Hence if M has type secret and N has type public then $\{M\}_N$ has type secret.

If M has type secret and c has type public then $\mathsf{send}_c\langle M\rangle; P$ is not well-typed.

If M has type public and c has type public then $\mathsf{send}_c\langle M\rangle; P$ is well-typed.

0 has type public.

halt is well-typed.

...

We consider the process $P(x) \triangleq \mathsf{send}_c\langle 0\rangle; \mathsf{halt};$ which is trivially secure.

For information flow analysis, we use an environment $E = \{c : \mathsf{public}(L)\}$ meaning c has type public (L will be explained later).

We have $E \vdash c : \mathsf{public}$ and $E \vdash 0 : \mathsf{public}$, hence the send action is well-typed.

Also $E \vdash \mathsf{halt}$, meaning that halt is well-typed in environment E.

Combining all these we get $E \vdash P(x)$.

An environment E provides information about the classes to which names and variables belong.

We define typing rules for the following kinds of judgments

	$\vdash E$	environment E is well formed
E	$\vdash M : T$	term M is of class T in environment E
E	$\vdash P$	process P is well typed in environment E

Consider again the insecure protocol $\mathsf{send}_c\langle\{0\}_x\rangle; \mathsf{halt}$.

We consider c to have level public. If we consider

x to be of type secret then $\{0\}_x$ is of type public and the process is well-typed. That's not what we want!

Solution: if we are interested in the secrecy of variable x then we consider x to have type any.

We always protect the data of level any as if it were secret, but can exploit it only as if it were public.

Informally we would like to show that if environment E has only any variables and public names and $E \vdash P$ then P does not leak any variables $x \in dom(E)$.

Consider: $\mathsf{send}_c\langle x\rangle; \mathsf{halt}$

Consider $E = \{x : \mathsf{any}, c : \mathsf{public} :: L\}$.

x is of level any but is sent out on c of level public, which will be forbidden by our typing rules.

Consider: $\mathsf{send}_c\langle\{0\}_x\rangle; \mathsf{halt}$

Consider $E = \{x : \mathsf{any}, c : \mathsf{public} :: L\}$.

Only terms of type secret or public can be used as encryption keys.

x is of type any, so the term $\{0\}_x$ has no type.

We follow some conventions for designing safe protocols.

$A \longrightarrow S : A, B$

$S \longrightarrow A : \{A, B, Na, \{Nb\}_{K_{sb}}\}_{K_{sa}}$

$A \longrightarrow B : \{Nb\}_{K_{sb}}$

A participant may play the role of A in one session and of B in another session.

We need a clear way of distinguishing the types of messages received and their components.

This is important only for messages sent on secret channels and for messages encrypted with secret keys.

We adopt the following standard format:

Messages sent on secret channels should have three components of levels secret, any and public respectively.

Messages encrypted with secret will similarly have secret, any and public components.

Consider protocol

$B \longrightarrow A : Nb$

$A \longrightarrow B : \{M, Nb\}_{K_{ab}}$

By replaying nonces, an attacker can find out whether the same M is sent more than once, or different ones.

$\longrightarrow$ leak of information.

Another example: the spi-calculus process

$$P(x, y) \triangleq \quad \mathsf{new}\ K; \quad (\mathsf{recv}_c(z); \mathsf{send}_c\langle\{x, z\}_K\rangle;$$
$$\mathsf{recv}_d(z); \mathsf{send}_d\langle\{y, z\}_K\rangle; \mathsf{halt})$$

By sending the same z twice, once on channcel c and once on d, one can know whether $x = y$.

In particular we have $P(0, 0) \not\simeq P(0, \mathsf{succ}\ (0))$.

Hence $P(x, y)$ does not preserve secrecy of x and y.

To prevent this we include an extra fresh nonce (confounder) in each message encrypted with secret keys.

A confounder is used at most once in an encrypted message.

Our previous protocol now becomes:

$B \longrightarrow A : Nb$

$A \longrightarrow B : \{M, Nb, Na\}_{K_{ab}}$

And the previous spi-calculus process becomes

$$P(x,y) \triangleq \quad \mathsf{new}\ K; \quad (\mathsf{recv}_c(z); \mathsf{new}\ m; \mathsf{send}_c\langle\{x,z,m\}_K\rangle; \\ \mathsf{recv}_d(z); \mathsf{new}\ n; \mathsf{send}_d\langle\{y,z,n\}_K\rangle; \mathsf{halt})$$

Combining with the ides of message formatting, we arrive at the following format for messages encrypted with secret keys:

$$\{M_1, M_2, M_3, n\}_K$$

where M_1 : secret, M_2 : any, M_3 : public, and n is the confounder.

n can be used as confounder only in this term and nowhere else.

This information is remembered by the environment E.

If $n : T :: \{M_1, M_2, M_3, n\}_K \in E$ then

then n can be used as a confounder only in $\{M_1, M_2, M_3, n\}_K$.

The typing rules

Well formed environments

$$\vdash \emptyset$$

$$\frac{\vdash E \qquad x \notin dom(E)}{\vdash E, x : T}$$

The empty environment $\emptyset$ is well-formed.

$dom(E)$ denotes the variables and names about which E has the typing information.

For names, the environment should provide a type as well as information about where it is used as a confounder.

$$\frac{\vdash E \quad E \vdash M_1 : T_1 \quad \dots \quad E \vdash M_k : T_k \quad n \notin dom(E) \quad E \vdash N : R}{\vdash E, n : T :: \{M_1, \dots, M_k, n\}_N}$$

The environment $\{x : \mathsf{secret}, m : \mathsf{any} :: \{m\}_0, y : \mathsf{public}, K : \mathsf{secret} :: \{K\}_0, n : \mathsf{public} :: \{x, m, y, n\}_K\}$ is well-formed.

The environment $\{x : \mathsf{secret}, m : \mathsf{any} :: \{m\}_0, y : \mathsf{public}, n : \mathsf{public} :: \{x, m, y, n\}_K, K : \mathsf{secret} :: \{K\}_0\}$ is not well-formed.

Environment lookups and subsumption:

$$\frac{E \vdash M : T \qquad T \sqsubseteq R}{E \vdash M : R}$$

$$\frac{\vdash E \qquad x : T \in E}{E \vdash x : T}$$

$$\frac{\vdash E \qquad n : T :: \{M_1, \ldots, M_k, n\}_N \in E}{E \vdash n : T}$$

Data of type public and secret are also of type any.

$$\frac{\vdash E}{E \vdash 0 : \mathsf{public}}$$

$$\frac{E \vdash M : T}{E \vdash \mathsf{succ}\,(M) : T}$$

$$\frac{E \vdash M : T \qquad E \vdash N : T}{E \vdash \langle M, N \rangle : T}$$

Hence if $E = \{x : \mathsf{public}, y : \mathsf{secret}\}$ then $E \vdash \langle x, y \rangle : \mathsf{any}$.

Encryption

$$\frac{E \vdash M_1 : T \quad \ldots \quad E \vdash M_k : T \quad E \vdash N : \mathsf{public} \quad T = \mathsf{public} \text{ if } k = 0}{E \vdash \{M_1, \ldots, M_k\}_N : T}$$

$$\frac{\begin{array}{ccc} E \vdash M_1 : \mathsf{secret} & E \vdash M_2 : \mathsf{any} & E \vdash M_3 : \mathsf{public} \\ E \vdash N : \mathsf{secret} & & n : T :: \{M_1, M_2, M_3, n\}_N \in E \end{array}}{E \vdash \{M_1, M_2, M_3, n\}_N : \mathsf{public}}$$

Keys of type any are not used for encryption.

$$\frac{E \vdash M : \mathsf{public} \quad E \vdash M_1 : \mathsf{public} \quad \ldots \quad E \vdash M_k : \mathsf{public} \quad E \vdash P}{E \vdash \mathsf{send}_M \langle M_1, \ldots, M_k \rangle; P}$$

$$\frac{E \vdash M : \mathsf{secret} \quad E \vdash M_1 : \mathsf{secret} \quad E \vdash M_2 : \mathsf{any} \quad E \vdash M_3 : \mathsf{public} \quad E \vdash P}{E \vdash \mathsf{send}_M \langle M_1, M_2, M_3 \rangle; P}$$

Only public data may be sent on public channels.

On secret channels, data is always sent in our standard format.

Channels of type any are not used.

We consider pairing as left-associative.

For example (M_1, M_2, M_3, M_4) is same as $((M_1, M_2), M_3, M_4)$

Similar rules for inputs.

$$\frac{E \vdash M : \mathsf{public} \quad E, x_1 : \mathsf{public}, \ldots, x_k : \mathsf{public} \vdash P}{E \vdash \mathsf{recv}_M(x_1, \ldots, x_k); P}$$

$$\frac{E \vdash M : \mathsf{secret} \quad E, x_1 : \mathsf{secret}, x_2 : \mathsf{any}, x_3 : \mathsf{public} \vdash P}{E \vdash \mathsf{recv}_M(x_1, x_2, x_3); P}$$

The appropriate class information for the input variables is added to the environment, and the new environment is used for typing the remaining process.

Let $E = \{c : \mathsf{public} :: \{c\}_0\}$ and $P = \mathsf{recv}_c(x); \mathsf{send}_c\langle x\rangle; \mathsf{halt}$.

To show that $E \vdash P$

we consider $E' = E, x : \mathsf{public}$ and show that $E' \vdash \mathsf{send}_c\langle x\rangle; \mathsf{halt}$.

$$\frac{\vdash E}{E \vdash \mathsf{halt}} \qquad \frac{E \vdash P \qquad E \vdash Q}{E \vdash P \mid Q}$$

$$\frac{E \vdash P}{E \vdash \mathsf{repeat}\ P} \qquad \frac{E, n : T :: L \vdash P}{E \vdash \mathsf{new}\ n; P}$$

The newly created name can be chosen to be kept secret or can be revealed, and can be chosen to used as a confounder in some message.

L could be anything trivial if we don't want to use n as a confounder.

$$\frac{E \vdash M : T \qquad E \vdash N : R \qquad E \vdash P \qquad T, R \in \{\mathsf{public}, \mathsf{secret}\}}{E \vdash \mathsf{check}\ (M == N); P}$$

Equality checks are not allowed on data of class any to prevent implicit information flow.

Equality checks are freely allowed among data of type public and secret!

Example Consider $P \triangleq \mathsf{recv}_c(y); \mathsf{check}\ (x == y); \mathsf{send}_c\langle 0\rangle; \mathsf{halt}$ where x is the data whose secrecy we are interested in.

Secrecy of x is not maintained. $P[M/x]$ and $P[M'/x]$ are not equivalent for $M \neq M'$.

Consider test $(Q, \overline{d})$ where $Q \triangleq \mathsf{send}_c\langle M\rangle; \mathsf{recv}_c(z); \mathsf{send}_d\langle 0\rangle; \mathsf{halt}$.

$P[M/x] \mid Q$ passes the test:

$P[M/x] \mid Q \xrightarrow{\tau} \mathsf{check}\ (M = M); \mathsf{send}_c\langle 0\rangle; \mathsf{halt} \mid \mathsf{recv}_c(z); \mathsf{send}_d\langle 0\rangle; \mathsf{halt} \xrightarrow{\tau}$ $\mathsf{halt} \mid \mathsf{send}_d\langle 0\rangle; \mathsf{halt} \xrightarrow{\overline{d}} \langle 0\rangle(\mathsf{halt} \mid \mathsf{halt})$

$P[M'/x] \mid Q$ does not pass the test.

Similarly, case analysis on data of class any are disallowed.

$$\frac{E \vdash M : T \qquad E, x : T, y : T \vdash P \qquad T \in \{\mathsf{public}, \mathsf{secret}\}}{E \vdash \mathsf{let}\ (x, y) = M; P}$$

$$\frac{E \vdash M : T \qquad E \vdash P \qquad E, x : T \vdash Q \qquad T \in \{\mathsf{secret}, \mathsf{public}\}}{E \vdash \mathsf{case}\ M\ \mathsf{of}\ 0 : P,\ \mathsf{succ}\ (x) : Q}$$

Decryption

$$\frac{E \vdash L : T \quad E \vdash N : \mathsf{public} \quad E, x_1 : T, \ldots, x_k : T \vdash P \quad T \in \{\mathsf{secret}, \mathsf{public}\}}{E \vdash \mathsf{case}\ L\ \mathsf{of}\ \{x_1, \ldots, x_k\}_N : P}$$

$$\frac{\begin{array}{c} E \vdash L : T \qquad E \vdash N : \mathsf{secret} \qquad T \in \{\mathsf{secret}, \mathsf{public}\} \\ E, x_1 : \mathsf{secret}, x_2 : \mathsf{any}, x_3 : \mathsf{public}, x_4 : \mathsf{any} \vdash P \end{array}}{E \vdash \mathsf{case}\ L\ \mathsf{of}\ \{x_1, x_2, x_3, x_4\}_N : P}$$

The confounder x_4 in the second rule is assumed to be of type any because we have no more information about it.

Typing implies noleak of information

Suppose

- $\vdash E$
- all variables in $dom(E)$ are of level any and all names in $dom(E)$ are of level public.
- $E \vdash P$
- P has free variables $x_1, \ldots, x_k$
- $fn(M_i), fn(M_i') \subseteq dom(E)$ for $1 \leq i \leq k$.

then $P[M_1/x_1, \ldots, M_k/x_k] \simeq P[M_1'/x_1, \ldots, M_k'/x_k]$

Well typed processes maintain secrecy of the free variables $(x_1, \ldots, x_k)$, i.e. they are not leaked.

Our previous example $P \triangleq \mathsf{recv}_c(y); \mathsf{check}\ (x == y); \mathsf{send}_c\langle 0\rangle; \mathsf{halt}$

We take $E \triangleq \{x : \mathsf{any}, c : \mathsf{public} :: \{c\}_0\}$. c is not meant to be used as a confounder, hence we have the dummy term $\{c\}_0$.

We have $\vdash E$.

In order to show $E \vdash P$ we need to find some T such that

$E, y : \mathsf{public} \vdash \mathsf{check}\ (x == y); \mathsf{send}_c\langle 0\rangle; \mathsf{halt}$.

But this is impossible because equality checks should not involve data of class any.

Hence the process doesn't type-check, as required.

Consider $P \triangleq \mathsf{new}\ K; \mathsf{new}\ m; \mathsf{new}\ n; \mathsf{send}_c\langle\{m, x, 0, n\}_K\rangle; \mathsf{halt}$.

We take $E \triangleq \{x : \mathsf{any}, c : \mathsf{public} :: \{c\}_0\}$. We have $\vdash E$.

To show $E \vdash P$ we choose
$E' \triangleq E, K : \mathsf{secret} :: \{K\}_0, m : \mathsf{secret} :: \{m\}_0, n : \mathsf{secret} :: \{m, x, 0, n\}_K$
and show that $E' \vdash \mathsf{send}_c\langle\{m, x, 0, n\}_K\rangle; \mathsf{halt}$.

This is ok because $E' \vdash m : \mathsf{secret}$, $E' \vdash x : \mathsf{any}$, $E' \vdash 0 : \mathsf{public}$, $E' \vdash n : \mathsf{secret}$, $E' \vdash K : \mathsf{secret}$ and $E' \vdash \mathsf{halt}$.