The spi-calculus notion of secrecy is stronger than our usual notions of secrecy.

The spi-calculus notion of secrecy is stronger than our usual notions of secrecy.

- The secret x should not be leaked ...

$\longrightarrow$ This process is insecure: $\quad \mathsf{send}_c\langle x\rangle; \mathsf{halt}$

The spi-calculus notion of secrecy is stronger than our usual notions of secrecy.

- The secret x should not be leaked ...

  $\longrightarrow$ This process is insecure: $\mathsf{send}_c\langle x\rangle; \mathsf{halt}$

- ... and even any partial information about x should not be leaked.

  $\longrightarrow$ This process is insecure: $\mathsf{send}_c\langle \{0\}_x\rangle; \mathsf{halt}$

$$P(x) \triangleq \mathsf{send}_c\langle\{0\}_x\rangle; \mathsf{halt}$$

$$P(x) \triangleq \mathsf{send}_c\langle\{0\}_x\rangle; \mathsf{halt}$$

- Assuming perfect cryptography, one cannot compute x from $\{0\}_x$.

$$P(x) \triangleq \mathsf{send}_c\langle\{0\}_x\rangle; \mathsf{halt}$$

- Assuming perfect cryptography, one cannot compute x from $\{0\}_x$.

- But one can get partial information about x.

  For example, one can find out whether x is 0 or not, by using the property:

$$x = 0 \quad \text{iff} \quad \{0\}_x = \{0\}_0$$

$$P(x) \triangleq \mathsf{send}_c\langle\{0\}_x\rangle; \mathsf{halt}$$

- Assuming perfect cryptography, one cannot compute x from $\{0\}_x$.

- But one can get partial information about x.

  For example, one can find out whether x is 0 or not, by using the property:

$$x = 0 \quad \text{iff} \quad \{0\}_x = \{0\}_0$$

$\longrightarrow$ $P(x)$ does not preserve the secrecy of x.

$$P(x) \triangleq \mathsf{send}_c\langle\{0\}_x\rangle; \mathsf{halt}$$

In spi-calculus terminology, we consider the test $(Q, \overline{d})$, where

$$Q \triangleq \mathsf{recv}_c(y); Q_1(y) \qquad Q_1(y) \triangleq \mathsf{check}\ (y == \{0\}_0); Q_2 \qquad Q_2 \triangleq \mathsf{send}_d\langle 0\rangle; \mathsf{halt}$$

$$P(x) \triangleq \mathsf{send}_c\langle\{0\}_x\rangle; \mathsf{halt}$$

In spi-calculus terminology, we consider the test $(Q, \overline{d})$, where

$$Q \triangleq \mathsf{recv}_c(y); Q_1(y) \qquad Q_1(y) \triangleq \mathsf{check}\ (y == \{0\}_0); Q_2 \qquad Q_2 \triangleq \mathsf{send}_d\langle 0\rangle; \mathsf{halt}$$

We show that $P(0) \not\simeq P(\mathsf{succ}\ (0))$.

$$P(x) \triangleq \mathsf{send}_c\langle\{0\}_x\rangle; \mathsf{halt}$$

In spi-calculus terminology, we consider the test $(Q, \overline{d})$, where

$Q \triangleq \mathsf{recv}_c(y); Q_1(y) \qquad Q_1(y) \triangleq \mathsf{check}\ (y == \{0\}_0); Q_2 \qquad Q_2 \triangleq \mathsf{send}_d\langle 0\rangle; \mathsf{halt}$

We show that $P(0) \not\simeq P(\mathsf{succ}\ (0))$.

$P(0) \xrightarrow{\overline{c}} \langle\{0\}_0\rangle\mathsf{halt}$ and $Q \xrightarrow{c} (y)Q_1(y)$

$$P(x) \triangleq \mathsf{send}_c\langle\{0\}_x\rangle; \mathsf{halt}$$

In spi-calculus terminology, we consider the test $(Q, \overline{d})$, where

$$Q \triangleq \mathsf{recv}_c(y); Q_1(y) \qquad Q_1(y) \triangleq \mathsf{check}\ (y == \{0\}_0); Q_2 \qquad Q_2 \triangleq \mathsf{send}_d\langle 0\rangle; \mathsf{halt}$$

We show that $P(0) \not\approx P(\mathsf{succ}\ (0))$.

$$P(0) \xrightarrow{\overline{c}} \langle\{0\}_0\rangle\mathsf{halt} \qquad \text{and} \qquad Q \xrightarrow{c} (y)Q_1(y)$$

$$P(0) \mid Q \quad \xrightarrow{\tau} \quad \overline{c}\langle\{0\}_0\rangle\mathsf{halt} \quad @ \quad (y)Q_1(y)$$

$$P(x) \triangleq \mathsf{send}_c\langle\{0\}_x\rangle; \mathsf{halt}$$

In spi-calculus terminology, we consider the test $(Q, \overline{d})$, where

$$Q \triangleq \mathsf{recv}_c(y); Q_1(y) \qquad Q_1(y) \triangleq \mathsf{check}\ (y == \{0\}_0); Q_2 \qquad Q_2 \triangleq \mathsf{send}_d\langle 0\rangle; \mathsf{halt}$$

We show that $P(0) \not\approx P(\mathsf{succ}\ (0))$.

$$P(0) \xrightarrow{\overline{c}} \langle\{0\}_0\rangle\mathsf{halt} \qquad \text{and} \qquad Q \xrightarrow{c} (y)Q_1(y)$$

$$\begin{array}{rcl} P(0) \mid Q & \xrightarrow{\tau} & \overline{c}\langle\{0\}_0\rangle\mathsf{halt} \quad @ \quad (y)Q_1(y) \\ & = & \mathsf{halt} \mid Q_1(\{0\}_0) \end{array}$$

$$P(x) \triangleq \mathsf{send}_c\langle\{0\}_x\rangle; \mathsf{halt}$$

$$Q \triangleq \mathsf{recv}_c(y); Q_1(y) \qquad Q_1(y) \triangleq \mathsf{check}\ (y == \{0\}_0); Q_2 \qquad Q_2 \triangleq \mathsf{send}_d\langle 0\rangle; \mathsf{halt}$$

$$Q_1(\{0\}_0) \quad > \quad Q_2 \quad \xrightarrow{\overline{d}} \quad \langle 0\rangle\mathsf{halt}$$

Hence $Q_1(\{0\}_0) \quad \xrightarrow{\overline{d}} \quad \langle 0\rangle\mathsf{halt}$

Hence
$$\begin{aligned} \mathsf{halt} \mid Q_1(\{0\}_0) &\xrightarrow{\overline{d}} \mathsf{halt} \mid \langle 0\rangle\mathsf{halt} \\ &= \langle 0\rangle(\mathsf{halt} \mid \mathsf{halt}) \end{aligned}$$

Hence $P(0)$ passes the test $(Q, \overline{d})$.

And we can check that $P(\mathsf{succ}\ (0))$ does not pass the test $(Q, \overline{d})$.

Similarly the following challenge response step does not preserve secrecy of K_{ab} in the spi-calculus model, although the key K_{ab} cannot be computed by an attacker.

$$A \longrightarrow B : N_a$$

$$B \longrightarrow A : \{N_a\}_{K_{ab}}$$

One session of the protocol can be represented by the process

$$\mathsf{new}\ K; (\mathsf{new}\ N; \mathsf{send}_c\langle N\rangle; \mathsf{halt} \mid \mathsf{recv}_c(x); \mathsf{send}_c\langle\{x\}_K\rangle; \mathsf{halt})$$

Intuitively, an attacker can send send a desired message in place of N_a and then get partial information about the secret key as in the previous example.

Another example: $\quad P_1(x) \triangleq \mathsf{new}\ K; \mathsf{send}_c\langle\{x\}_K\rangle; \mathsf{halt}.$

Another example: $P_1(x) \triangleq \mathsf{new}\ K; \mathsf{send}_c\langle\{x\}_K\rangle; \mathsf{halt}.$

The protocol is secure.

Another example: $P_1(x) \triangleq \mathsf{new}\ K; \mathsf{send}_c\langle\{x\}_K\rangle; \mathsf{halt}.$

The protocol is secure. How to prove it?

Another example: $P_1(x) \triangleq \mathsf{new}\ K; \mathsf{send}_c\langle\{x\}_K\rangle; \mathsf{halt}.$

The protocol is secure. How to prove it?

Consider arbitrary terms M_1 and M_2. We show that:
if $P_1(M_1)$ passes some test (Q, β) then $P_1(M_2)$ also passes the test (Q, β)

Another example: $P_1(x) \triangleq \mathsf{new}\ K; \mathsf{send}_c\langle\{x\}_K\rangle; \mathsf{halt}.$

The protocol is secure. How to prove it?

Consider arbitrary terms M_1 and M_2. We show that:
if $P_1(M_1)$ passes some test (Q, β) then $P_1(M_2)$ also passes the test (Q, β)

For this we show that for all $n \geq 0$ and for all R, if $P_1(M_1) \mid R$ can make a sequence of actions $\beta_1, \ldots, \beta_n$ then $P_1(M_2) \mid R$ can also do so.

Note: we must have $\beta_i = \tau$ for $i \leq n-1$.

Another example: $P_1(x) \triangleq \mathsf{new}\ K; \mathsf{send}_c\langle\{x\}_K\rangle; \mathsf{halt}.$

The protocol is secure. How to prove it?

Consider arbitrary terms M_1 and M_2. We show that:
if $P_1(M_1)$ passes some test (Q, β) then $P_1(M_2)$ also passes the test (Q, β)

For this we show that for all $n \geq 0$ and for all R, if $P_1(M_1) \mid R$ can make a sequence of actions $\beta_1, \ldots, \beta_n$ then $P_1(M_2) \mid R$ can also do so.

Note: we must have $\beta_i = \tau$ for $i \leq n-1$.

By induction on n.

Another example: $P_1(x) \triangleq \mathsf{new}\ K; \mathsf{send}_c\langle\{x\}_K\rangle; \mathsf{halt}.$

The protocol is secure. How to prove it?

Consider arbitrary terms M_1 and M_2. We show that:
if $P_1(M_1)$ passes some test (Q, β) then $P_1(M_2)$ also passes the test (Q, β)

For this we show that for all $n \geq 0$ and for all R, if $P_1(M_1) \mid R$ can make a sequence of actions $\beta_1, \ldots, \beta_n$ then $P_1(M_2) \mid R$ can also do so.

Note: we must have $\beta_i = \tau$ for $i \leq n-1$.

By induction on n. For $n = 0$ there is nothing is prove.

$$P_1(x) \triangleq \mathsf{new}\ K; \mathsf{send}_c\langle\{x\}_K\rangle; \mathsf{halt}$$

Case 1: the left component makes an action.

$$P_1(M_1) \mid R \xrightarrow{\overline{c}} (\mathsf{new}\ K)\langle\{M_1\}_K\rangle(\mathsf{halt} \mid R)$$

$$P_1(x) \triangleq \mathsf{new}\ K; \mathsf{send}_c\langle\{x\}_K\rangle; \mathsf{halt}$$

Case 1: the left component makes an action.

$P_1(M_1) \mid R \xrightarrow{\bar{c}} (\mathsf{new}\ K)\langle\{M_1\}_K\rangle(\mathsf{halt} \mid R)$

Then we also have

$P_1(M_2) \mid R \xrightarrow{\bar{c}} (\mathsf{new}\ K)\langle\{M_2\}_K\rangle(\mathsf{halt} \mid R)$

No further transitions are possible in either case. Hence we are done.

Case 2: the right component R is a process and makes an action.

$R \xrightarrow{\beta_1} B$ so that $P_1(M_1) \mid R \xrightarrow{\beta_1} P_1(M_1) \mid B$

and $P_1(M_1) \mid B$ makes a sequence of actions $\beta_2, \ldots, \beta_n$.

Case 2: the right component R is a process and makes an action.

$R \xrightarrow{\beta_1} B$ so that $P_1(M_1) \mid R \xrightarrow{\beta_1} P_1(M_1) \mid B$

and $P_1(M_1) \mid B$ makes a sequence of actions $\beta_2, \ldots, \beta_n$.

Then we also have $P_1(M_2) \mid R \xrightarrow{\beta_1} P_1(M_2) \mid B$

and by induction hypothesis, $P_1(M_2) \mid B$ makes a sequence of actions $\beta_2, \ldots, \beta_n$.

$$P_1(x) \triangleq \mathsf{new}\ K; \mathsf{send}_c\langle\{x\}_K\rangle; \mathsf{halt}$$

Case 3: the two components communicate over channel c.

$$\begin{array}{lrcl} & P_1(M_1) & \xrightarrow{\bar{c}} & (\mathsf{new}\ K)\langle\{M_1\}_K\rangle\mathsf{halt} \\ \text{and} & R & \xrightarrow{c} & (y)R' \\ \text{so that} & P_1(M_1) \mid R & \xrightarrow{\tau} & \mathsf{new}\ K; (\mathsf{halt} \mid R'(\{M_1\}_K)) \end{array}$$

and $\mathsf{new}\ K; (\mathsf{halt} \mid R'(\{M_1\}_K))$ makes the sequence of actions $\beta_2, \ldots, \beta_n$.

$$P_1(x) \triangleq \mathsf{new}\ K; \mathsf{send}_c\langle\{x\}_K\rangle; \mathsf{halt}$$

Case 3: the two components communicate over channel c.

$$\begin{array}{lrcl} & P_1(M_1) & \xrightarrow{\bar{c}} & (\mathsf{new}\ K)\langle\{M_1\}_K\rangle\mathsf{halt} \\ \text{and} & R & \xrightarrow{c} & (y)R' \\ \text{so that} & P_1(M_1) \mid R & \xrightarrow{\tau} & \mathsf{new}\ K; (\mathsf{halt} \mid R'(\{M_1\}_K)) \end{array}$$

and $\mathsf{new}\ K; (\mathsf{halt} \mid R'(\{M_1\}_K))$ makes the sequence of actions $\beta_2, \ldots, \beta_n$.

Then we also have

$$\begin{array}{rcl} P_1(M_2) & \xrightarrow{\bar{c}} & (\mathsf{new}\ K)\langle\{M_1\}_K\rangle\mathsf{halt} \\ P_1(M_2) \mid R & \xrightarrow{\tau} & \mathsf{new}\ K; (\mathsf{halt} \mid R'(\{M_2\}_K)) \end{array}$$

$$P_1(x) \triangleq \mathsf{new}\ K; \mathsf{send}_c\langle\{x\}_K\rangle; \mathsf{halt}$$

Case 3: the two components communicate over channel c.

$$\begin{array}{lrcl} & P_1(M_1) & \xrightarrow{\overline{c}} & (\mathsf{new}\ K)\langle\{M_1\}_K\rangle\mathsf{halt} \\ \text{and} & R & \xrightarrow{c} & (y)R' \\ \text{so that} & P_1(M_1) \mid R & \xrightarrow{\tau} & \mathsf{new}\ K; (\mathsf{halt} \mid R'(\{M_1\}_K)) \end{array}$$

and $\mathsf{new}\ K; (\mathsf{halt} \mid R'(\{M_1\}_K))$ makes the sequence of actions $\beta_2, \ldots, \beta_n$.

Then we also have

$$\begin{array}{rcl} P_1(M_2) & \xrightarrow{\overline{c}} & (\mathsf{new}\ K)\langle\{M_1\}_K\rangle\mathsf{halt} \\ P_1(M_2) \mid R & \xrightarrow{\tau} & \mathsf{new}\ K; (\mathsf{halt} \mid R'(\{M_2\}_K)) \end{array}$$

It remains to show that ...

Claim: For all $S(x)$, if $K \notin fn(S)$, and if $S(\{M_1\}_K) \xrightarrow{\beta} A$ then

- $A = B(\{M_1\}_K)$ with $K \notin fn(S)$
- $S(\{M_2\}_K) \xrightarrow{\beta} B(\{M_1\}_K)$

Claim: For all $S(x)$, if $K \notin fn(S)$, and if $S(\{M_1\}_K) \xrightarrow{\beta} A$ then

- $A = B(\{M_1\}_K)$ with $K \notin fn(S)$
- $S(\{M_2\}_K) \xrightarrow{\beta} B(\{M_1\}_K)$

Proof: by structural induction on S.

Case 1: $S = \mathsf{send}_c\langle N_1, \ldots, N_k\rangle; S_1$

We have $S(\{M_1\}_K) \xrightarrow{\overline{c}} (\langle N_1, \ldots, N_k\rangle S_1)(\{M_1\}_K)$

and also $S(\{M_2\}_K) \xrightarrow{\overline{c}} (\langle N_1, \ldots, N_k\rangle S_1)(\{M_2\}_K)$

Case 1: $S = \mathsf{send}_c\langle N_1, \ldots, N_k\rangle; S_1$

We have $S(\{M_1\}_K) \xrightarrow{\overline{c}} (\langle N_1, \ldots, N_k\rangle S_1)(\{M_1\}_K)$

and also $S(\{M_2\}_K) \xrightarrow{\overline{c}} (\langle N_1, \ldots, N_k\rangle S_1)(\{M_2\}_K)$

Case 2: $S = \mathsf{recv}_c(x_1, \ldots, x_k); S_1$

We have $S(\{M_1\}_K) \xrightarrow{c} (x_1, \ldots, x_k)S_1(\{M_1\}_K)$

and also We have $S(\{M_2\}_K) \xrightarrow{c} (x_1, \ldots, x_k)S_1(\{M_2\}_K)$

Case 1: $S = \mathsf{send}_c\langle N_1, \ldots, N_k\rangle; S_1$

We have $S(\{M_1\}_K) \xrightarrow{\overline{c}} (\langle N_1, \ldots, N_k\rangle S_1)(\{M_1\}_K)$

and also $S(\{M_2\}_K) \xrightarrow{\overline{c}} (\langle N_1, \ldots, N_k\rangle S_1)(\{M_2\}_K)$

Case 2: $S = \mathsf{recv}_c(x_1, \ldots, x_k); S_1$

We have $S(\{M_1\}_K) \xrightarrow{c} (x_1, \ldots, x_k)S_1(\{M_1\}_K)$

and also We have $S(\{M_2\}_K) \xrightarrow{c} (x_1, \ldots, x_k)S_1(\{M_2\}_K)$

Case 3: $S = \mathsf{halt}$. Trivial, since no actions are possible.

Case 4: $S = S_1 \mid S_2$,

$$\begin{array}{lll} \text{and} & S_1(\{M_1\}_K & \xrightarrow{\beta} & A_1 \\ \text{so that} & S(\{M_1\}_K) & \xrightarrow{\beta} & A_1 \mid S_2(\{M_1\}_K) \end{array}$$

By induction hypothesis, $A_1 = B_1(\{M_1\}_K)$, $K \notin fn(B_1)$ and $S_1(\{M_2\}_K) \xrightarrow{\beta} B_1(\{M_2\}_K)$.

Then we have $S(\{M_2\}_K) \xrightarrow{\beta} B_1(\{M_2\}_K/x) \mid S_2(\{M_2\}_K)$.

Case 4: $S = S_1 \mid S_2$,

$$\begin{array}{lccc} \text{and} & S_1(\{M_1\}_K & \xrightarrow{\beta} & A_1 \\ \text{so that} & S(\{M_1\}_K) & \xrightarrow{\beta} & A_1 \mid S_2(\{M_1\}_K) \end{array}$$

By induction hypothesis, $A_1 = B_1(\{M_1\}_K)$, $K \notin fn(B_1)$ and $S_1(\{M_2\}_K) \xrightarrow{\beta} B_1(\{M_2\}_K)$.

Then we have $S(\{M_2\}_K) \xrightarrow{\beta} B_1(\{M_2\}_K/x) \mid S_2(\{M_2\}_K)$.

We argue similarly if the right component S_2 makes an action.

Case 5: $S = S_1 \mid S_2$,

$$S_1(\{M_1\}_K) \xrightarrow{\bar{c}} A_1 \quad \text{and} \quad S_2(\{M_1\}_K) \xrightarrow{c} A_2$$

so that $$S(\{M_1\}_K) \xrightarrow{\tau} A_1 \mathbin{@} A_2$$

By induction hypothesis,

$A_1 = B_1(\{M_1\}_K)$, $A_2 = B_2(\{M_1\}_K)$, $K \notin fn(B_1) \cup fn(B_2)$,

$S_1(\{M_2\}_K) \xrightarrow{\bar{c}} B_1(\{M_2\}_K)$ and $S_2(\{M_2\}_K) \xrightarrow{c} B_2(\{M_2\}_K)$.

Hence $S(\{M_2\}_K) \xrightarrow{\tau} B_1(\{M_2\}_K) \mathbin{@} B_2(\{M_2\}_K)$

Case 6: $S = \mathsf{repeat}\ S_1$ and

- either $S_1(\{M_1\}_K) \xrightarrow{\beta} A$ so that $S(\{M_1\}_K) \xrightarrow{\beta} A \mid S(\{M_1\}_K)$.
- or $S_1(\{M_1\}_K) \xrightarrow{\bar{c}} A_1$ and $S_2(\{M_1\}_K) \xrightarrow{c} A_2$ so that $S(\{M_1\}_K) \xrightarrow{\tau} (A_1 \mathbin{@} A_2) \mid S(\{M_1\}_K)$

The cases are similar to Case 4 and Case 5.

Case 6: $S = \mathsf{repeat}\ S_1$ and

- either $S_1(\{M_1\}_K) \xrightarrow{\beta} A$ so that $S(\{M_1\}_K) \xrightarrow{\beta} A \mid S(\{M_1\}_K)$.
- or $S_1(\{M_1\}_K) \xrightarrow{\overline{c}} A_1$ and $S_2(\{M_1\}_K) \xrightarrow{c} A_2$ so that $S(\{M_1\}_K) \xrightarrow{\tau} (A_1 \ @\ A_2) \mid S(\{M_1\}_K)$

The cases are similar to Case 4 and Case 5.

Case 7: $S = \mathsf{new}\ n; S_1$. Again a straightforward application of induction hypothesis.

Case 8: $S = \mathsf{check}\ (M == N); S_1$,

$M[\{M_1\}_K/x] = N[\{M_1\}_K/x]$ and $S_1(\{M_1\}_K) \xrightarrow{\beta} A$ so that $S(\{M_1\}_K) \xrightarrow{\beta} A$

Since $K \notin fn(M) \cup fn(N)$, we have $M = N$. (Proof: exercise)

Hence $M[\{M_2\}_K/x] = N[\{M_2\}_K/x]$.

Also by induction hypothesis, $A = B(\{M_1\}_K)$ and

$S_1(\{M_2\}_K) \xrightarrow{\beta} B(\{M_2\}_K)$

so that $S(\{M_2\}_K) \xrightarrow{\beta} B(\{M_2\}_K)$.

Case 9: $S = \mathsf{let}\ (x, y) = M; S_1$

Case 10: $S = \mathsf{case}\ M\ \mathsf{of}\ 0 : S_1,\ \mathsf{succ}\ (y) : S_2$

These are similar to (and simpler than) Case 11.

Case 11: $S = \mathsf{case}\ M\ \mathsf{of}\ \{x_1, \ldots, x_k\}_N : S_1$.

$K \neq N$ because $K \notin fn(S)$.

Hence if M is the variable x then no action is possible.

For an action to be possible we must have $M = \{N_1, \ldots, N_k\}_N$.

Let $S_1[N_1/x_1, \ldots, N_k/x_k][\{M_1\}_K/x] \xrightarrow{\beta} A$

so that $S[\{M_1\}_K/x] \xrightarrow{\beta} A$.

By induction hypothesis, $A = B[\{M_1\}_K/x]$ and

$S_1[N_1/x_1, \ldots, N_k/x_k][\{M_2\}_K/x] \xrightarrow{\beta} B[\{M_2\}_K/x]$

so that $S[\{M_2\}_K/x] \xrightarrow{\beta} B[\{M_2\}_K/x]$.

After all this, we conclude:

the process $\mathsf{new}\ K; \mathsf{send}_c\langle\{x\}_K\rangle; \mathsf{halt}$ preserves the secrecy of x.

Unfortunately too tedious proof for an extremely simple protocol.

Need simpler methods of showing security of a protocol....

After all this, we conclude:

the process $\mathsf{new}\ K; \mathsf{send}_c\langle\{x\}_K\rangle; \mathsf{halt}$ preserves the secrecy of x.

Unfortunately too tedious proof for an extremely simple protocol.

Need simpler methods of showing security of a protocol....

We define rules for controlling the flow of information in the protocol.

Information flow analysis for the Spi-calculus

- Information flow analaysis is used in various programming languages (imperative, functional, object-oriented languages, process calculi,...) to study security properties.

- Data is classified into various security levels representing varying degrees of confidentiality.

- A program is secure if information from more confidential data does not flow to less confidential data.

Consider the C language.

Assume variable x has security level high and variable y has security level low.

Then the following statement cannot be allowed in the program:

$$y^{low} = x^{high} + 1;$$

By reading the less confidential data y, we can get information about the high confidential data x.

The following statement is fine.

$$x^{high} = y^{low} + 1;$$

The following code should be disallowed.

$$
\begin{array}{l}
\mathrm{z} = 2 * \mathrm{x}^{high} + 1; \\
\mathrm{if}\ (\mathrm{z} > 100) \\
\quad \mathrm{y}^{low} = 10; \\
\mathrm{else} \\
\quad \mathrm{y}^{low} = 20;
\end{array}
$$

By observing y we can get some information about x.

$\longrightarrow$ Implicit flows should also be controlled.

For the Spi-calculus . . .

We classify data into three classes

secret	data which should not be leaked
public	data which can be communicated to anyone
any	arbitrary data

Subsumption relation on classes:

$$\begin{array}{lll} \textsf{secret} & \preceq \textsf{any} & \\ \textsf{public} & \preceq \textsf{any} & \\ T & \preceq T & \text{for } T \in \{\textsf{secret}, \textsf{public}, \textsf{any}\} \end{array}$$

Some initial ideas.

- Secret data should not be sent on public channels.
- Secret data should encrypted with public key should not be public.
- Public data encrypted with secret key may be made public.
- Data encrypted with private key may be made public.

We formulate these as a set of typing rules.

type of message M	= secrecy level of M
process P is well-types	= P does not allow bad information flow