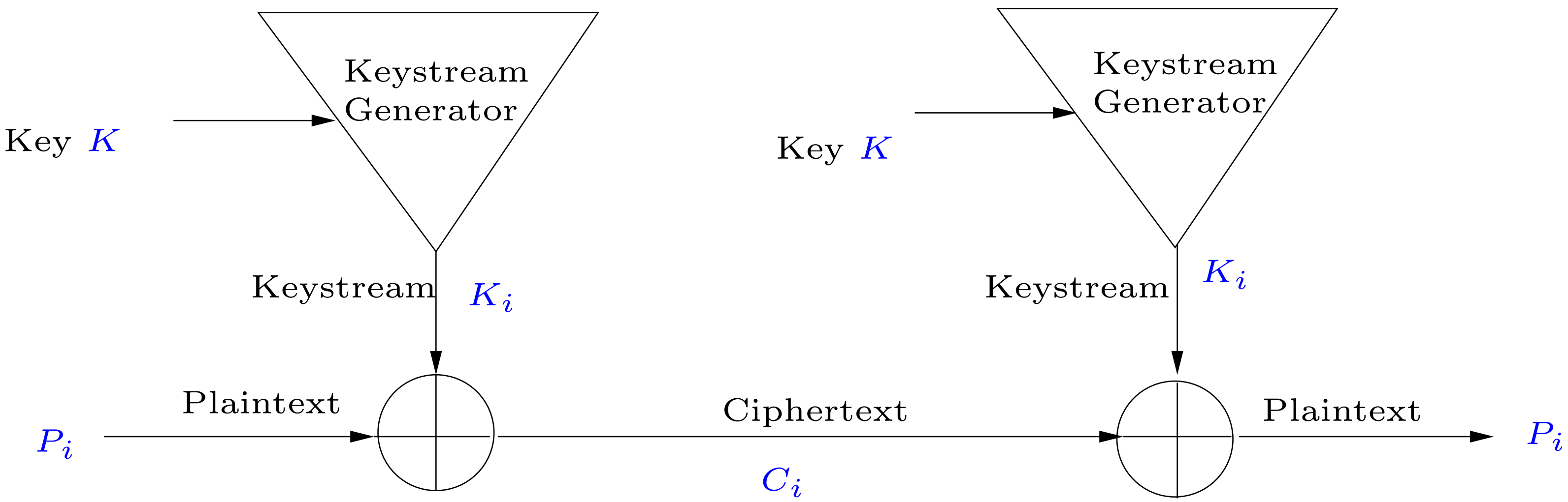
Stream Ciphers

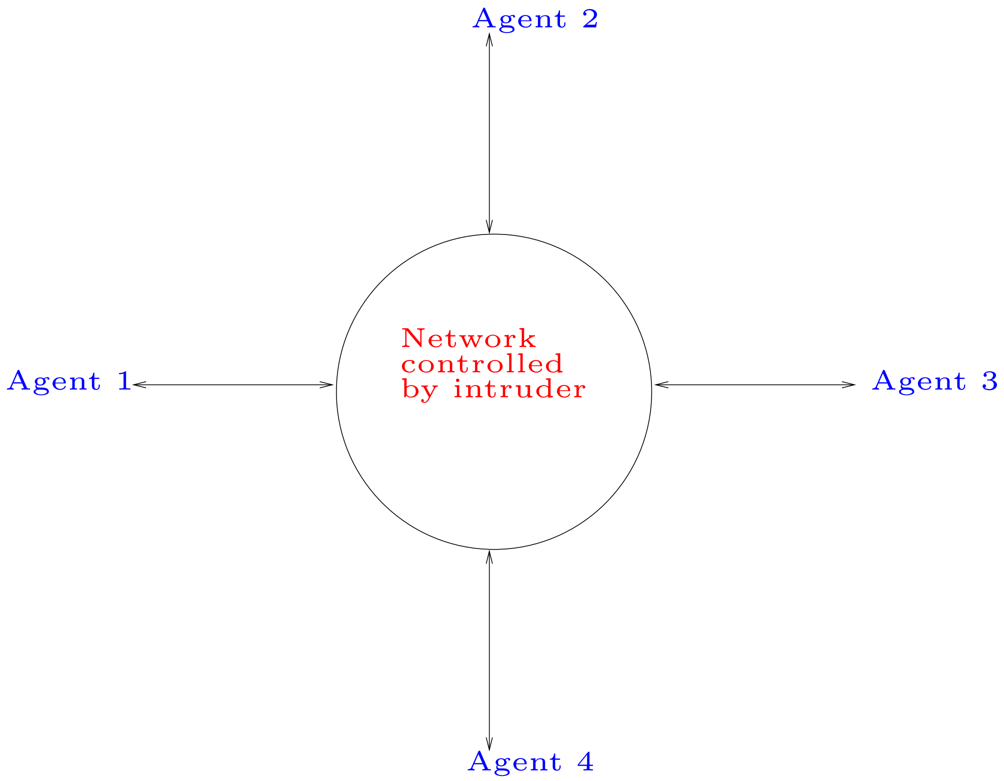


$$C_i = P_i \oplus K_i \qquad P_i = C_i \oplus K_i$$

The intruder deduction problem

The Dolev Yao model

A set of assumptions to make reasoning about cryptographic protocols feasible. Based on the paper *On the security of public key protocols* by D. Dolev and A. C. Yao (1983).



First part of the assumptions: the network is completely insecure.

- All messages sent by agents are actually sent to the intruder.
- All messages received by agents are actually received by the intruder.
- Hence the intruder can read all messages, delete them, modify them, ...
- The intruder also remembers all messages ever sent over the network.
- Besides, the intruder can create nonces, encrypt and decrypt messages using known keys...

Second part of the assumptions: perfect cryptography

- The plaintext m cannot be obtained from the ciphertext $\{m\}_k$ except by knowing the key k^{-1}.
- If $\{m\}_k = \{m'\}_{k'}$ then $m = m'$ and $k = k'$.
- $\{\ldots\{m\}_k\ldots\}_k \neq m$.
- A nonce is distinct from other nonces and messages
- $\ldots$

This is summarized by considering messages as symbolic terms.

$m ::=$	c	constants: identities, nonces
	$\mathtt{enc}(m_1, m_2)$	representing $\{m_1\}_{m_2}$
	$\mathtt{pair}(m_1, m_2)$	representing pair $\langle m_1, m_2 \rangle$
	$\mathtt{h}(m)$	hash
	$\ldots$	

$m ::=$	c	constants: identities, nonces
	$\mathtt{enc}(m_1, m_2)$	representing $\{m_1\}_{m_2}$
	$\mathtt{pair}(m_1, m_2)$	representing pair $\langle m_1, m_2 \rangle$
	$\mathtt{h}(m)$	hash
	$\ldots$	

We have rules of the form:

If intruder knows m_1 and he knows m_2 then he knows $\{m_1\}_{m_2}$.

If intruder knows $\mathtt{enc}(m_1, m_2)$ and he knows m_2 then he knows m_1.

$\ldots$

Hence starting from a given set of messages an intruder can compute possibly infinitely many new messages.

Suppose the intruder knows the messages

$\{\{m_1\}_{m_5}\}_{\langle m_2,m_3\rangle} \quad \{m_5\}_{m_6} \quad \{m_2\}_{m_4} \quad \{m_3\}_{m_4} \quad m_4$

Can the intruder compute the message $\{m_1\}_{m_5}$?

Suppose the intruder knows the messages

$\{\{m_1\}_{m_5}\}_{\langle m_2, m_3\rangle} \quad \{m_5\}_{m_6} \quad \{m_2\}_{m_4} \quad \{m_3\}_{m_4} \quad m_4$

Can the intruder compute the message $\{m_1\}_{m_5}$?

Yes. He computes: m_2, m_3, $\langle m_2, m_3\rangle$, $\{m_1\}_{m_5}$

Suppose the intruder knows the messages

$\{\{m_1\}_{m_5}\}_{\langle m_2,m_3\rangle} \quad \{m_5\}_{m_6} \quad \{m_2\}_{m_4} \quad \{m_3\}_{m_4} \quad m_4$

Can the intruder compute the message $\{m_1\}_{m_5}$?

Yes. He computes: m_2, m_3, $\langle m_2, m_3\rangle$, $\{m_1\}_{m_5}$

Can he compute m_1 ?

Suppose the intruder knows the messages

$\{\{m_1\}_{m_5}\}_{\langle m_2, m_3\rangle} \quad \{m_5\}_{m_6} \quad \{m_2\}_{m_4} \quad \{m_3\}_{m_4} \quad m_4$

Can the intruder compute the message $\{m_1\}_{m_5}$?

Yes. He computes: m_2, m_3, $\langle m_2, m_3\rangle$, $\{m_1\}_{m_5}$

Can he compute m_1 ? No.

Suppose the intruder knows the messages

$\{\{m_1\}_{m_5}\}_{\langle m_2,m_3\rangle} \quad \{m_5\}_{m_6} \quad \{m_2\}_{m_4} \quad \{m_3\}_{m_4} \quad m_4$

Can the intruder compute the message $\{m_1\}_{m_5}$?

Yes. He computes: m_2, m_3, $\langle m_2, m_3\rangle$, $\{m_1\}_{m_5}$

Can he compute m_1 ? No.

Given a finite set S of messages and a message m how to check if the intruder can compute m from S ? ...and efficiently.

This is called the intruder deduction problem : $S \stackrel{?}{\vdash} m$.

The most basic problem related to the secrecy property of cryptographic protocols. At least it solves the secrecy problem for a passive intruder for finitely many sessions.

Rules expressing the intruder knowledge

Member: $$\frac{}{S \vdash m}\; m \in S$$

Intruder can synthesize messages

Pairing: $$\frac{S \vdash m_1 \quad S \vdash m_2}{S \vdash \langle m_1, m_2 \rangle}$$

Encryption: $$\frac{S \vdash m_1 \quad S \vdash m_2}{S \vdash \{m_1\}_{m_2}}$$

FunctionApplication: $$\frac{S \vdash m_1 \quad \ldots \quad S \vdash m_n}{S \vdash h(m_1, \ldots, m_n)}$$

Intruder can analyze messages

$$\text{Unpairing:} \qquad \frac{S \vdash \langle m_1, m_2 \rangle}{S \vdash m_i}\; i \in \{1,2\}$$

$$\text{DecryptionSym:} \qquad \frac{S \vdash \{m_1\}_{m_2} \quad S \vdash m_2}{S \vdash m_1}\; m_2 \text{ is symmetric}$$

$$\text{DecryptionAsym:} \qquad \frac{S \vdash \{m\}_k \quad S \vdash k'}{S \vdash m}\; k' \text{ is inverse key of } k$$

We consider asymmetric encryption to involve only atomic keys.

$$\text{FunctionInversion:} \qquad \frac{S \vdash h(m_1, \ldots, m_n)}{S \vdash m_i}\; h \text{ is invertible, } 1 \leq i \leq n$$

Example of a derivation using the inference rules.

Let $S = \{\langle m_1, m_2\rangle, m_3, \{m_4\}_{\langle m_1, m_3\rangle}\}$

Then m_4 is deducible from S:

$$\dfrac{\dfrac{}{S \vdash \{m_4\}_{\langle m_1, m_3\rangle}} \qquad \dfrac{\dfrac{\dfrac{}{S \vdash \langle m_1, m_2\rangle}}{S \vdash m_1} \qquad \dfrac{}{S \vdash m_3}}{S \vdash \langle m_1, m_3\rangle}}{S \vdash m_4}$$

The intruder deduction problem is now: is there a derivation of $S \vdash m$ using these rules ?

Here is another derivation possible of $S \vdash m_4$!

$$\dfrac{\dfrac{\dfrac{\dfrac{}{S \vdash \{m_4\}_{\langle m_1, m_3 \rangle}} \quad \dfrac{\dfrac{\dfrac{}{S \vdash \langle m_1, m_2 \rangle}}{S \vdash m_1} \quad \dfrac{}{S \vdash m_3}}{S \vdash \langle m_1, m_3 \rangle}}{S \vdash m_4} \quad \dfrac{}{S \vdash m_3}}{S \vdash \{m_4\}_{m_3}} \quad \dfrac{}{S \vdash m_3}}{S \vdash m_4}$$

All derivations are not short enough!

Simplification rules

1. The derivation

$$\dfrac{\dfrac{\begin{array}{c}\delta_1\\ \vdots\\ S \vdash m_1\end{array} \qquad \begin{array}{c}\delta_2\\ \vdots\\ S \vdash m_2\end{array}}{S \vdash \langle m_1, m_2 \rangle}}{S \vdash m_i}$$

where $i \in \{1, 2\}$ can be simplified to

$$\begin{array}{c}\delta_1\\ \vdots\\ S \vdash m_1\end{array}$$

2. The derivation

$$\dfrac{\dfrac{\begin{array}{c}\delta_1\\ \vdots\\ S \vdash m_1\end{array} \quad \begin{array}{c}\delta_2\\ \vdots\\ S \vdash m_2\end{array}}{S \vdash \{m_1\}_{m_2}} \quad \begin{array}{c}\delta_3\\ \vdots\\ S \vdash m_2'\end{array}}{S \vdash m_1}$$

can be simplified to

$$\begin{array}{c}\delta_1\\ \vdots\\ S \vdash m_1\end{array}$$

3. The derivation

$$\dfrac{\dfrac{\begin{matrix}\delta_1 \\ \vdots \\ S \vdash m_1\end{matrix} \qquad \ldots \qquad \begin{matrix}\delta_2 \\ \vdots \\ S \vdash m_2\end{matrix}}{S \vdash h(m_1, \ldots, m_n)}}{S \vdash m_i}$$

can be simplified to

$$\begin{matrix}\delta_i \\ \vdots \\ S \vdash m_i\end{matrix}$$

Given any derivation, we can repeatedly apply the above simplification rules on the subderivations to get simpler derivations.

This process always terminates, because the simplification rules strictly decrease the size of the derivation.

Define normal derivation as a derivation which cannot be simplified further.

If $S \vdash m$ is derivable using our inference rules then there is a normal derivation of $S \vdash m$ using these inference rules.s

Let $sub(S)$ denote the set of all subterms of terms in the set S.

The subterm property: If

$$\begin{array}{c} \delta \\ \vdots \\ S \vdash m \end{array}$$

is a normal derivation, then δ involves only the messages in $sub(S \cup \{m\})$.

Hence to check whether some message m is deducible from S, we don't need to deduce arbitrarily large messages.

Proof: Do induction on the size of the derivation of $S \vdash m$.

Case 1: δ is of the from

$$\frac{}{S \vdash m}$$

where $m \in S$. There is nothing to prove.

Case 2: $m = \langle m_1, m_2 \rangle$ and δ is of the form

$$\dfrac{\begin{array}{c}\delta_1\\ \vdots\\ S \vdash m_1\end{array} \qquad \begin{array}{c}\delta_2\\ \vdots\\ S \vdash m_2\end{array}}{S \vdash \langle m_1, m_2 \rangle}$$

By induction hypothesis δ_i involves only messages in $sub(S \cup \{m_i\})$.
Hence δ involves only the messages from $S \cup \{\langle m_1, m_2 \rangle\}$.

Case 3: $m = \{m_1\}_{m_2}$ and δ is of the form

$$\dfrac{\begin{array}{c}\delta_1\\ \vdots\\ S \vdash m_1\end{array} \qquad \begin{array}{c}\delta_2\\ \vdots\\ S \vdash m_2\end{array}}{S \vdash \{m_1\}_{m_2}}$$

By induction hypothesis δ_i involves messages in $sub(S \cup \{m_i\})$.

Hence δ involves only the messages from $S \cup \{\langle m_1, m_2\rangle\}$.

Case 4: $m = h(m_1, \ldots, m_n)$ and δ is of the form

$$\dfrac{\begin{array}{c}\delta_1\\ \vdots\\ S \vdash m_1\end{array} \qquad \ldots \qquad \begin{array}{c}\delta_n\\ \vdots\\ S \vdash m_n\end{array}}{S \vdash h(m_1, \ldots, m_n)}$$

By induction hypothesis δ_i involves only messages in $sub(S \cup \{m_i\})$. Hence δ involves only the messages from $S \cup \{h(m_1, \ldots, m_n)\}$.

Case 5: δ is of the form

$$\dfrac{\begin{array}{c}\delta' \\ \vdots \\ S \vdash \langle m_1, m_2 \rangle\end{array}}{S \vdash m_i}$$

for some $i \in \{1, 2\}$. By induction hypothesis, δ' involves only messages from $sub(S \cup \{\langle m_1, m_2 \rangle\})$.

If $\langle m_1, m_2 \rangle \in sub(S)$ then there is nothing to prove.
Otherwise consider the last inference rule used in δ'.

The last inference rule used in δ' cannot be the 'Member' rule because $\langle m_1, m_2 \rangle \notin sub(S)$.

The last inference rule used in δ' cannot be an analysis rule because it would involve strict superterms of $\langle m_1, m_2 \rangle$ which are not in $sub(S)$.

The last inference rule used in δ' cannot be 'Encryption' or 'FunctionApplication' because they don't create pairs.

Hence δ' is of the from

$$\dfrac{\begin{array}{c}\delta_1\\ \vdots \\ S \vdash m_1\end{array} \qquad \begin{array}{c}\delta_2\\ \vdots \\ S \vdash m_2\end{array}}{S \vdash \langle m_1, m_2 \rangle}$$

In other words, δ is of the form

$$\dfrac{\dfrac{\begin{matrix}\delta_1 \\ \vdots \\ S \vdash m_1\end{matrix} \qquad \begin{matrix}\delta_2 \\ \vdots \\ S \vdash m_2\end{matrix}}{S \vdash \langle m_1, m_2 \rangle}}{S \vdash m_i}$$

But then δ can be simplified to the derivation δ_i.

Hence δ is not normal, giving a contradiction.

Case 6: δ is of the form

$$\dfrac{\begin{array}{c}\delta_1 \\ \vdots \\ S \vdash \{m\}_{m_1}\end{array} \qquad \begin{array}{c}\delta_2 \\ \vdots \\ S \vdash m_1\end{array}}{S \vdash m}$$

By induction hypothesis, δ_1 involves only messages from $sub(S \cup \{\{m\}_{m_1}\})$.

If $\{m\}_{m_1} \in sub(S)$ then there is nothing to prove.

Otherwise consider the last inference rule used in δ_1.

The last inference rule used in δ_1 cannot be the 'Member' rule because $\{m\}_{m_1} \notin sub(S)$.

The last inference rule used in δ_1 cannot be an analysis rule because they would involve strict superterms of $\{m\}_{m_1}$ which are not in $sub(S)$.

The last inference rule used in δ' cannot be 'Pairing' or 'FunctionApplication' because they cannot create $\{m\}_{m_1}$. Hence δ_1 is of the from

$$\dfrac{\begin{array}{c}\delta_1' \\ \vdots \\ I \vdash m\end{array} \quad \begin{array}{c}\delta_1'' \\ \vdots \\ I \vdash m_1\end{array}}{I \vdash \{m\}_{m_1}}$$

But then δ is not normal, giving contradiction.

Case 7: δ is of the form

$$\dfrac{\begin{matrix}\delta_1 & & \delta_2 \\ \vdots & & \vdots \\ S \vdash \{m\}_k & & S \vdash k'\end{matrix}}{S \vdash m}$$

By induction hypothesis, δ_2 involves only messages from $sub(S) \cup \{k'\}$. If $k' \notin sub(S)$ then we have a contradiction. Hence δ_2 involves only messages from $sub(S)$.

By induction hypothesis, δ_1 involves only messages from $sub(S \cup \{\{m\}_k\})$.

If $\{m\}_k \in sub(S)$ then there is nothing to prove.
Otherwise consider the last inference rule used in δ_1.

The last inference rule used in δ_1 cannot be the 'Member' rule because $\{m\}_k \notin sub(S)$.

The last inference rule used in δ_1 cannot be an analysis rule because they would involve strict superterms of $\{m\}_k$ which are not in $sub(S)$.

The last inference rule used in δ' cannot be 'Pairing' or 'FunctionApplication' because they cannot create $\{m\}_k$. Hence δ_1 is of the from

$$\dfrac{\begin{array}{c}\delta_1' \\ \vdots \\ I \vdash m\end{array} \quad \begin{array}{c}\delta_1'' \\ \vdots \\ I \vdash k\end{array}}{I \vdash \{m\}_k}$$

But then δ is not normal, giving contradiction.

Case 8: δ is of the form

$$\dfrac{\begin{array}{c}\delta' \\ \vdots \\ S \vdash h(m_1, \ldots, m_n)\end{array}}{S \vdash m_i}$$

for some $i \in \{1, \ldots, n\}$. By induction hypothesis, δ' involves only messages from $sub(S \cup \{h(m_1, \ldots, m_n)\})$.

If $h(m_1, \ldots, m_n) \in sub(S)$ then there is nothing to prove.
Otherwise consider the last inference rule used in δ'.

The last inference rule used in δ' cannot be the 'Member' rule because $h(m_1, \ldots, m_2) \notin sub(S)$. The last inference rule used in δ' cannot be an analysis rule because they would involve strict superterms of $h(m_1, \ldots, m_n)$ which are not in $sub(S)$.

The last inference rule used in δ' cannot be 'Encryption' or 'FunctionApplication' because they don't create pairs. Hence δ' is of the from

$$\dfrac{\begin{array}{c}\delta_1\\ \vdots\\ S \vdash m_1\end{array} \quad \ldots \quad \begin{array}{c}\delta_2\\ \vdots\\ S \vdash m_n\end{array}}{S \vdash h(m_1, \ldots, m_n)}$$

But then δ is not normal, giving a contradiction.

End of proof

Algorithm for the intruder deduction problem:

Input: set S of messages and message m

BEGIN

$T \leftarrow sub(S \cup \{m\})$.

$X \leftarrow \emptyset$

REPEAT

if some $m' \in T$ can be obtained from the messages in X

using one of the inference rules, and $m' \notin X$

then $X \leftarrow X \cup \{m'\}$.

UNTIL no new messages can be added

If $m \in X$ then RETURN 'yes' else RETURN 'no'

END

Define $size(m)$ to be the size of the DAG-representation of m and $size(S) = \sum_{m' \in S} size(m')$.

We have $cardinality(T) \leq size(S) + size(m)$.

The REPEAT-UNTIL loop is executed at most $cardinality(T)$ times.

Each execution of the loop takes time polynomial in $cardinality(T)$.

Conclusion: The intruder deduction problem can be decided in polynomial time.

Define $size(m)$ to be the size of the DAG-representation of m and $size(S) = \sum_{m' \in S} size(m')$.

We have $cardinality(T) \leq size(S) + size(m)$.

The REPEAT-UNTIL loop is executed at most $cardinality(T)$ times.

Each execution of the loop takes time polynomial in $cardinality(T)$.

Conclusion: The intruder deduction problem can be decided in polynomial time.

Exercise: how to do this is linear time ?

Special case: the two-phase intruder

In the above discussion, compound messages were allowed as keys for symmetric encryption. We now allow only atomic keys to be used for encryption and decryption.

Then it is sufficient for the intruder to deduce new messages from a given set S of messages in two phases:

Analysis phase: the intruder applies the analysis rules to get simpler messages.

Synthesis phase: the intruder applies the synthesis rules to create complex messages starting from the messages from the previous step.

This does not hold if compound keys are used:

Let $S = \{\langle m_1, m_2\rangle, m_3, \{m_4\}_{\langle m_1, m_3\rangle}\}$

$$\dfrac{\dfrac{}{S \vdash \{m_4\}_{\langle m_1, m_3\rangle}} \qquad \dfrac{\dfrac{\dfrac{}{S \vdash \langle m_1, m_2\rangle}}{S \vdash m_1} \qquad \dfrac{}{S \vdash m_3}}{S \vdash \langle m_1, m_3\rangle}}{S \vdash m_4}$$

The above derivation is normal but an analysis step occurs after a synthesis step.

Fact: If keys are restricted to be atomic then in any normal derivation, no analysis rule occurs after a synthesis rule.

Proof: Consider all the analysis rules.

1.
$$\dfrac{\begin{array}{c}\delta\\ \vdots\\ S \vdash \langle m_1, m_2 \rangle\end{array}}{S \vdash m_i}$$

The only synthesis rule that can occur above it is the pairing rule. But then the derivation would not be normal.

2. The main case:

$$\dfrac{\begin{matrix}\delta_1 \\ \vdots \\ S \vdash \{m_1\}_{m_2}\end{matrix} \qquad \begin{matrix}\delta_2 \\ \vdots \\ S \vdash m_2\end{matrix}}{S \vdash m_1}$$

By assumption m_2 is atomic hence no synthesis rule can occur in δ_2. The only synthesis rule that can occur above above $S \vdash \{m_1\}_{m_2}$ is an encryption rule. But then the derivation would not be normal.

3.

$$\dfrac{\begin{matrix}\delta_1 \\ \vdots \\ S \vdash \{m\}_k\end{matrix} \qquad \begin{matrix}\delta_2 \\ \vdots \\ S \vdash k'\end{matrix}}{S \vdash m}$$

As the keys k, k' are atomic this case is similar to the previous one.

4.

$$\dfrac{\begin{array}{c}\delta\\ \vdots\\ S \vdash h(m_1, \ldots, m_n)\end{array}}{S \vdash m_i}$$

The only synthesis rule that can occur above it is the 'FunctionApplication' rule. But then the derivation would not be normal.

End of proof

Hence the algorithm for the intruder deduction problem in the case of atomic keys can be modified to first exhaustively apply the analysis rules and then exhaustively apply the synthesis rules.

The TLS protocol

TLS (Transport Layer Security) Version 1.0

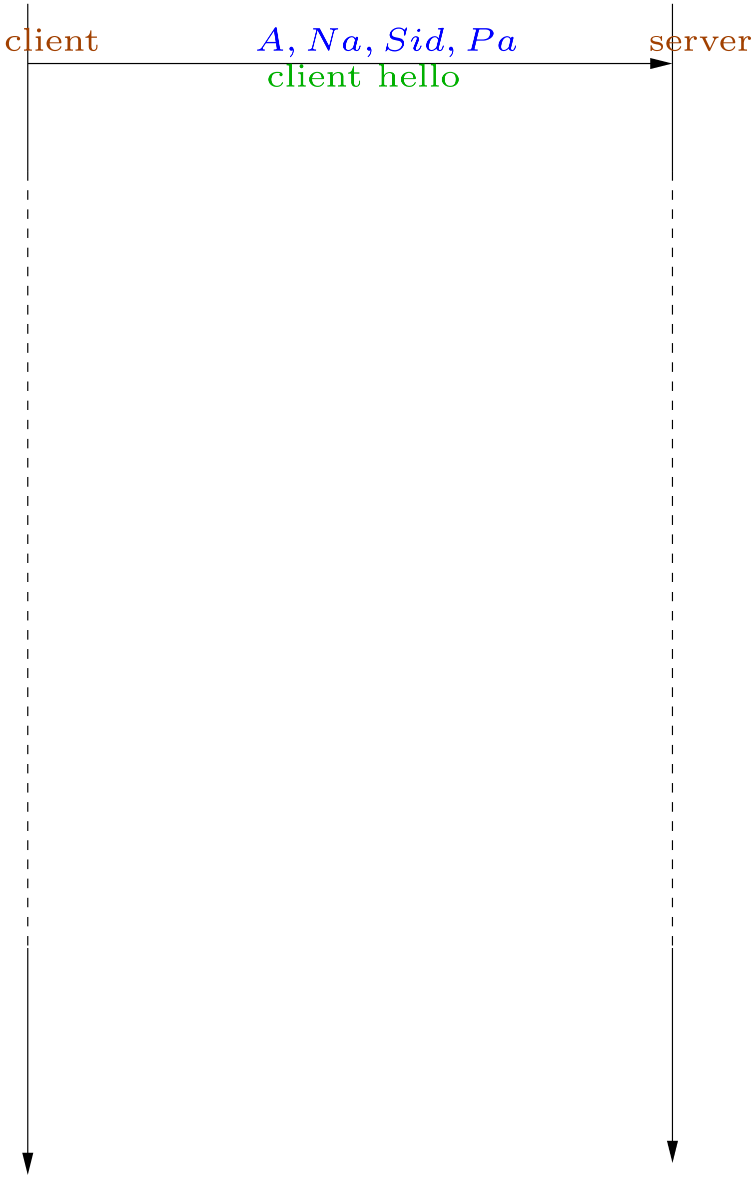
succeeds

SSL (Secure Sockets Layer) Version 3.0

Designed for secure communication on the Internet e.g. for commercial transactions.

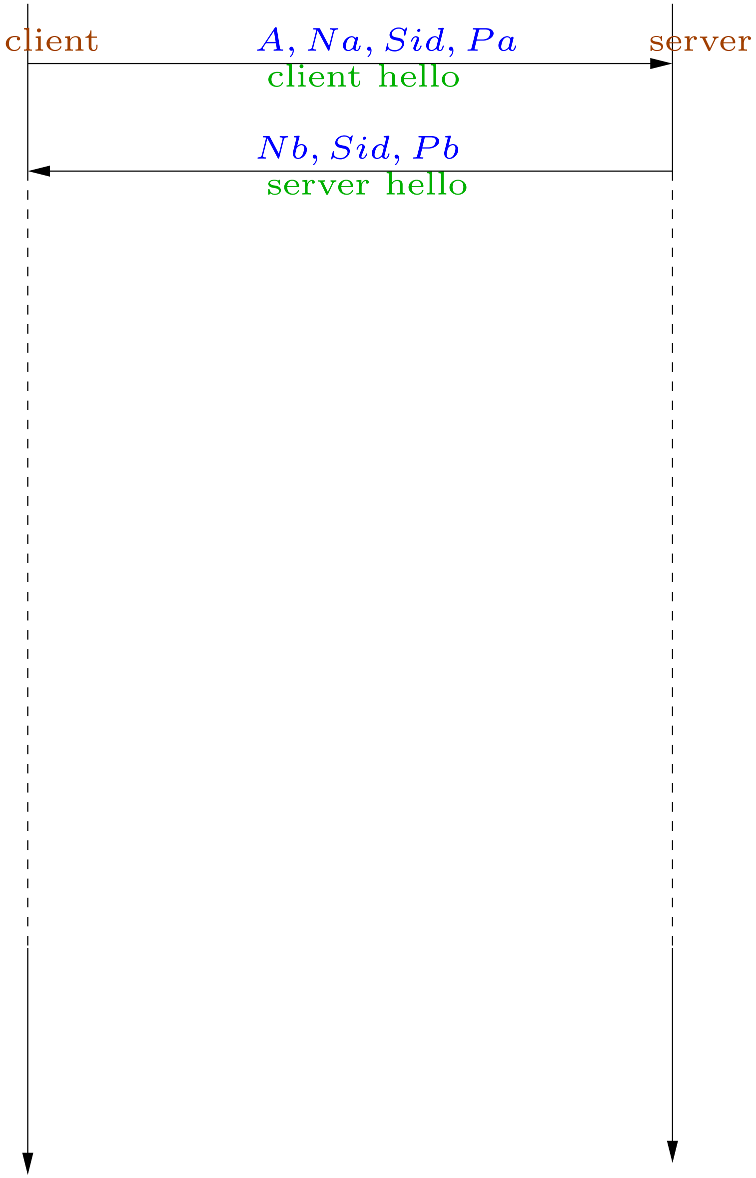
We will use the analysis of this protocol by
Lawrence C. Paulson (Univ. Cambridge):
Inductive Analysis of the Internet Protocol TLS
Transactions on Computer and System Security 2(3):332-351, 1999.

Such an analysis involves some abstractions from the actual protocol.

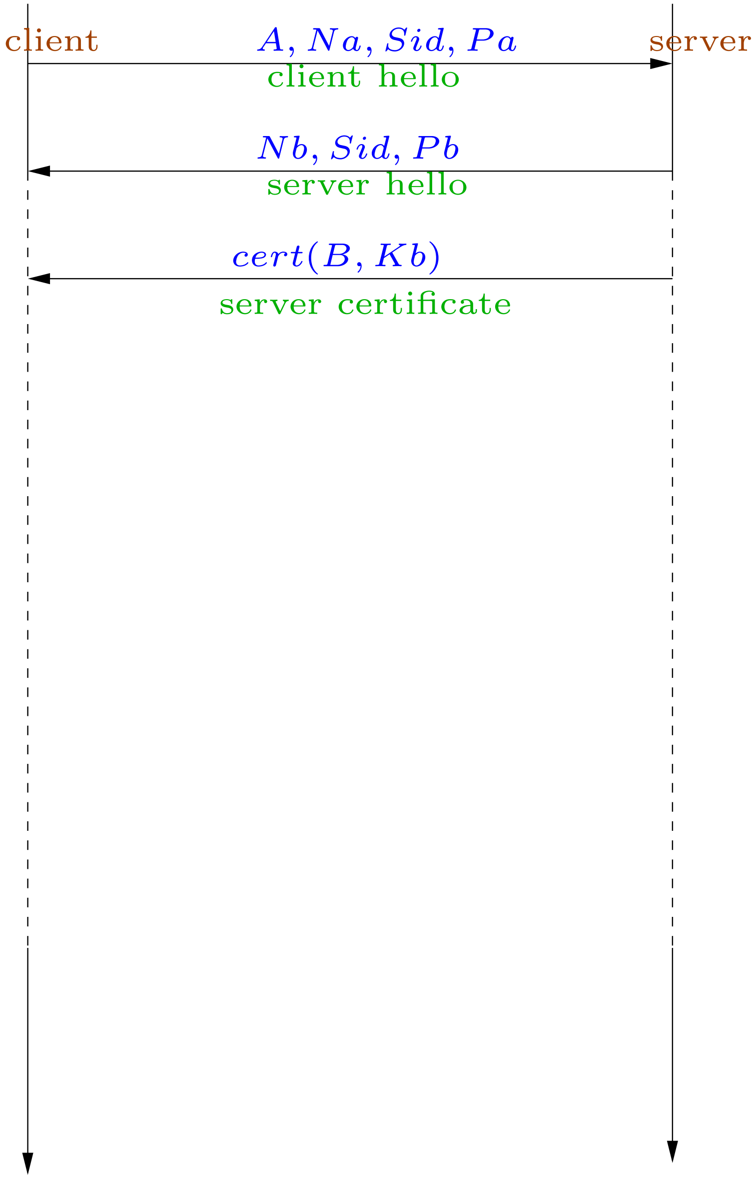


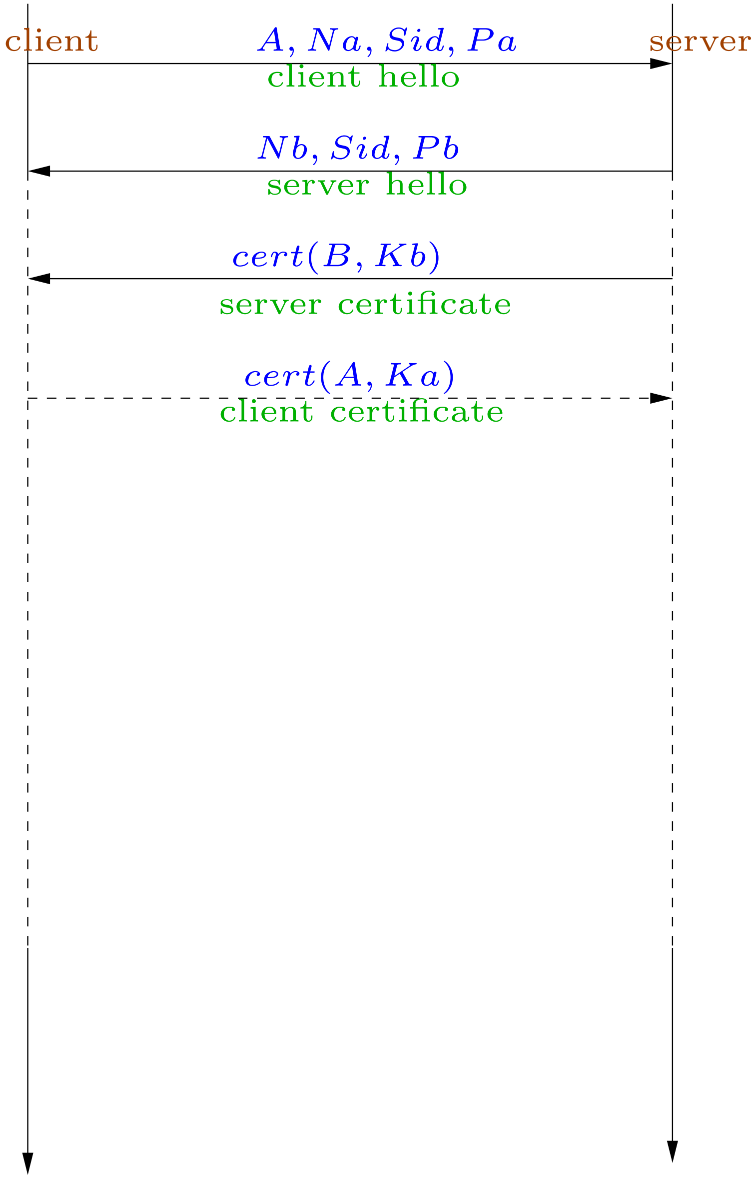
The Handshake protocol

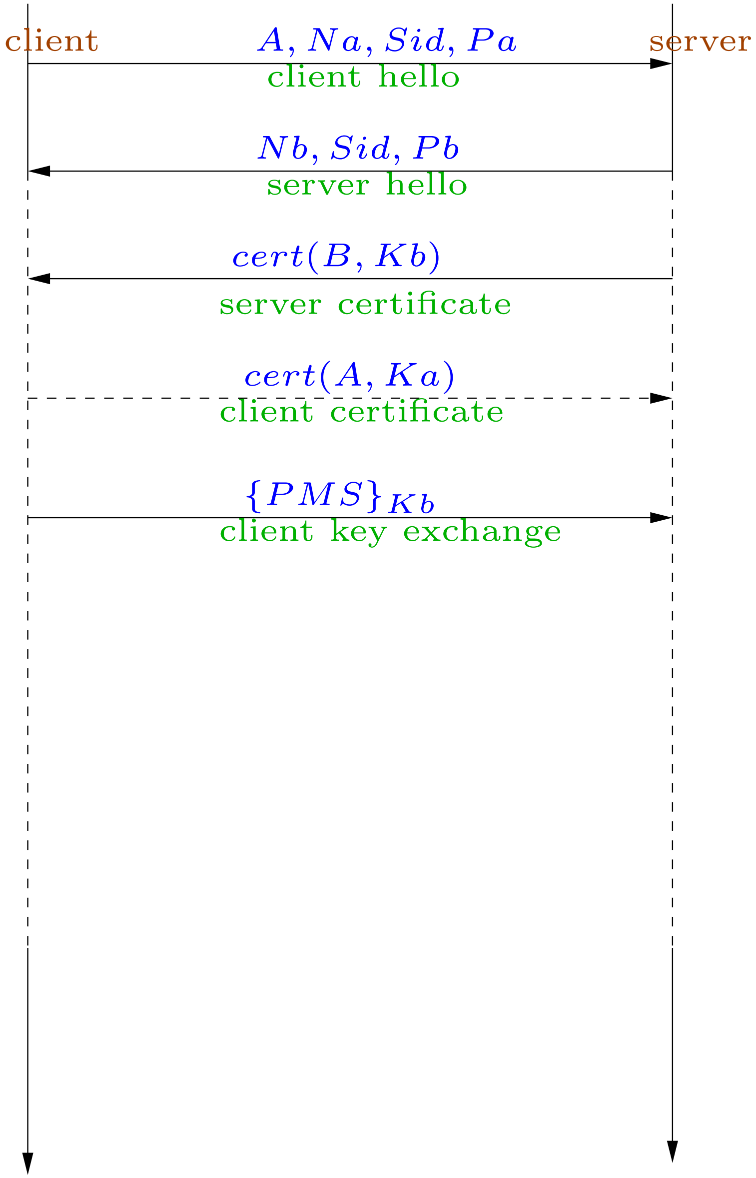
Sid is the session identifier.
Pa contains the client's preferences for encryption method, etc.



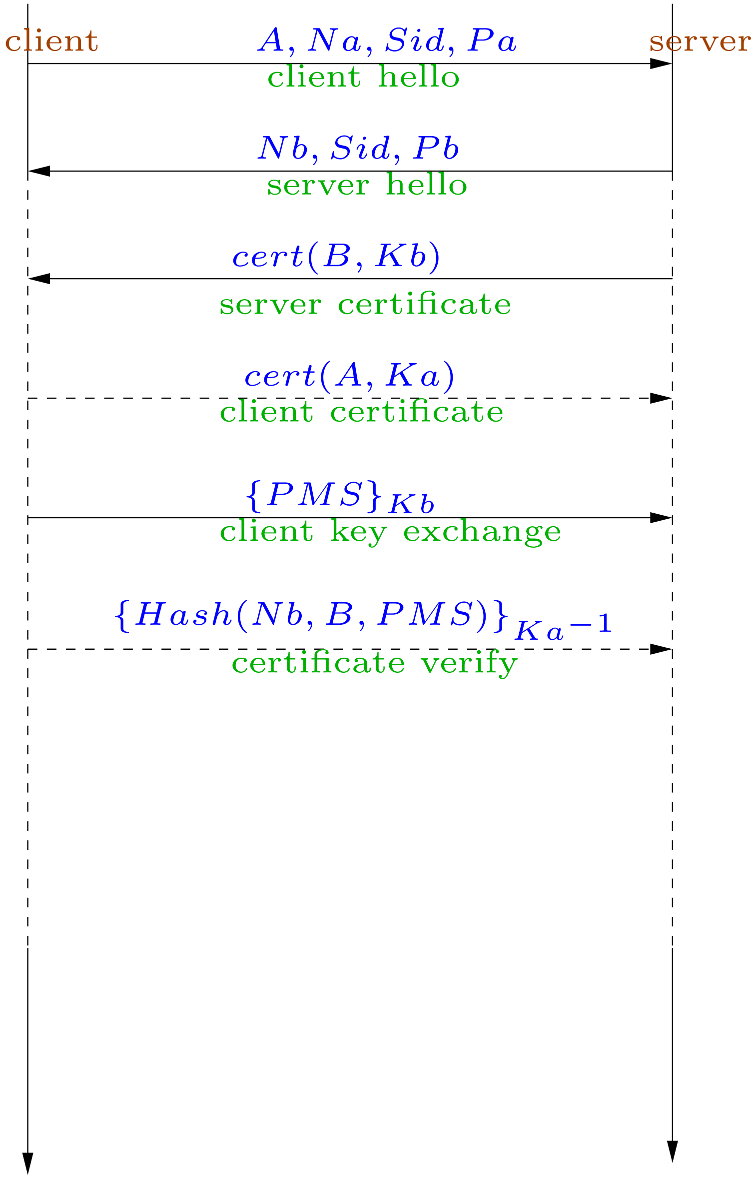
Pb contains the server's preferences.


client
server
A, Na, Sid, Pa
client hello
Nb, Sid, Pb
server hello
$cert(B, Kb)$
server certificate

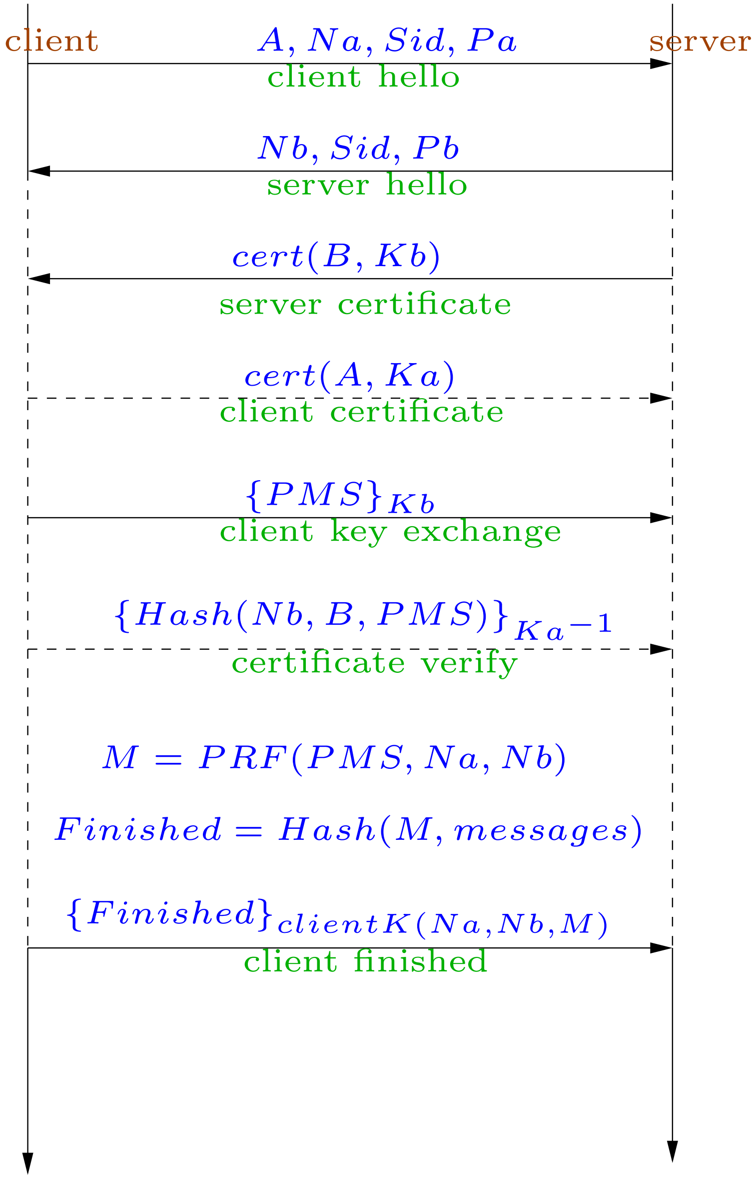

client
A, Na, Sid, Pa
server
client hello
Nb, Sid, Pb
server hello
$cert(B, Kb)$
server certificate
$cert(A, Ka)$
client certificate



PMS, the Pre-Master-Secret, is a nonce.



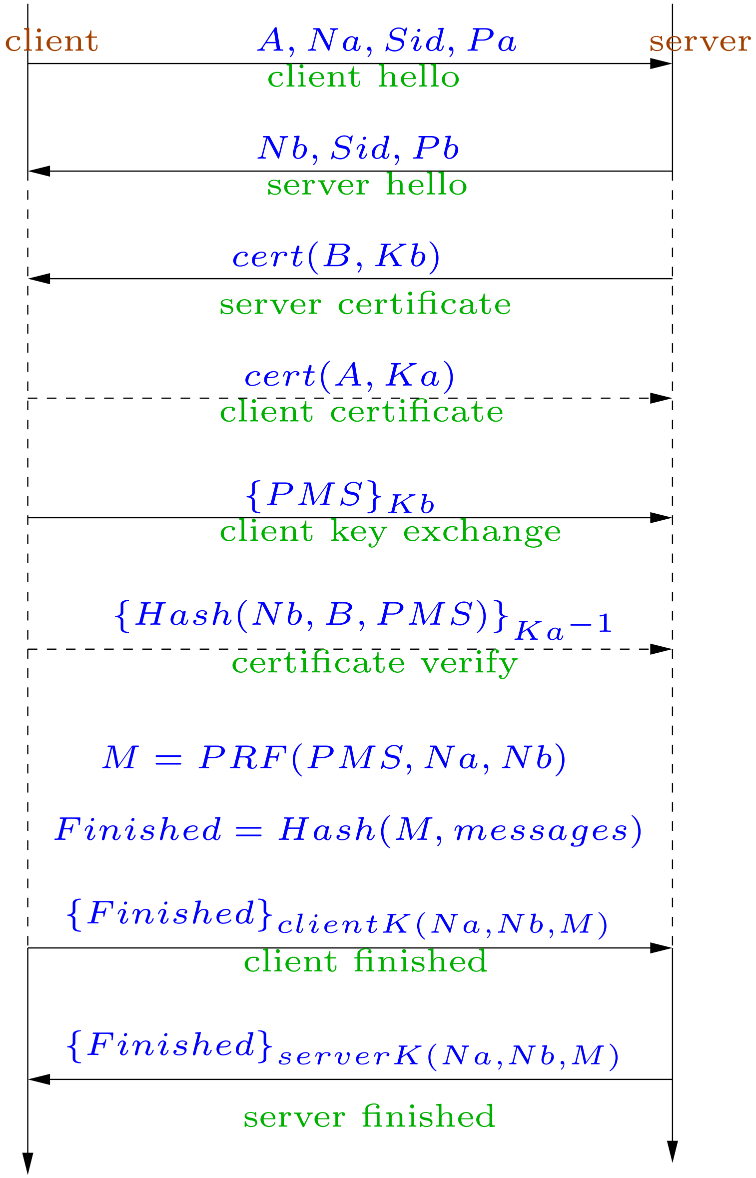
client
server
A, Na, Sid, Pa
client hello
Nb, Sid, Pb
server hello
$cert(B, Kb)$
server certificate
$cert(A, Ka)$
client certificate
$\{PMS\}_{Kb}$
client key exchange
$\{Hash(Nb, B, PMS)\}_{Ka-1}$
certificate verify



M is the Master-Secret.

PRF: Pseudo-Random Function

$messages$ is the set of all messages exchanged so far.



Each side should receive the finish message before continuing, to prevent the ciphersuite rollback attack of SSL Version 3.0:

the encryption preferences Pa and Pb can be changed be the intruder to request weak encryption.