Station-to-Station Protocol

$$
\begin{array}{ccc}
U & & V \\
b_U = \alpha^{a_U} & \xrightarrow{\quad b_U \quad} & \\
 & \xleftarrow{\quad \mathbf{Cert}(V), b_V, y_V \quad} & b_V = \alpha^{a_V} \\
 & & y_V = sig_V(b_V \parallel b_U) \\
y_U = sig_U(b_U \parallel b_V) & \xrightarrow{\quad \mathbf{Cert}(U), y_U \quad} &
\end{array}
$$

Security Properties of STS

- the scheme is secure against known session key attacks and provides perfect forward secrecy
- the scheme is a secure mutual identification scheme (i.e., if the adversary is active during a given flow of the protocol, then no honest paritcipant will "accept" after that time)
- in addition, the scheme is a secure KAS wrt a passive adversary (i.e., U and V can both compute the same session key, K, and the adversary cannot compute any information about K)
- if U "accepts", it means that she believes that
  - she has been communicating with V
  - U and V can compute the same session key, and
  - no one other than V can compute any information about the session key

Key Authentication and Key Confirmation

Suppose U and V are honest, and they execute an SKDS or KAS. At the end of the session, U and V should each be able to compute the same session key, K, whose value should be unknown to the adversary. Suppose that U "accepts". The following properties discuss various types of assurance that may be provided to U:

implicit key authentication

U is assured that no one other than V can compute K

implicit key confirmation

U is assured that V can compute K, and no one other than V can compute K

explicit key confirmation

U is assured that V has computed K, and no one other than V can compute K

Key Authentication/Confirmation in Specific Protocols

- the *Needham-Schroeder* and *Kerberos* SKDS provide explicit key confirmation
- the *Station-to-Station* KAS provides implicit key confirmation
- the *Bellare-Rogaway* SKDS provides implicit key authentication
- it is now generally accepted that explicit key confirmation is not an important requirement of an SKDS or KAS
- if desired, a protocol to ensure explicit key confirmation can be carried out when the session key is going to be used for its intended purpose

Identification and Key Authentication/Confirmation

- implicit or explicit key confirmation is not possible without simultaneous mutual identification (entity authentication)
- implicit key authentication is possible without identification
- of course, mutual identification can be done in the absence of any key distribution or key agreement
- a KAS (or SKDS) with key confirmation is “secure” if it is a secure mutual identification scheme, and the specified type of key confirmation is acheived

Matsumoto-Takashima-Imai Protocols

- the MTI protocols are public-key KAS that use LL-keys (authenticated via certificates)
- the algebraic setting is the same as for Diffie-Hellman protocols, i.e., we work in a subgroup of ${\mathbb{Z}_p}^*$ of prime order q, in which the DDH is believed to be intractible
- the protocols are two-pass key agreement protocols without identification, in which signatures are not used
- every user U has a private LL-key a_U and a corresponding public key

$$b_U = \alpha^{a_U} \bmod p$$

- the users' public keys are signed by the TA and stored on certificates, as usual

Protocol: MTI/A0 KEY AGREEMENT

1. U chooses r_U at random, where $0 \leq r_U \leq q-1$. Then U computes
$$s_U = \alpha^{r_U} \bmod p$$
and sends **Cert**(U) and s_U to V.

2. V chooses r_V at random, where $0 \leq r_V \leq q-1$. Then V computes
$$s_V = \alpha^{r_V} \bmod p$$
and sends **Cert**(V) and s_V to U.

3. U computes
$$K = (b_V)^{r_U}(s_V)^{a_U} \bmod p$$
and V computes
$$K = (b_U)^{r_V}(s_U)^{a_V} \bmod p.$$

Analysis

- the session key is

$$K = \alpha^{r_U a_V + r_V a_U} \bmod p$$

- the *MTI/A0* protocol attempts to provide implicit key authentication without identification (similar to the *Bellare-Rogaway SKDS*)
- the protocol is secure against passive attacks (assuming the DDH is intractible)
- the protocol is apparently secure against intruder-in-the-middle attacks
- however, the *MTI/A0* protocol is vulnerable to a certain kind of parallel session known session key attack, as well as the more esoteric Burmester "triangle attack", which we describe now

A Parallel Session Attack

- W pretends to be V in a session $\mathcal{S}_1$ with U; and W pretends to be U in a session $\mathcal{S}_2$ with V:

$$V \quad \boxed{\begin{array}{c} \xrightarrow{\quad\mathbf{Cert}(V), s_V\quad} \\[1em] \xleftarrow{\quad\mathbf{Cert}(U), s_U\quad} \end{array}} \quad W \quad \boxed{\begin{array}{c} \xleftarrow{\quad\mathbf{Cert}(U), s_U\quad} \\[1em] \xrightarrow{\quad\mathbf{Cert}(V), s_V\quad} \end{array}} \quad U$$

- then W requests the key K for session $\mathcal{S}_2$, which is allowed in a known session key attack
- K is also the key for session $\mathcal{S}_1$

The Triangle Attack

V		W		U
		$\xleftarrow{\mathbf{Cert}(U),\, s_U}$		
		$\xrightarrow{\mathbf{Cert}(V),\, s_V}$		
	$\xleftarrow{\mathbf{Cert}(W),\, s_U}$			
	$\xrightarrow{\mathbf{Cert}(V),\, s'_V}$			
			$\xrightarrow{\mathbf{Cert}(W),\, s_V}$	
			$\xleftarrow{\mathbf{Cert}(U),\, s'_U}$	

Analysis

- the session keys for the three sessions are as follows:

$$
\begin{aligned}
K_1 &= \alpha^{r_U a_V + r_V a_U} \\
K_2 &= \alpha^{r_U a_V + r'_V a_W} \\
K_3 &= \alpha^{r'_U a_W + r_V a_U}
\end{aligned}
$$

- therefore

$$
K_1 = \frac{K_2 K_3}{(s'_V s'_U)^{a_W}}
$$

- suppose that W requests K_2 and K_3 (in a known session key attack)
- then W can compute K_1

A Secure (?) Variation

- the parallel session attack can be carried out because the key is a symmetric function of the inputs provided by the two parties (i.e., $K((r_U, a_U), (r_V, a_V)) = K((r_V, a_V), (r_U, a_U)))$
- to eliminate the attack, we should destroy this symmetry property
- the triangle attack can be eliminated through the use of a *key derivation function*, which is a "random" public function used to derive the session key from the values $\alpha^{r_U a_V}$ and $\alpha^{r_V a_U}$
- suppose that the key derivation function is defined as

$$K = h_{RO}(\alpha^{r_U a_V} \parallel \alpha^{r_V a_U}),$$

  where h_{RO} is a truly random function (i.e., a *random oracle*)
- the resulting scheme is conjectured to be secure by Blake-Wilson, Johnson and Menezes

Further Settings For Two-Party Key Establishment Protocols

- we have considered two-party key agreement schemes in the public-key setting
- it is also possible to study *key transport schemes*, where one party chooses the session key and "transports" it to the other party (i.e., by encrypting it)
- KAS and KTS can also be studied in the secret-key setting, where signature schemes are replaced by MACs, and public-key encryption is replaced by secret-key encryption
- in the public-key setting, identity based schemes have been devised, in which public keys are derived from users' identities; this eliminates the need for certificates to authenticate public keys