Summer School on the Foundations of Security, University of Oregon, June 16—26 2003

Cryptographic Protocols

Cédric Fournet
Microsoft Research

Contents

Main subject:
cryptographic protocols
for distributed communications

Tools from concurrent programming theory:
the applied pi calculus

Detailed applications and examples:
private authentication
key-exchange for IPSEC (JFK)
web services security
secure implementations

Just Fast Keying ?

W. Aiello, S.M. Bellovin, M. Blaze, R. Canetti, J. Ionnidis, A.D Keromytis, and O. Reingold. Efficient, DoS Resistant, Secure Key Exchange for Internet Protocols. In *ACM Conference on Computer and Communications Security (CCS'02)*, November 2002.

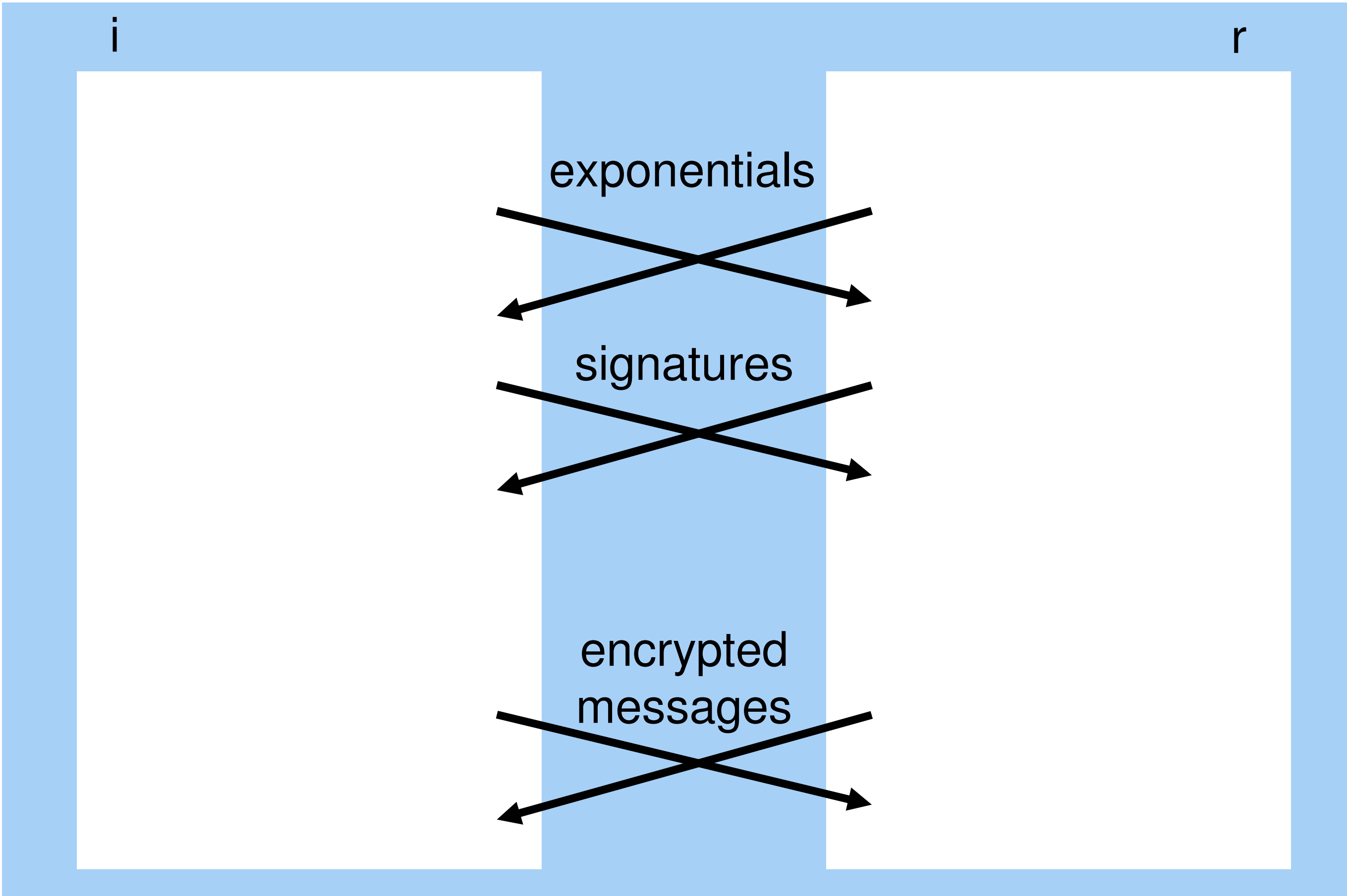
Session establishment (again)

- Two parties want to open a secure session
  - Telnet (SSH)
  - Web connection (SSL, TLS)
  - IP tunnel (VPN)
  - Wireless network
- They need to
  - Generate a shared secret (the "session key")
  - Agree on many parameters
  - Verify each other's identity
- Attackers might eavesdrop, delete, and insert messages, may impersonate principals,… in order to
  - gain information
  - confuse or hinder the participants

Building blocks

- Shared-key encryption
- Cryptographic hash (HMAC)
- Tokens (or cookies)
- Diffie-Hellman computation
- Public-key signature

Two-round Diffie-Hellman



- Against active attackers,
  first create a shared key, then authenticate

Complications

- Configuration
  - Different security needs according to the application
  - Many cryptographic algorithms to choose from
  - Many flavours of authentication (PKIs)
  - Different modes
- Concurrency
  - Parallel sessions
  - Various principals using several shared proxies
- Efficiency concerns
  - Round-trips are expensive
  - Cryptography can be expensive
- Session management
  - Key derivation
  - Rekeying
  - Dead peer detection

IKE and its successors

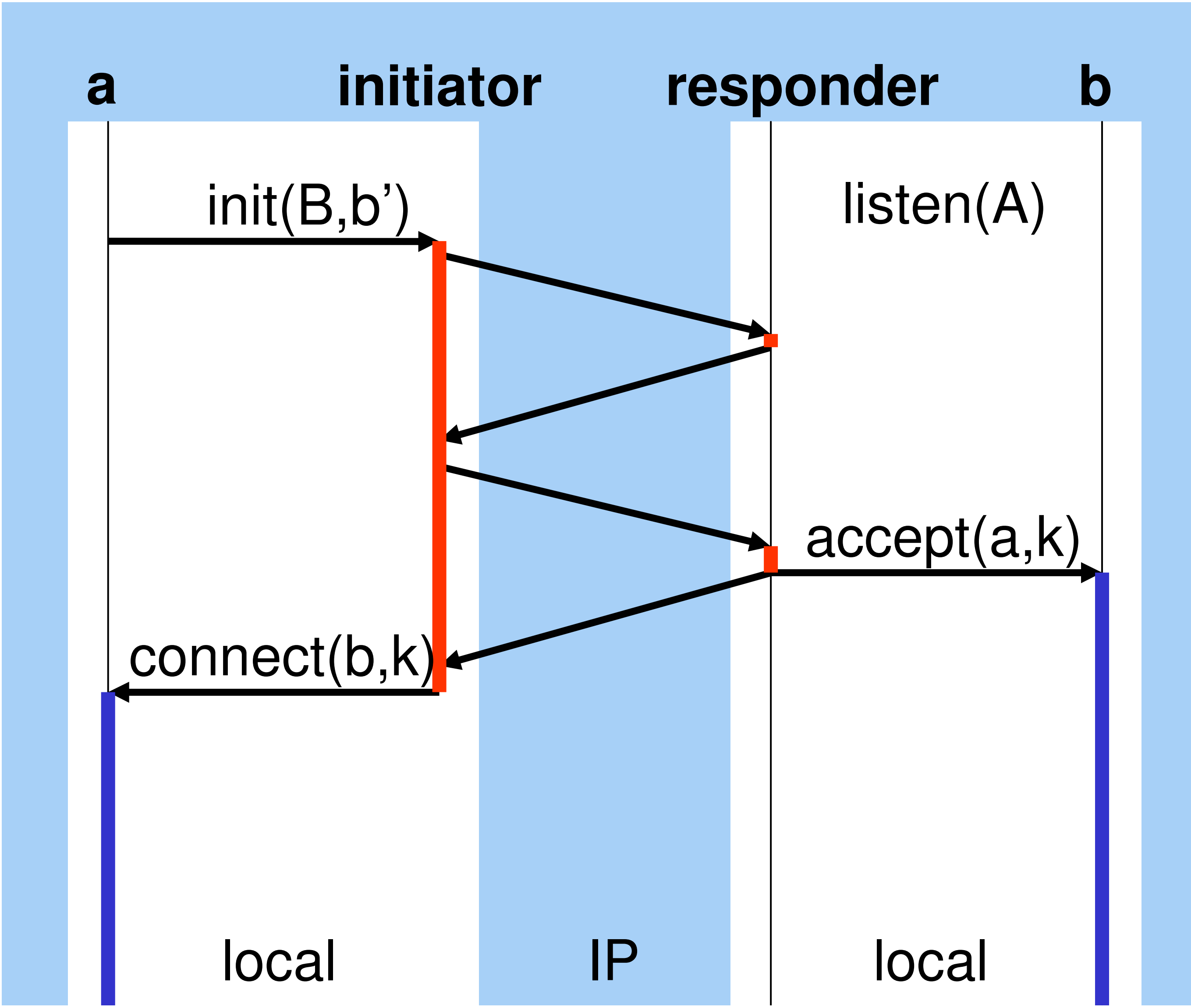
- IKE (Internet Key Exchange)
  - Session management for IPSEC
  - Quite secure
  - Some concerns
    - Too complicated
    - Inefficient (too many messages & expensive operations)
    - Poor resistance against denial of service
- The IETF is considering a successor for IKE,
  (now merging the different proposals into IKEv2)
- JFK (Just Fast Keying) is a simple proposal that incorporates several new mechanisms.
  http://www.ietf.org/internet-drafts/draft-ietf-ipsec-jfk-04.txt

Design goals for JFK

- Security
  "The key should be cryptographically secure, according to standard measures of cryptographic security for key exchange"
- Simplicity
- Resistance to Memory DoS
- Resistance to CPU DoS
- Privacy
  Identity protection for some parties, against some classes of attacks
- Efficiency
- Non-negotiated
- "Flexible" perfect forward secrecy
  With reuse of exponentials
- Plausible deniability

These goals are (sometimes) contradictory.

Using JFK


a
initiator
responder
b
init(B,b')
listen(A)
accept(a,k)
connect(b,k)
local
IP
local

The JFKr protocol

Msg 1	$i \rightarrow r$:	$n_i, g\hat{\ }d_i$
Msg 2	$r \rightarrow i$:	$n_i, n_r, g\hat{\ }d_r, p_r, h_t$
Msg 3	$i \rightarrow r$:	$n_i, n_r, g\hat{\ }d_i, g\hat{\ }d_r, h_t, e_i, \mathsf{Hmac}\{k_a\}(\text{`i'}, e_i)$
Msg 4	$r \rightarrow i$:	$n_i, n_r, e_r, \mathsf{Hmac}\{k_a\}(\text{`r'}, e_r)$

where

$$h_t = \mathsf{Hmac}\{k_t\}(g\hat{\ }d_r, n_r, n_i, \mathsf{IP}_i)$$

$$e_i = \mathsf{Encrypt}\{k_e\}(\mathsf{id}_i, \mathsf{id}'_r, sa_i, s_i)$$
$$e_r = \mathsf{Encrypt}\{k_e\}(\mathsf{id}_r, sa', s_r)$$

$$s_i = \mathsf{Sign}\{i\}(n_i, n_r, g\hat{\ }d_i, g\hat{\ }d_r, p_r)$$
$$s_r = \mathsf{Sign}\{r\}(g\hat{\ }d_r, n_r, g\hat{\ }d_i, n_i)$$

$$k_u = \mathsf{Hmac}\{g\hat{\ }(d_i\, d_r)\}(n_i, n_r, \text{`u'}) \text{ for } u = a, e, v$$

The JFKr protocol: flexible PFS

Msg 1	$i \rightarrow r$:	$n_i, g\hat{}d_i$
Msg 2	$r \rightarrow i$:	$n_i, n_r, g\hat{}d_r, p_r, h_t$
Msg 3	$i \rightarrow r$:	$n_i, n_r, g\hat{}d_i, g\hat{}d_r, h_t, e_i, \mathsf{Hmac}\{k_a\}(\text{‘i’}, e_i)$
Msg 4	$r \rightarrow i$:	$n_i, n_r, e_r, \mathsf{Hmac}\{k_a\}(\text{‘r’}, e_r)$

where … $(g\hat{}d_r, n_r, n_i, \mathsf{IP}_i)$

The pair of nonces is unique to this session

… $(\mathsf{id}_i, \mathsf{id}'_r, sa_i, s_i)$

$e_r = \mathsf{Encrypt}\{k_e\}(\mathsf{id}_r,$ …

Many keys can be derived from the same exponentials for different usages

$s_i = \mathsf{Sign}\{i\}(n_i, n_r, g\hat{}$ …

$s_r = \mathsf{Sign}\{r\}(g\hat{}d_r, n_r, g$ …

$$k_u = \mathsf{Hmac}\{g\hat{}(d_i\, d_r)\}(n_i, n_r, \text{‘u’}) \text{ for } u = a, e, v$$

The JFKr protocol: DoS

The responder uses an authenticator against DoS

Msg 1	$i \rightarrow r:$	$n_i, g\hat{}d_i$
Msg 2	$r \rightarrow i:$	$n_i, n_r, g\hat{}d_r, p_r, h_t$
Msg 3	$i \rightarrow r:$	$n_i, n_r, g\hat{}d_i, g\hat{}d_r, h_t, e_i, \mathsf{Hmac}\{k_a\}(\text{'i'}, e_i)$
Msg 4	$r \rightarrow i:$	$n_i, n_r, e_r, \mathsf{Hmac}\{k_a\}(\text{'r'}, e_r)$

where $h_t \;=\; \mathsf{Hmac}\{k_t\}(g\hat{}d_r, n_r, n_i, \mathsf{IP}_i)$

$e_i \;=\; \mathsf{Encrypt}\{k$ … $i, \mathsf{id}'_r, sa_i, s_i)$

… $sa', s_r)$

The responder can check that the contents of msg 3 matches the contents of msg 1 & 2

… $i, g\hat{}d_r, p_r)$

… $\hat{}d_i, n_i)$

$k_u \;=\; \mathsf{Hmac}\{g\hat{}(d_i\, d_r)\}(n_i, n_r, \text{'u'})$ for $u = a, e, v$

The JFKr protocol: Privacy

$$
\begin{array}{lll}
\text{Msg 1} & i \rightarrow r: & n_i, g\hat{}d_i \\
\text{Msg 2} & r \rightarrow i: & n_i, n_r, g\hat{}d_r, p_r, h_t \\
\text{Msg 3} & i \rightarrow r: & n_i, n_r, g\hat{}d_i, g\hat{}d_r, h_t, e_i, \mathsf{Hmac}\{k_a\}(\text{'i'}, e_i) \\
\text{Msg 4} & r \rightarrow i: & n_i, n_r, e_r, \mathsf{Hma}
\end{array}
$$

$$
\begin{array}{llll}
\text{where} & h_t & = & \mathsf{Hmac}\{k_t\}(g\hat{}d_r, n_r, n_i, \mathsf{IP} \\
 & e_i & = & \mathsf{Encrypt}\{k_e\}(\mathsf{id}_i, \mathsf{id}'_r, sa_i, s_i) \\
 & e_r & = & \mathsf{Encrypt}\{k_e\}(\mathsf{id}_r, sa', s_r) \\
 & s_i & = & \mathsf{Sign}\{i\}(n_i, n_r, g\hat{}d_i, g\hat{}d_r, p_r) \\
 & s_r & = & \mathsf{Sign}\{r\}(g\hat{}d_r, n_r, g\hat{}d_i, n_i) \\
 & k_u & = & \mathsf{Hmac}\{g\hat{}(d_i\, d_r)\}(
\end{array}
$$

Identities are always encrypted

Identities are never signed

The JFKr protocol

$$\begin{array}{lll}
\text{Msg 1} & i \rightarrow r: & n_i, g\hat{\ }d_i \\
\text{Msg 2} & r \rightarrow i: & n_i, n_r, g\hat{\ }d_r, p_r, h_t \\
\text{Msg 3} & i \rightarrow r: & n_i, n_r, g\hat{\ }d_i, g\hat{\ }d_r, h_t, e_i, \mathsf{Hmac}\{k_a\}(\text{`i'}, e_i) \\
\text{Msg 4} & r \rightarrow i: & n_i, n_r, e_r, \mathsf{Hmac}\{k_a\}(\text{`r'}, e_r)
\end{array}$$

where

$$\begin{array}{rcl}
h_t & = & \mathsf{Hmac}\{k_t\}(g\hat{\ }d_r, n_r, n_i, \mathsf{IP}_i) \\
e_i & = & \mathsf{Encrypt}\{k_e\}(\mathsf{id}_i, \mathsf{id}'_r, sa_i, s_i) \\
e_r & = & \mathsf{Encrypt}\{k_e\}(\mathsf{id}_r, sa', s_r) \\
s_i & = & \mathsf{Sign}\{i\}(n_i, n_r, g\hat{\ }d_i, g\hat{\ }d_r, p_r) \\
s_r & = & \mathsf{Sign}\{r\}(g\hat{\ }d_r, n_r, g\hat{\ }d_i, n_i) \\
k_u & = & \mathsf{Hmac}\{g\hat{\ }(d_i\, d_r)\}(n_i, n_r, \text{`u'}) \text{ for } u = a, e, v
\end{array}$$

Some minor problems

Identity protection?

- Two variants with different trade-offs
  - "JFKi protects id_i against active attacks"
  - "JFKr protects id_r against active attacks and protects id_i against passive attacks"

- What is guaranteed? Does it make sense for the responder? This depends on relations between principals and roles

- Various leaks:
  - An attacker can perform traffic analysis using nonces, IP addresses, and insider knowledge (cf. private authentication)

  - A passive attacker can observe shared exponentials
    - if exponentials are re-used by a single principal, all these sessions involve the same principal
    - an active attacker (or an insider) may obtain the identity for one of these sessions

  ...

Identity protection in JFKr ?

$$
\begin{array}{llll}
\text{Msg 1} & i \rightarrow r: & n_i, g\hat{\ }d_i \\
\text{Msg 2} & r \rightarrow i: & n_i, n_r, g\hat{\ }d_r, p_r, h_t \\
\text{Msg 3} & i \rightarrow r: & n_i, n_r, g\hat{\ }d_i, g\hat{\ }d_r, h_t, e_i, \mathsf{Hmac}\{k_a\}(\text{‘i’}, e_i) \\
\text{Msg 4} & r \rightarrow i: & n_i, n_r, e_r, \mathsf{Hmac}\{k_a\}(\text{‘r’}, e_r)
\end{array}
$$

where $h_t = \mathsf{Hmac}\{k_t\}(g\hat{\ }d_r, n_r, n_i, \mathsf{IP}_i)$

An attacker E can

1. Intercept message 2
2. Initiate its own session with R with the same nonce n_i and its exponential
3. Swap the two messages 2
4. Guess id_r and proceed as usual
5. Observe messages 4

The responder accepts two sessions (I,R) and (E,R) only if E's guess is right

Fix: MAC the initiator exponential too

Non-negotiated?

- Usually, the cryptographic algorithms are negotiated:
  hash, encryption, certificates, compression, …
  Some algorithms are weak (legacy, legal…), or even nil.

- The protocol must (at least) authenticate the negotiation, and also relies on these operations for authentication! Cf. SSL

- "JFK is non-negotiated": the responder demands specific algorithms, the initiator takes it or leaves it. Still…
  - If the responder demands weak algorithms,
    there is no guarantees at all.
  - What if the attacker modifies the responder's demands?
  - Recent fix in JFKi: sign the algorithm demands

Caching message 3?

- “The responder caches answers to identical messages 3”
- More precisely, the responder should answer just once for every valid token received in a message 3.
- Otherwise, several attacks appear

Caching message 3?

Msg 1	$i \to r$:	$n_i, g\hat{\ }d_i$
Msg 2	$r \to i$:	$n_i, n_r, g\hat{\ }d_r, p_r, h_t$
Msg 3	$i \to r$:	$n_i, n_r, g\hat{\ }d_i, g\hat{\ }d_r, h_t, e_i, \mathsf{Hmac}\{k_a\}(\text{‘i’}, e_i)$
Msg 4	$r \to i$:	$n_i, n_r, e_r, \mathsf{Hmac}\{k_a\}(\text{‘r’}, e_r)$

where $h_t = \mathsf{Hmac}\{k_t\}(g\hat{\ }d_r, n_r, n_i, \mathsf{IP}_i)$

$e_i = \mathsf{E}$ …

$e_r =$ …

$s_i =$ …

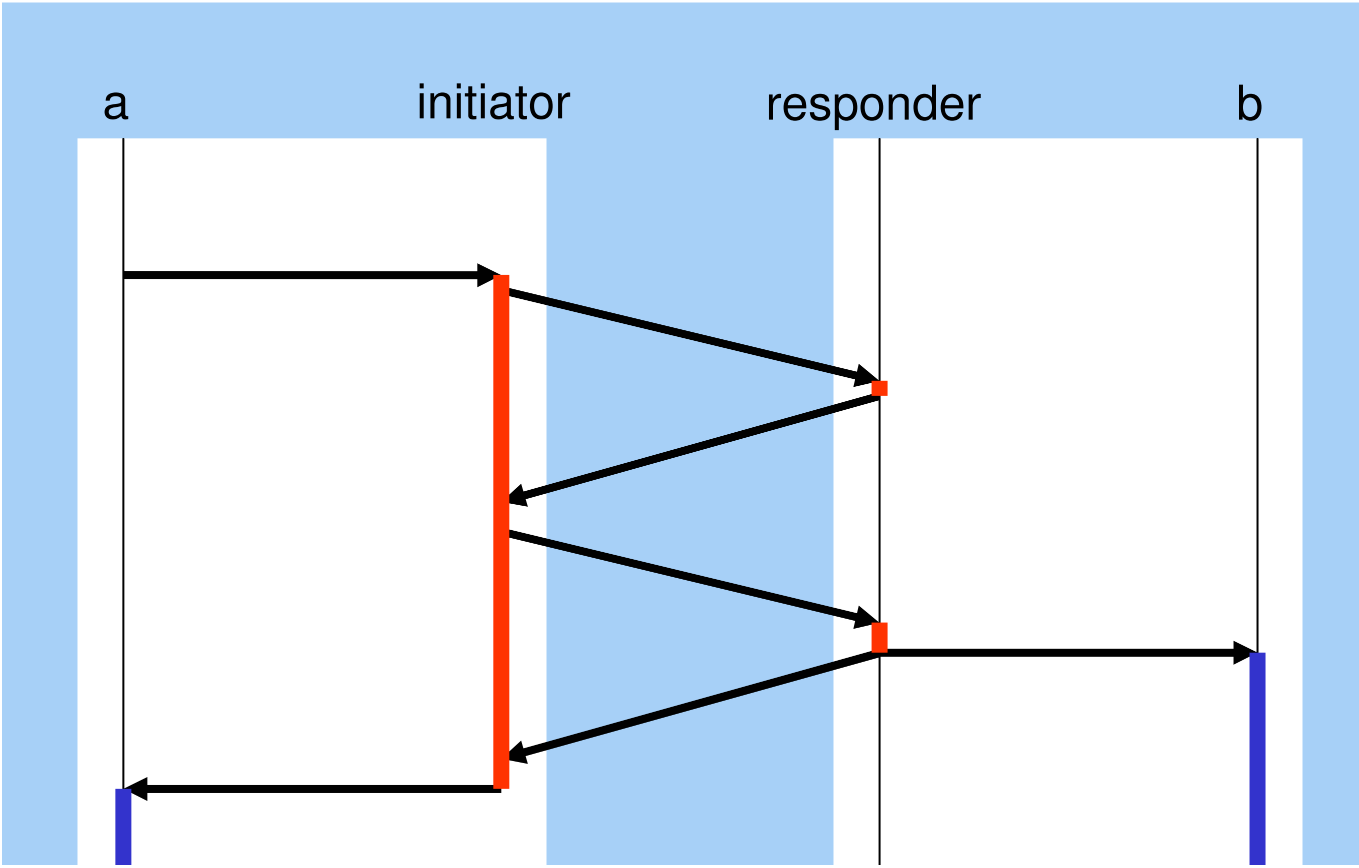
$s_r = \mathsf{Sign}$ …

$k_u = \mathsf{Hmac}\{$ …

An active attacker could trick R into accepting many sessions with the same key

An attacker E could obtain a valid message 3 and modify eg the exponential (an easy, “blind” DoS attack against R)

A model of JFK in applied pi



Public key signature

- To model public-key signature, we construct the public verification key form the private signing key:

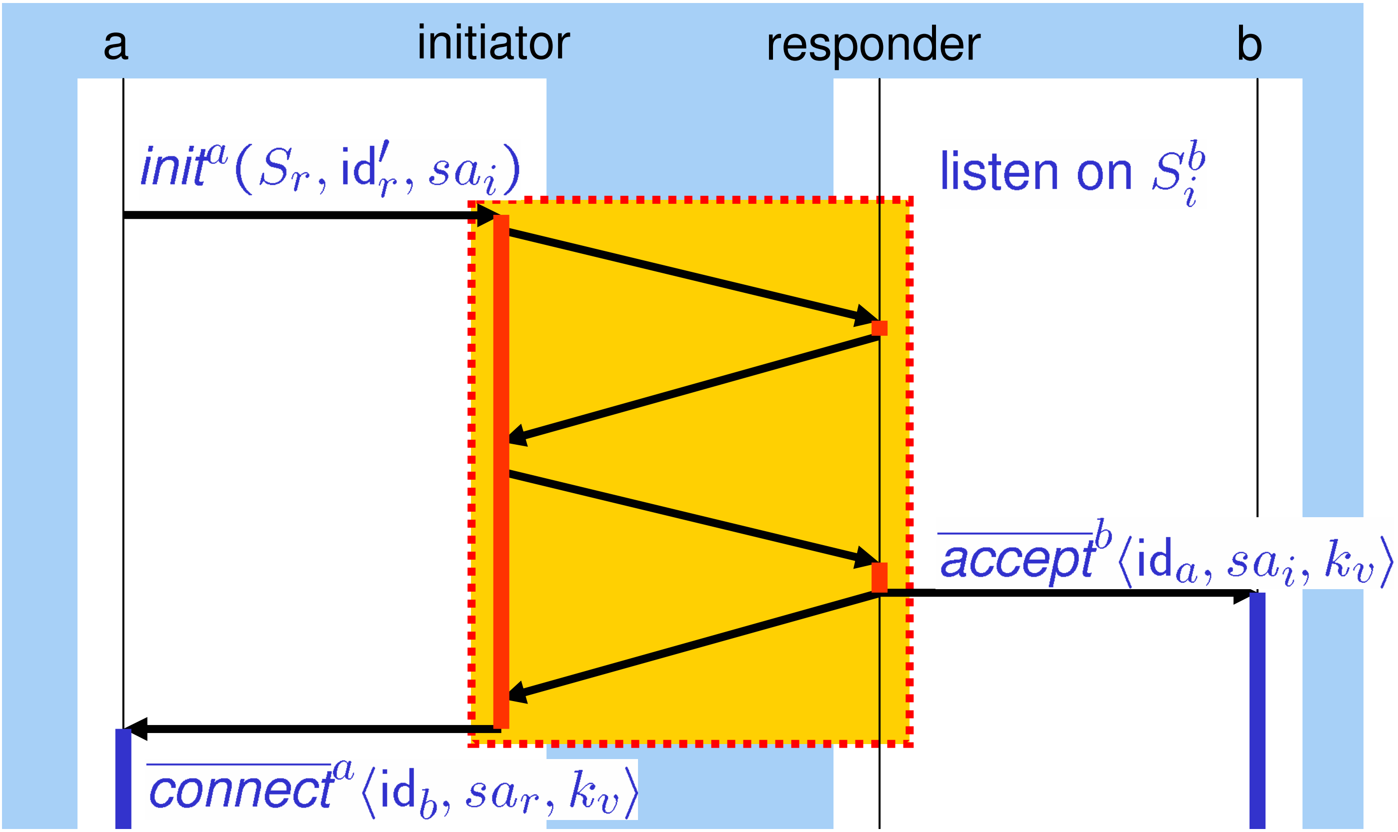
$$\mathsf{Verify}\{\mathsf{Pk}(k), \mathsf{Sign}\{k\}(v)\}(v) \quad = \quad \mathsf{True}$$

- Using active substitutions, we can write a process that exports the public key, and keeps the signing key secret.

$$\nu s.\Big(\{pk = \mathsf{Pk}(s)\} \mid \overline{a}\langle \mathsf{Sign}\{s\}(M)\rangle\Big)$$

Control actions

- We distinguish between
  - principals (signers)
  - JFK roles: initiator, responder (exponentials)
- We provide an API between applications & JFK



Grammar for terms

$M, N ::=$	terms
x, y, z	variable
m, n, s, t	name
$B\hat{}X$	exponential
$\mathsf{Pk}(K)$	public key (and identity)
$\mathsf{S}\{K\}(T)$	public-key signature
$\mathsf{V}\{K, S\}(D)$	public-key signature verification
$\mathsf{H}\{K\}(T)$	keyed crypto hash function
$\mathsf{E}\{K\}(T)$	shared-key encryption
$\mathsf{D}\{K\}(T)$	shared-key decryption
$\mathsf{T}_e, \mathsf{T}_a, \mathsf{T}_v$	constant tags for key derivation
$\mathsf{1}(\_, \_), \mathsf{2}(\_, \_, \_, \_, \_), \ldots$	constructors for JFK messages
$\mathsf{F}_1^1(\_), \ldots, \mathsf{F}_4^2(\_)$	selectors for JFK messages
$K \in S$	sets (for authorized identities)

Equations for terms

$$
\begin{array}{rcll}
(g\hat{\ }y)\hat{\ }z & = & (g\hat{\ }z)\hat{\ }y & \text{Diffie-Hellman exponentials} \\
\mathsf{V}\{\mathsf{Pk}(k), \mathsf{S}\{k\}(v)\}(v) & = & \mathsf{True} & \text{Public key signature verification} \\
\mathsf{D}\{k\}(\mathsf{E}\{k\}(v)) & = & v & \text{Shared-key decryption} \\
\\
\mathsf{F}^i_n(n(v_1, \ldots, v_i, \ldots)) & = & v_i & \text{Selection of message fields} \\
K \in \{\ldots, K, \ldots\} & = & \mathsf{True} & \text{Set membership (authorization)} \\
\\
\mathsf{RecoverKey}(\mathsf{S}\{k\}(v)) & = & \mathsf{Pk}(k) & \text{Public key recovery *} \\
\mathsf{RecoverText}(\mathsf{S}\{k\}(v)) & = & v & \text{Signed text recovery *}
\end{array}
$$

JFK configuration initiator responder

$$
\begin{array}{rcll}
JFK & = & \prod_{a \in \mathcal{L}} PK^a[I^a | R^a] & \text{JFK for principals } a \in \mathcal{L} \\
PK^a[\_] & = & \nu a.\{id^a = \mathsf{Pk}(a)\} \mid [\_] & \text{Signing key } a \text{ and identity } id^a \\
D_z[\_] & = & \nu d_z.\{x_z = g\hat{\ }d_z\} \mid [\_] & \text{DH secret and exponential} \\
C_z & = & \nu h.\{h = x_{\bar{z}}\hat{\ }d_z\} \mid \prod_{u=a,e,v} K_u & \text{DH computation} \\
K_u & = & \{k_u = \mathsf{H}\{h\}(n_i, n_r, \mathsf{T}_u)\} & \text{derivation of key } k_u
\end{array}
$$

Only a specific subset of principals appear in $\mathcal{L}$

These are ”compliant principals”.

JFK configuration initiator responder

$I = \prod_{\widetilde{x}_i} D_i \left[!\mathit{init}^a(S_r, id'_r, sa_i).I_1 \right]$	init handler (for a)
$I_1 = \nu n_i.\langle 1(n_i, x_i)\rangle \mid$	*send message 1*
$(2(n'_i, n_r, x_r, p_r, h_t)).I_3$	*wait for message 2*
$I_3 = \nu k_e\, k_a\, k_v.C_i \mid$	*compute the keys*
$\nu s_i\, e_i\, h_i.$	*build message 3*
$\{s_i = \mathsf{S}\{a\}(n_i, n_r, x_i, x_r, p_r)\} \mid$	
$\{e_i = \mathsf{E}\{p_r, k_e\}(\mathit{id}_a, \mathit{id}'_r, sa_i, s_i)\} \mid$	
$\{h_i = \mathsf{H}\{k_a\}(\mathsf{T}_i, e_i)\} \mid$	
$\langle 3(n_i, n_r, x_i, x_r, h_t, e_i, h_i)\rangle$	*send message 3*
$(4(e_r, h_r)).I_5$	*wait for message 4*
$I_5 = \mathit{if}\ \mathsf{H}\{k_a\}(\mathsf{T}_r, e_r) = h_r\ \mathit{then}$	*check MAC*
$\nu \mathit{id}_r\, sa_r\, s_r.\{\mathit{id}_r, sa_r, s_r = \mathsf{D}\{p_r, k_e\}(e_r)\} \mid$	*decrypt*
$\mathit{if}\ id_r \in S_r\ \mathit{then}$	*authorize*
$\mathit{if}\ \mathsf{V}\{s_r, id_r\}(n_i, n_r, x_i, x_r)\ \mathit{then}$	*authenticate*
$\overline{\mathit{connect}}^a\langle \mathit{id}_r, sa_r, k_v\rangle$	*open session*

JFK configuration initiator responder

$$R = \prod_{\widetilde{x_r}} \nu c_t\, k_t . D_r \left[\begin{array}{l} !(1(n_i, x_i)).R_2 \mid \\ !(3(n_i, n_r, x_i, x_r, h_t, e_i, h_i)).R_4 \end{array} \right]$$

responder (for a)

$$R_2 = \nu n_r\, h_t . \{h_t = \mathsf{H}\{k_t\}(x_r, n_r, n_i)\} \mid \overline{c_t}\langle h_t \rangle \mid \langle 2(n_i, n_r, x_r, p_r, h_t)\rangle$$

token
send message 2

$$\begin{array}{rl} R_4 = & \mathit{if}\ \mathsf{H}\{k_t\}(x_r, n_r, n_i) = h_t\ \mathit{then} \\ & c_t(h'_t).\mathit{if}\ h'_t = h_t\ \mathit{then} \\ & \nu k_e\, k_a\, k_v . C_r \mid \\ & \mathit{if}\ \mathsf{H}\{k_a\}(\mathsf{T}_i, e_i) = h_i\ \mathit{then} \\ & \nu id_i\, sa_i\, s_i . \{id_i, \_, sa_i, s_i = \mathsf{D}\{k_e\}(e_i)\} \mid \\ & \mathit{if}\ id_i \in S_i^a\ \mathit{then} \\ & \mathit{if}\ \mathsf{V}\{s_i, id_i\}(n_i, n_r, x_i, x_r, p_r)\ \mathit{then} \\ & \overline{\mathit{accept}}^a\langle \mathit{id}_i, sa_i, k_v\rangle . \\ & \nu s_r\, e_r\, h_r . \\ & \{s_r = \mathsf{S}\{a\}(n_i, n_r, x_i, x_r)\} \mid \\ & \{e_r = \mathsf{E}\{k_e\}(\mathit{id}_a, sa_r, s_r)\} \mid \\ & \{h_r = \mathsf{H}\{k_a\}(\mathsf{T}_r, e_r)\} \mid \\ & \langle 4(e_r, h_r)\rangle \end{array}$$

verify token
accept token once
handle message 3

authorize
authenticate
complete keying

build message 4

send message 4

Security properties ?

- Main results:
  - In any state, the protocol can establish a secure session between compliant principals
  - There are causality relations between control actions (aka authentication)
  - When both protocols are compliant,
    the key is secure (aka perfect forward secrecy)

- Stated independently of low-level messages
- Compliant principals are also part of the "attacker"

- Additional results:
  - Some identity protection
  - Some DOS properties
  - Some plausible deniability

Operational correctness

Basic Operational Correctness

…ocol configuration with compli-

… $b \in$ … s

with $id_b \in S_r$. *We … ave:*

$$S \xrightarrow{init^a(S_r, id'_r, sa_i)} \rightarrow^8 \approx \xrightarrow{\nu k_v.\overline{accept}^b\langle id_a, sa_i, k_v\rangle} \xrightarrow{\overline{connect}^a\langle id_b, sa_r, k_v\rangle} S$$

The protocol uses internal steps:
- low-level communications
- tests after receiving messages

At the end of the protocol, we can use an observational equivalence to simplify the established keys.

We start from any reachable configuration of the p… (past & running sess…

Each party gets the other's identity & parameters, … shared key.

We end up exactly in the original configuration !
In particular, *kv* is a perfect key.

Operational correctness with eavesdropping

Let S be a running protocol configuration with compliant principals $\mathcal{L}$. Let $a, b \in \mathcal{L}$ and S_r be a set of terms with $\mathit{id}_b \in S_r$. We have:

In addition, the environment can observe mostly-opaque messages, still unrelated to the session key.

$$
\begin{aligned}
S \; & \xrightarrow{\mathit{init}^a(S_r, \mathit{id}'_r, sa_i)} \\
& \xrightarrow{\nu n_i.[1(n_i, x_i^a)]} \\
& \xrightarrow{\nu n_r\, h_t.[2(n_i, n_r, x_r^b, p_r^b, h_t)]} \\
& \xrightarrow{\nu e_i\, h_i.[3(n_i, n_r, x_i^a, x_r^b, h_t, e_i, h_i)]} \\
& \xrightarrow{\nu e_r\, h_r.[4(e_r, h_r)]} \\
& \rightarrow^4 \approx \xrightarrow{\nu k_v.\overline{\mathit{accept}}^b\langle \mathit{id}_a, sa_i, k_v\rangle} \xrightarrow{\overline{\mathit{connect}}^a\langle \mathit{id}_b, sa_r, k_v\rangle} S
\end{aligned}
$$

where x_i^a is an exponential defined by I^a, x_r^b and p_r^b are an exponential and the preferences defined by R^b, and n_i, n_r, h_t, e_i, h_i, e_r, h_r, k_v are all fresh names.

Correspondence properties

Let S_0 be an initial configuration with compliant principals $\mathcal{L}$ and labeled transitions $S_0 \xrightarrow{\mu}{}^{*} S$ with no immediate output on any channel accept^b in S.

The actions occuring in μ have the following properties:

1. *For any $\beta = \overline{\mathit{accept}}^b\langle \mathit{id}_a, sa_i, k_v\rangle$, we have $\mathit{id}_a \in S_i^b$.*
2. *For any β with $a \in \mathcal{L}$, there is a distinct $\alpha = \mathit{init}^a(S_r, \mathit{id}_r, sa_i)$ with $\mathit{id}_b \in S_r$.*
3. *For any $\gamma = \overline{\mathit{connect}}^a\langle \mathit{id}_b, sa_r, k_v\rangle$ there is a distinct $\alpha = \mathit{init}^a(S_r, \mathit{id}_r, sa_i)$ with $\mathit{id}_b \in S_r$.*
4. *For any γ with $b \in \mathcal{L}$, there is a distinct $\beta = \overline{\mathit{accept}}^b\langle \mathit{id}_a, sa_i, k_v\rangle$.*
5. *For any two other control actions that output a session key (either $\overline{\mathit{connect}}^a\langle \_, \_, k_v\rangle$ or $\overline{\mathit{accept}}^b\langle \_, \_, k_v\rangle$), the keys are equationally different.*

Anti-DoS properties

- We characterize "round-trip communication" as a trace property:

  $\nu n_r\, h_t.(2(\_, n_r, \_, \_, h_t)) \ldots \langle 3(\_, n_r, \_, \_, h_t, \_, \_)\rangle$

  and show an injective correspondence property from (informally) expensive responder steps to round-trips.

- The use of a token is a refinement, modelled as an equivalence
  - The basic model uses local responder state after message 1 & 2
  - The refined model uses the token instead

  This is much like the parallel law for CCS

  $(!P \mid !Q) \approx !(P \mid !Q)$

Plausible deniability

- What gets signed ?
  - Authentication for an active party
  - Deniability from some (data) evidence
- Example:
  - a opens a session with e (which may not comply with JFK)
  - e tries to prove that a opened the session from his data.
  - a refutes e's evidence by exhibiting a trace where
    - a complies with JFK
    - a never tries to open a session with e
    - e produces the same evidence

  for instance, a plausible trace may be
    - a opens a session with a compliant b $\neq$ e
    - e is an active attacker that impersonates b

Summary on JFK

- JFK is a state-of-the-art protocol, well-written but message-centric and sometimes imprecise
  - We had to interpret the spec and invent a service API
  - Writing down a precise definition for the intended properties of the protocol is difficult (and reveals problems)
- We wrote a “formal implementation” of JFKr in applied pi
- We obtained a formal counterpart for each informal claim, against a large class of active attackers (=contexts)

Questions?

See also http://research.microsoft.com/~fournet/