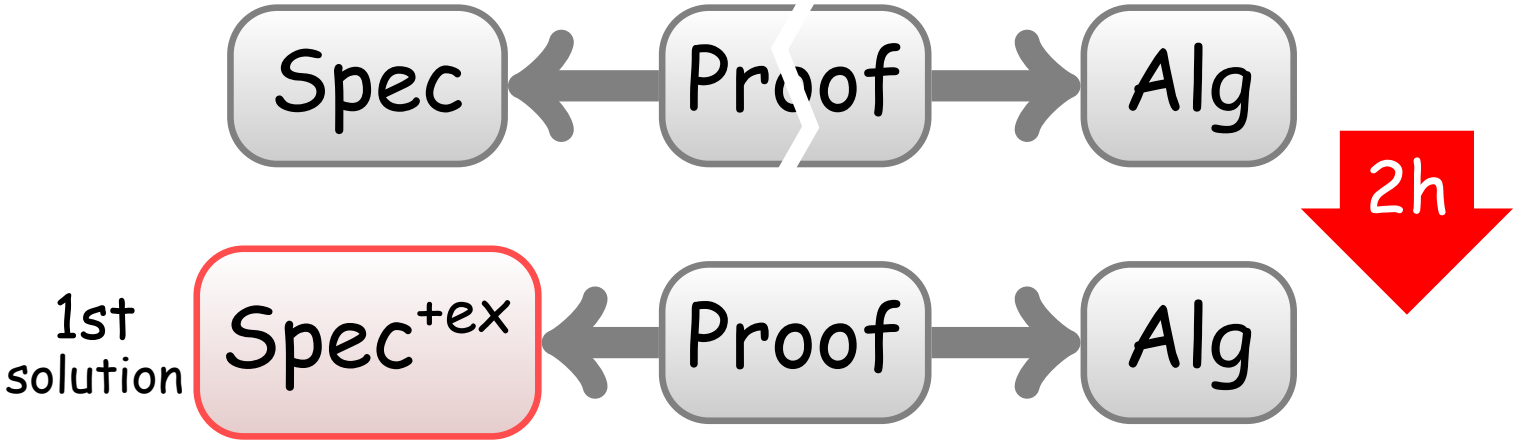
Proof-Carrying Code

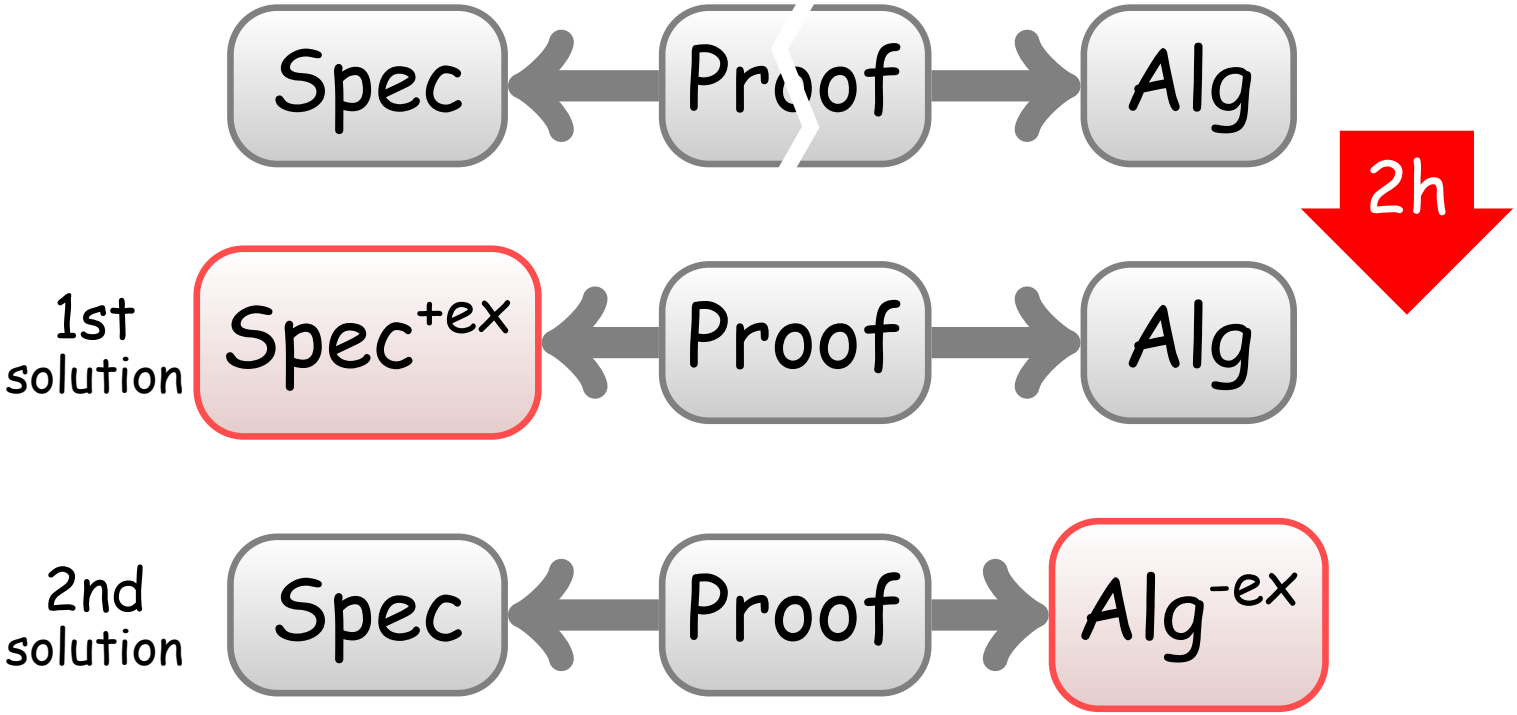
Idea:

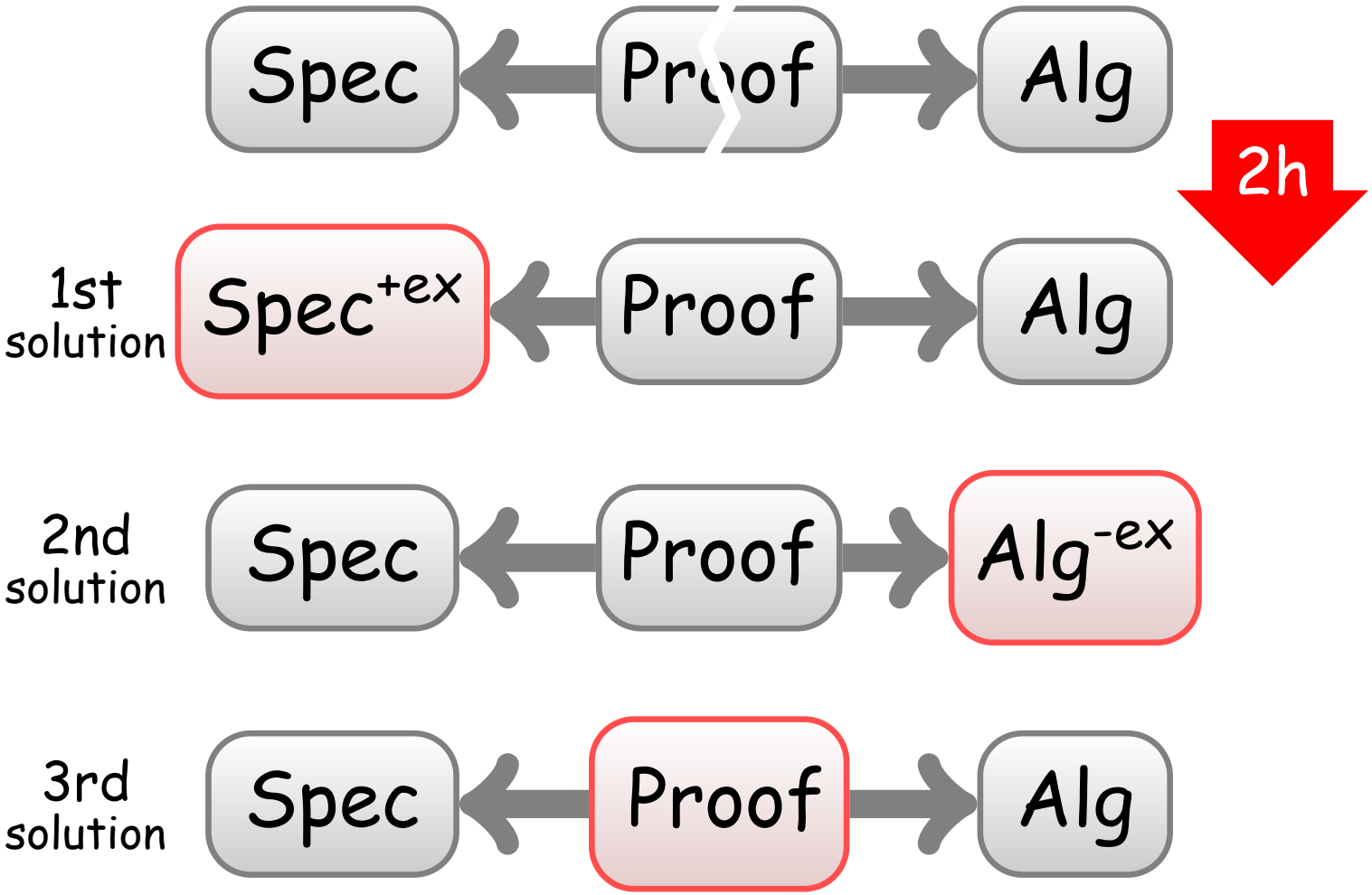



Spec
Proof
Alg

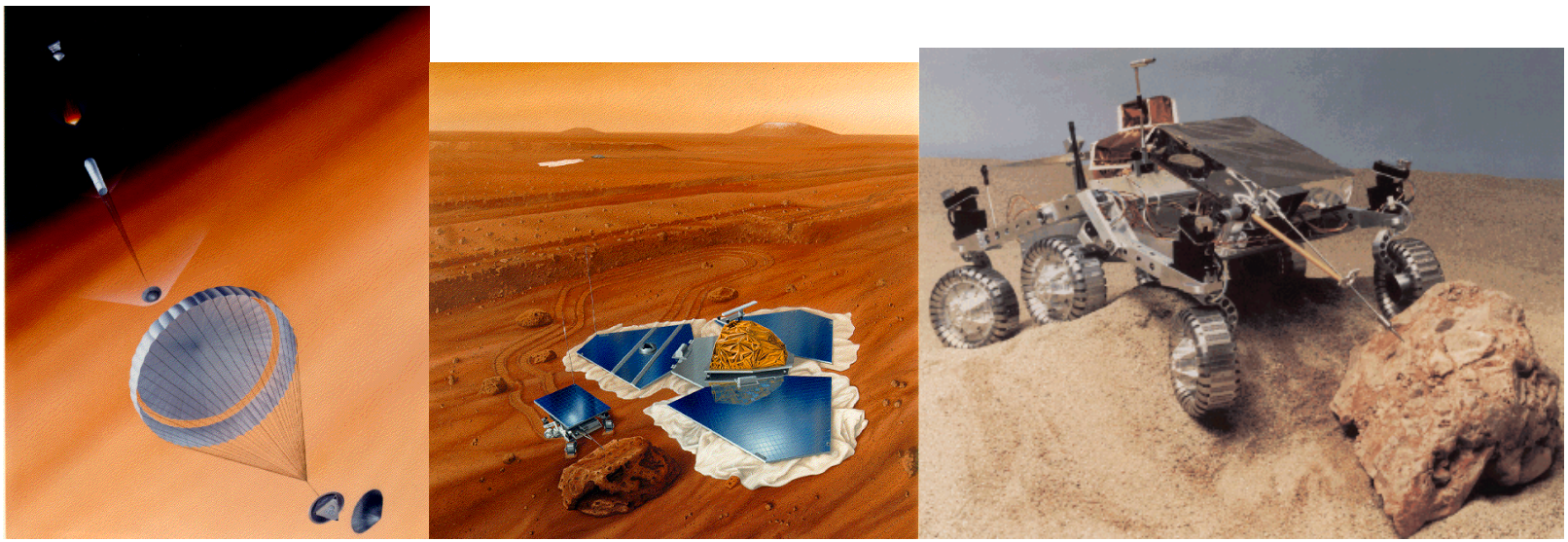

Spec
Proof
Alg


Spec
Proof
Alg
2h
1st solution
Spec+ex
Proof
Alg


Spec
Proof
Alg
2h
1st solution
Spec+ex
Proof
Alg
2nd solution
Spec
Proof
Alg-ex


Spec
Proof
Alg
2h
1st solution
Spec+ex
Proof
Alg
2nd solution
Spec
Proof
Alg-ex
3rd solution
Spec
Proof
Alg

Mars Pathfinder Mission 1997



- despite NASA's famous testing procedure, the lander crashed frequently on Mars
- problem was an algorithm not used in the OS

Trusted Third Party

Simple protocol for establishing a secure connection via a mutually trusted 3rd party (server):

Message 1 $A \rightarrow S : A, B$

Message 2 $S \rightarrow A : \{K_{AB}\}_{K_{AS}}$ and $\{\{K_{AB}\}_{K_{BS}}\}_{K_{AS}}$

Message 3 $A \rightarrow B : \{K_{AB}\}_{K_{BS}}$

Message 4 $A \rightarrow B : \{m\}_{K_{AB}}$

Encrypted Messages

- Alice sends a message m

Alice says m

Encrypted Messages

- Alice sends a message m

  Alice says m

- Alice sends an encrypted message m (with key K)

  Alice says $\{m\}_K$

Encrypted Messages

- Alice sends a message m

$$\text{Alice says } m$$

- Alice sends an encrypted message m (with key K)

$$\text{Alice says } \{m\}_K$$

- Decryption of Alice's message

$$\frac{\Gamma \vdash \text{Alice says } \{m\}_K \quad \Gamma \vdash \text{Alice says } K}{\Gamma \vdash \text{Alice says } m}$$

Encryption

- Encryption of a message

$$\frac{\Gamma \vdash \text{Alice says } m \quad \Gamma \vdash \text{Alice says } K}{\Gamma \vdash \text{Alice says } \{m\}_K}$$

Trusted Third Party

- Alice calls Sam for a key to communicate with Bob
- Sam responds with a key that Alice can read and a key Bob can read (pre-shared)
- Alice sends the message encrypted with the key and the second key it recieved

$$
\begin{array}{lcl}
A \text{ sends } S & : & \text{Connect}(A, B) \\
S \text{ sends } A & : & \{K_{AB}\}_{K_{AS}} \text{ and } \{\{K_{AB}\}_{K_{BS}}\}_{K_{AS}} \\
A \text{ sends } B & : & \{K_{AB}\}_{K_{BS}} \\
A \text{ sends } B & : & \{m\}_{K_{AB}}
\end{array}
$$

Sending Rule

$$\frac{\Gamma \vdash P \text{ says } F \quad \Gamma \vdash P \text{ sends } Q : F}{\Gamma \vdash Q \text{ says } F}$$

Sending Rule

$$\frac{\Gamma \vdash P \text{ says } F \quad \Gamma \vdash P \text{ sends } Q : F}{\Gamma \vdash Q \text{ says } F}$$

$$P \text{ sends } Q : F \stackrel{\text{def}}{=} (P \text{ says } F) \Rightarrow (Q \text{ says } F)$$

Trusted Third Party

A sends S : Connect(A, B)

S says (Connect$(A, B) \Rightarrow$
$\{K_{AB}\}_{K_{AS}} \wedge \{\{K_{AB}\}_{K_{BS}}\}_{K_{AS}}$)

S sends A : $\{K_{AB}\}_{K_{AS}} \wedge \{\{K_{AB}\}_{K_{BS}}\}_{K_{AS}}$

A sends B : $\{K_{AB}\}_{K_{BS}}$

A sends B : $\{m\}_{K_{AB}}$

Trusted Third Party

A sends S : Connect(A, B)

S says (Connect$(A, B) \Rightarrow$
$\{K_{AB}\}_{K_{AS}} \wedge \{\{K_{AB}\}_{K_{BS}}\}_{K_{AS}}$)

S sends A : $\{K_{AB}\}_{K_{AS}} \wedge \{\{K_{AB}\}_{K_{BS}}\}_{K_{AS}}$

A sends B : $\{K_{AB}\}_{K_{BS}}$

A sends B : $\{m\}_{K_{AB}}$

$\Gamma \vdash B$ says m?

Challenge-Response Protocol

- an engine E and a transponder T share a key K
- E sends out a nonce N (random number) to T
- T responds with $\{N\}_K$
- if E receives $\{N\}_K$ from T then starts engine

Challenge-Response Protokol

E says N (start)

E sends $T : N$ (challenge)

$(T \text{ says } N) \Rightarrow (T \text{ sends } E : \{N\}_K \wedge$
$T \text{ sends } E : \mathtt{Id}(T))$ (response)

T says K (key)

T says $\mathtt{Id}(T)$ (identity)

$(E \text{ says } \{N\}_K \wedge E \text{ says } \mathtt{Id}(T)) \Rightarrow$
$\mathtt{start\_engine}(T)$ (engine)

$\Gamma \vdash \mathtt{start\_engine}(T)$?