9. IDENTIFICATION, AUTHENTICATION

and E-COMMERCE

Most of today's applications of cryptography ask for authentic data rather than for secret data. The main problem is how to protect data and communications against an active attack.

Main problems to deal with:

1. User identification: How can a person prove his (her) identity?

2. Message authenticity: Are the ways to decide, for the recipient, whether the message obtained is from the person who is supposed to send it?

3. Message integrity: Can tools be provided to decide for the recipient whether or not the message was changed?

One particular objective is to construct an identification scheme that is simple enough so it can be implemented on smart cards — smart cards are essentially credit cars equipped with a chip that can perform arithmetical operations and communications.

4. E-commerce: One of the main new application of the cryptographic techniques for identification and authentication is to establish secure and convenient manipulation with digital money, especially for e-commerce.

Identification using a PKC system

- Alice chooses a random r and sends $e_B(r)$ to a person.
- Alice identifies the person as Bob if she gets back r.

A misuse of the system

Observe that a (non-honest) Alice could misuse such a system.

Indeed, she could intercept a communication of a Jane (a new "player") with Bob, to get a cryptotext $e_B(w)$ Jana sends to Bob, and then sends it to Bob.

Honest Bob, following protocol, would then return w to Alice and she would get the plaintext w.

ELEMENTARY PROTOCOLS

USER IDENTIFICATION

Static means of identification: People can be identified by their attributes (say fingerprints), possessions (say passports), knowledge.

Dynamic means of identification: Challenge and respond protocols.

Let both Alice and Bob share a key k and a one-way function f_k.

1. Bob sends Alice a random number, or a string, $RAND$.
2. Alice sends Bob $PI = f_k(RAND)$.
3. If Bob gets PI, then he verifies whether $PI = f_k(RAND)$.

If yes, he starts to believe that the person he communicated with is Alice.

The process can be repeated to increase the probability of making correct guess.

Message authentication

MAC – method (Message Authentication Code)
Alice and Bob share a key k and a coding algorithm A.

1. With a message m, Alice sends $(m, \overbrace{A_k(m)}^{\text{MAC}})$
2. If Bob gets $(m'$, MAC), then he computes $A_k(m')$ and compares it with MAC.

MAC in practice

Let n be a fixed message length.

Let f map a key and a message of the length n into a message of length n

1. Message m is divided into blocks $m_1, m_2, \dots$ of length n
2. Computation is done

$$c_1 = f_k(m_1)$$
$$c_i = f_k(c_{i-1} \oplus m_i), i = 2, 3, \dots$$

MAC computed this way has fixed length and depends on the whole message.

Disadvantage of static identification schemes

Everybody who knows your password or PIN can impersonate you.

Zero-knowledge identification schemes

Using so called zero-knowledge identification schemes you can identify yourself without giving to the authenticator the ability to impersonate you.

Simplified Fiat-Shamir identification scheme

A trusted authority (TA) chooses large random primes $p, q \quad n = pq$, a quadratic residue $v \in QR_n$, and an s such that $s^2 \equiv v (\bmod n)$.

public-key: v
private-key: s (that Alice knows)

Identification protocol

(1) Alice chooses a random $r < n$, computes $a = r^2 \bmod n$ and sends x to Bob.
(2) Bob sends to Alice a random bit b.
(3) Alice sends Bob $y = rs^b$.
(4) Bob verifies if and only if $y^2 = av^b \bmod n$, proving that Alice knows a square root of a.

This protocol is a so-called single accreditation

Alice proves her identity by convincing Bob that she knows square root of s (without revealing s to Bob).

If protocol is repeated t times, Alice has a chance 2^{-t} to fool Bob if she does not known s.

Analysis of Fiat-Shamir identification

public-key: v
private-key: s (of Alice) such that $s^2 = v$.

Protocol

(1) Alice chooses a random $r < n$, computes $a = r^2 \bmod n$ and sends x to Bob.
(2) Bob sends to Alice a random bit b.
(3) Alice sends Bob $y = rs^b$.
(4) Bob verifies if and only if $y^2 = av^b \bmod n$, proving that Alice knows a square root of a.

Analysis

1. The first message is a commitment by Alice that she knows a square root of a.
2. The second message is a challenge by Bob.
   - If Bob sends $b = 0$, then Alice has to open her commitment and to reveal r.
   - If Bob sends $b = 1$, then Alice has to show her secret s in an "encrypted form".
3. The third message is Alice's response to Bob's challenge.

Completeness If Alice knows s, and both Alice and Bob follow the protocol, then the response rs^b is a square root of av^b.

Observe that Eve can cheat with probability of success $\frac{1}{2}$ as follows:

- Eve chooses random $r \in \mathbf{Z}_n^*$, random $b_1 \in \{0, 1\}$ and sends $a = r^2 v^{-b_1}$, to Bob.
- Bob chooses $b \in \{0, 1\}$ at random and sends it to Alice.
- Alice sends r to Bob.

Fiat-Shamir identification scheme parallel version

In the following version of the Fiat-Shamir idenitification scheme the probability of a false identification is decreased.

Choose an integer k, primes p, q, and compute $n = pq$.

Choose (quadratic residues) $v_1, \ldots, v_k \in QR_n$.

Compute $s_1, \ldots, s_k$ such that $s_i = \sqrt{v_i} \bmod n$

public-key: $v_1, \ldots, v_k$
secret-key: $s_1, \ldots, s_k$ (of Alice)

(1) Alice chooses a random $r < n$, computes $a = r^2 \bmod n$ and sends a to Bob.
(2) Bob sends to Alice a random k-bit string $b_1 \ldots b_k$.
(3) Alice sends to Bob

$$y = r \prod_{i=1}^{k} s_i^{b_i} \bmod n$$

. (4) Bob accepts if and only if

$$y^2 = a \prod_{i=1}^{k} v_i^{b_i} \bmod n$$

.

Alice and Bob repeat the above protocol several (t) times, until Bob is convinced that Alice knows $s_1, \ldots, s_k$.

The chance that Alice fools Bob is 2^{-kt}, a decrease comparing with the chance $\frac{1}{2}$ of the previous version of the identification scheme.

The Schnorr identification scheme — setting

A practically attractive, computationally efficient (in time, space + comm.) scheme which minimizes storage + computations performed by Alice (a smart card).

Scheme requires also a trusteed authority (TA) that

(1) chooses: a large prime $p (\geq 2^{512})$,
a large prime q dividing $p-1$ and $q \leq 2^{140}$,
an $\alpha \in \mathbb{Z}_p^\star$ of order q,
a security parameter t such that $2^t < q$
(a secure hash function (used with DSS), and p, q, α, t are public).

(2) establishes: a secure digital signature scheme with a secret signing algorithm sig_{TA} and a public verification algorithm ver_{TA}.

Protocol for issuing a certificate to Alice

1. TA establishes Alice's identity by conventional means and forms a string ID(Alice) which contains identification information.

2. Alice chooses a secret random $0 \leq a \leq q-1$ and computes

$$v = \alpha^{-a} \bmod p$$

and sends v to the TA.

3. TA generates the signature

$$s = sig_{TA}(ID(Alice), v)$$

and sends to Alice the certificate

$$C(Alice) = (ID(Alice), v, s)$$

Schnorr identification scheme

1. Alice chooses a random $0 \leq k < q$ and computes
$$\gamma = \alpha^k \bmod p.$$
2. Alice sends her certificate $C(Alice) = (ID(Alice), v, s)$ and γ to Bob.

3. Bob verifies the signature of TA by checking that
$$ver_{TA}(ID(Alice), v, s) = true.$$

4. Bob chooses a random $1 \leq r \leq 2^t$, where $t < \lg q$ is a security parameter and sends r to Alice (often $t \leq 40$ is chosen).

5.Alice computes and sends to Bob
$$y = (k + ar) \bmod q$$
.

6. Bob verifies that
$$\gamma \equiv \alpha^y v^r (\bmod p)$$

This way Alice shows its identity to Bob. Indeed,

$$\begin{aligned} \alpha^y v^r &\equiv \alpha^{k+ar} \alpha^{-ar} (\bmod p) \\ &\equiv \alpha^k (\bmod p) \\ &\equiv \gamma (\bmod p). \end{aligned}$$

Total storage: 512 bits for ID(Alice), 512 bits for v, 320 bits for s (if DSS is used), total – 1344 bits.

Total communication: Alice $\rightarrow$ Bob 1996 bits,
Bob $\rightarrow$ Alice 40 Bits.

Okamoto identification scheme

The disadvantage of Schnorr identification scheme is that there is no proof of its security. A proof of security exists for a modification of Schnorr identification scheme presented below.

To set up an dentification scheme a TA chooses:

- a large prime $p \leq 2^{512}$,
- a large prime $q \geq 2^{140}$ dividing $p-1$;
- two elements $\alpha_1, \alpha_2 \in \mathbf{Z}_p^*$ of order q.

TA makes public p, q, α_1, α_2 and keeps secret (also before Alice and Bob)

$$c = \lg_{\alpha_1} \alpha_2.$$

Finally, TA chooses a signature scheme and a hash function.

Issuing a certificate to Alice

- TA establishes Alice's identity and issues an identification string ID(Alice).
- Alice secretly and randomly chooses $0 \leq a_1, a_2 \leq q-1$ and sends to TA
$$v = \alpha_1^{-a_1} \alpha_2^{-a_2} \bmod p.$$
- TA generates a signature $s = sig_{TA}(\mathrm{ID(Alice)}, v)$ and sends to Alice the certificate
$$\mathrm{C(Alice)} = (\mathrm{ID(Alice)}, v, s).$$

Okamoto identification scheme - continuation

Basic setting TA chooses: a large prime $p \leq 2^{512}$, large prime $q \geq 2^{140}$ dividing $p-1$; two elements $\alpha_1, \alpha_2 \in \mathbf{Z}_p^*$ of order q. TA keep secret (also from Alice and Bob) $c = \lg_{\alpha_1} \alpha_2$.

Issuing a certificate to Alice

- TA establishes Alice's identity and issues an identification string ID(Alice).
- Alice randomly chooses $0 \leq a_1, a_2 \leq q-1$ and sends to TA.

$$v = \alpha_1^{-a_1} \alpha_2^{-a_2} \bmod p.$$

- TA generates a signature $s = sig_{TA}(\text{ID(Alice)}, v)$ and sends to Alice the certificate

$$\text{C(Alice)} = (\text{ID(Alice)}, v, s).$$

Okamoto identification scheme.

- Alice chooses random $0 \leq k_1, k_2 \leq q-1$ and computes

$$\gamma = \alpha_1^{k_1} \alpha_2^{k_2} \bmod p.$$

- Alice sends to Bob her certificate $(\text{ID(Alice)}, v, s)$ and γ.
- Bob verifies the signature of TA by checking that

$$ver(\text{ID(Alice)}, v, s) = \text{true}.$$

- Bob chooses a random $1 \leq r \leq 2^t$ and sends it to Alice.
- Alice sends to Bob

$$y_1 = (k_1 + a_1 r) \bmod q; \qquad y_2 = (k_2 + a_2 r) \bmod q.$$

- Bob verifies

$$\gamma \equiv \alpha_1^{y_1} \alpha_2^{y_2} v^r \pmod{p}.$$

Authentication codes

Authentication codes denote methods of ensuring integrity of messages — that a message has not been tampered and that originated with the presumed transmitter.

The goal is to achieve authentication, with sufficiently large probability, even in the presence of Mallot, a man in the middle, who can observe transmitted messages and can also introduce into the channel messages of his own choosing.

Formally, an authentication code consists:

- A set $\mathcal{M}$ of possible messages.
- A set $\mathcal{T}$ of possible authentication tags.
- A set $\mathcal{K}$ of possible keys.
- For each $k \in \mathcal{K}$, an authentication rule $a_k : \mathcal{M} \to \mathcal{T}$.

Transmission process

- Alice and Bob jointly choose a secret key k.
- If Alice wants to send a message w to Bob, she sends (w, t), where $t = a_k(w)$.
- If Bob receives (w, t) he computes $t' = a_k(w)$ and if $t = t'$ Bob accepts the message as authentic.

Attacks and deception probabilities

There are two basic types of attacks Mallot, the man in the middle, can do.

Impersonation. Mallot introduces a message (w, t) into the channel expecting that message will be received by Bob as message sent by Alice.

Substitution. Mallot replaces a message (w, t) in the channel by a new one, (w', t'), expecting that message will be accepted by Bob as message sent by Alice.

With any impersonation (substitution) attack let P_i (P_s) denote probability that Mallot deceives Bob, if Mallot follows an optimal strategy.

In order to determine such probabilities we need to know probability distributions p_m on messages and p_k on keys.

The $|\mathcal{K}| \times |\mathcal{M}|$ **authentication matrix** tabulates all authenticated tags: the item in a row corresponding to a key k and a message w contains authentication tag $a_k(w)$.

The goal of authentication codes is to decrease probabilities that Mallot performs successfully impersonation or substituition.

Example

Let $\mathcal{M} = \mathcal{T} = \mathbf{Z}_3$, $\mathcal{K} = \mathbf{Z}_3 \times \mathbf{Z}_3$.

For $(i, j) \in \mathcal{K}$ and $w \in \mathcal{M}$, let

$$a_{ij}(w) = (iw + j) \bmod 3.$$

The matrix of the authentication tags has the form

key	0	1	2
(0,0)	0	0	0
(0,1)	1	1	1
(0,2)	2	2	2
(1,0)	0	1	2
(1,1)	1	2	0
(1,2)	2	0	1
(2,0)	0	2	1
(2,1)	1	0	2
(2,2)	2	1	0

Impersonation attack: Mallot picks a message w and attempts to guess the correct authentication tag.

However, for each message w and each a there are exactly three keys k such that $e_k(w) = a$. Hence $P_i = \frac{1}{3}$.

Substitution attack: By checking the table one can see that if Mallot observes an authenticated message (w, t), then there are only three possibilities for the key used.

Moreover, for each choice (w', t'), $w \neq w'$, there is exactly one of the three possible keys for (w', t') that could be used. Therefore, $P_s = \frac{1}{3}$.

Computation of deception probabilities I

Probability of impersonation: For $w \in \mathcal{M}, t \in \mathcal{T}$, let us define *payoff*$(w,t)$ to be the probability that Bob accepts the message (w,t) as authentic. Then

$$\textit{payoff}(w,t) = Pr(t = a_{k_0}(w)) \tag{4}$$

$$= \sum_{\{k \in \mathcal{K} \mid a_k(w)=t\}} Pr_{\mathcal{K}}(k) \tag{5}$$

In other words, *payoff*(w,t) is computed by selecting those rows of the authentication matrix that have entry t in the column w and then by summing up probabilities of the corresponding keys.

Therefore

$$P_i = \max\{\textit{payoff}(w,t) \mid w \in \mathcal{M}, t \in \mathcal{A}\}.$$

Probability of substitution: Define, for $w, w' \in \mathcal{M}, w \neq w'$ and $t, t' \in \mathcal{A}$, *payoff*(w',t',w,t) to be the probability that a substitution of (w,t) with (w',t') succeeds to deceive Bob. Hence

$$\textit{payoff}(w',t',w,t) = Pr(t' = a_{k_0}(w') \mid t = a_{k_0}(w)) \tag{6}$$

$$= \frac{Pr(t' = a_{k_0}(w') \cap t = a_{k_0}(w))}{Pr(t = a_{k_0}(w))} \tag{7}$$

$$= \frac{\sum_{\{k \in \mathcal{K} \mid a_k(s)=t, a_k(w')=t'\}} p_{\mathcal{K}}(k)}{\textit{payoff}(w,t)} \tag{8}$$

Observe that the numerator in the last fraction is found by selecting rows of the authentication matrix with value t in column w and t' in column w'.

Computation of deception probabilities II

Since Mallot wants to maximize his chance of deceiving Bob, he needs to compute

$$p_{w,t} = \max\{\textit{payoff}(w', t', w, t) \mid w' \in \mathcal{M}, w \neq w', t' \in \mathcal{A}\}.$$

$p_{w,t}$ therefore denotes the probability that Mallot can deceive Bob with a substitution in the case (w, t) is the message observed.

If $Pr_{\mathcal{M}a}(w, t)$ is the probability of observing a message (w, t) in the channel, then

$$P_s = \sum_{(w,t) \in \mathcal{M}a} Pr_{\mathcal{M}a}(w, t) p_{w,t}$$

and

$$Pr_{\mathcal{M}a}(w, t) = Pr_{\mathcal{M}}(w) Pr_{\mathcal{K}}(t|w) = Pr_{\mathcal{M}}(w) \times \textit{payoff}(w, t).$$

The next problem is to show how to construct authentication code that the deception probabilities are as low as possible.

The concept of **orthogonal arrays** introduced next serves well such a purpose.

Orthogonal arrays

Definition An orthogonal array $OA(n, k, \lambda)$ is a $\lambda n^2 \times k$ array of n symbols, such that in any two columns of the array every one of the possible n^2 pairs of symbols occurs in exactly λ rows.

Example $OA(3, 3, 1)$ obtained from the authentication matrix presented before:

$$\begin{pmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 2 & 2 & 2 \\ 0 & 1 & 2 \\ 1 & 2 & 0 \\ 2 & 0 & 1 \\ 0 & 2 & 1 \\ 1 & 0 & 2 \\ 2 & 1 & 0 \end{pmatrix}.$$

Theorem Suppose we have an orthogonal array $OA(n, k, \lambda)$. Then there is an authentication code with $|\mathcal{M}| = k$, $|\mathcal{A}| = n$, $|\mathcal{K}| = \lambda n^2$ and $P_i = P_s = \frac{1}{n}$.

Proof Use each row of the orthogonal array as an authentication rule (key) with equal probability. Therefore we have the following correspondence:

orthogonal array	authentication code
row	authentication code
column	message
symbol	authentication tag

Construction and bounds for OAs

In an orthogonal array $OA(n, k, \lambda)$

- n determines the number of authenticators (security of the code);
- k is the number of messages the code can accommodate;
- λ relates to the number of keys — λn^2.

The following holds for orthogonal arrays.

- If p is prime, then $OA(p, p, 1)$ exits.
- Suppose there exists an $OA(n, k, \lambda)$. Then
$$\lambda \geq \frac{k(n-1)+1}{n^2};$$
- Suppose that p is a prime and $d \leq 2$ an integer. Then there is an orthogonal array $OA(p, (p^d - 1)/(p-1), p^{d-2})$.
- Let us have an authentication code with $|\mathcal{A}| = n$ and $P_i = P_s = \frac{1}{n}$.Then $|\mathcal{K}| \geq n^2$. Moreover, $|\mathcal{K}| = n^2$ if and only if there is an orthogonal array $OA(n, k, 1)$, where $|\mathcal{M}| = k$ and $P_{\mathcal{K}}(k) = \frac{1}{n^2}$ for every key $k \in \mathcal{K}$.

The last claim shows that there are no much better approaches to authentication codes with deception probabilities as small as possible than orthogonal arrays.

Shamir's threshold secret sharing scheme

Secret sharing schemes distribute a "secret" among several users in such a way that only predefined sets of users can "assemble" the secret.

For example, a vault in the bank can be opened only if at least two out of three responsible employees use their knowledge and tools to open the vault.

An important special simple case of secret sharing schemes are **threshold secret sharing schemes** at which a certain threshold of participant is needed and sufficient to assemble the secret.

Definition Let $t \leq n$ be positive integers. A (n, t)-threshold scheme is a method of sharing a secret S among a set P of n participants ($\mathcal{P} = \{P_i \,|\, 1 \leq i \leq n\}$), in such a way that any t, or more, participants can compute the value S, but no group of $t-1$, or less, participants can compute S.

Secret S is chosen by a "dealer" $D \notin \mathcal{P}$.

It is assumed that the dealer "distributes" the secret to participants secretly and in such a way that no participant knows shares of other participants.

Shamir's (n, t)-threshold scheme

Initiation phase:
Dealer D chooses a prime p, n distinct $x_i, 1 \leq i \leq n$, and D gives the values x_i to the user P_i.

The values x_i are public.

Share distribution: Suppose D wants to share a secret $S \in \mathbf{Z}_p$ among the users. D randomly chooses $t-1$ elements of $\mathbf{Z}_p$, $a_1, \ldots, a_{t-1}$.

D first computes "shares" $y_i = a(x_i)$, $1 \leq i \leq n$,

where

$$a(x) = S + \sum_{j=1}^{t-1} a_j x^j \bmod p.$$

D then sends the share y_i, $1 \leq i \leq n$ to the participant P_i.

Secret cumulation: Let participants $P_{i_1}, \ldots, P_{i_t}$ wants to determine the secret S. Since $a(x)$ has degree $t-1$, $a(x)$ has the form

$$a(x) = a_0 + a_1 x + \ldots + a_{t-1} x^{t-1},$$

and the coeficients a_i can be obtained from t equations $a(x_{i_j}) = y_{i_j}$, where all arithmetic is done modulo p.

It can be easily show that equations obtained this way are linearly independent and the system has a unique solution.

In such a case

$$S = a_0.$$

Shamir's scheme - technicalities

Shamir's scheme uses the following result concerning polynomials over fields $\mathbf{Z}_p$, where p is prime.

Theorem. Let $f(x) = \Sigma_{i=0}^{t-1} a_i X^i \in \mathbf{Z}_p[X]$ be a polynomial of degree $t-1$ and let $P = \{(x_i, f(x_i)) \,|\, x_i \in \mathbf{Z}_p, i = 1, \ldots, t, x_i \neq x_j, i \neq j\}$. For $Q \subseteq P$, let $\mathcal{P}_Q = \{g \in \mathbf{Z}_p[X] \,|\, \deg(g) = t-1, g(x) = y \text{ for all } (x, y) \in Q\}$.

- $\mathcal{P}_P = \{f(X)\}$, i.e. f is the only polynomial of degree $t-1$, whose graph contains all t points in P.
- If $Q \subset \mathcal{P}$ is a proper subset and $x \neq 0$ for all $(x, y) \in Q$, then each $a \in \mathbf{Z}_p$ appears with the same frequency as the constant coefficient of a polynomial in $\mathcal{P}_Q$.

Corollary (Lagrange formula) Let $f(X) = \Sigma_{i=0}^{t-1} a_i X^i \in \mathbf{Z}_p[X]$ be a polynomial and let $P = \{(x_I, f(x_i)) \,|\, i = 1, \ldots, t, x_i \neq x_j, i \neq j\}$. Then

$$f(X) = \sum_{i=1}^{t} f(x_i) \prod_{1 \leq j \leq t, j \neq i} \frac{X - x_j}{x_i - x_j}.$$

Shamir's (n, t)-threshold scheme

To distributes n shares of a secret S among users $P_1, \ldots, P_n$ a trusted centre T proceeds as follows:

- T chooses a prime $p > \max\{s, n\}$ and sets $a_0 = S$.
- T selects randomly $a_1, \ldots, a_{t-1} \in \mathbf{Z}_p$ and creates polynomial $f(X) = \Sigma_{i=0}^{t-1} a_i X^i$.
- T computes $s_i = f(i)$, $i = 1, \ldots, n$ and transfers (i, s_i) to the user P_i in a secure way.

Any group J of t or more users can compute the secret. Indeed, from the previous collolary we have

$$S = a_0 = f(0) = \sum_{i \in J} f(i) \prod_{j \in J, j \neq i} \frac{j}{j - i}.$$

In case $|J| < t$, then each $a \in \mathbf{Z}_p$ is likely to be the secret.

E-COMMERCE

Very important is to ensure security of e-money transactions needed for e-commerce.

In addition to providing security and privacy, its goal is to prevent alterations of purchase orders and forgery of credit card information.

Basic requirements for e-commerce system:

Authenticity: Participants in transactions cannot be impersonated and signatures cannot be forged.

Integrity: Documents (purchase orders, payment instructions,...) cannot be forged.

Privacy: Details of transaction should be kept secret.

Security: Sensitive information (as credit card numbers) must be protected.

Anonimity: Anonimity of any money sender should be garanteed.

Additional requirement: In order to allow an efficient fighting of the organized crime a system for processing e-money has to be such that, under well defined conditions, it has to be possible to revoke customer's identity and flow of e-money – to have a fair payment system.

DUAL SIGNATURE PROTOCOL

We describe protocol to solve the problem that at shopping bank should not know what cardholder has ordered, and no shop could learn card numbers.

Participants of the protocol: bank, cardholder, shop

The cardholder uses the following information:

- GSO – Goods and/or Service Order (cardholder's name, shop's name, items ordered, their quantity,...).
- PI – Payment instructions (shop's name, card number, total price,...)

System uses a public hash function h.

RSA is used and

- e_C, e_S and e_B are public keys of cardholder, shop, bank and
- d_C, d_S and d_B are their secret keys.

CARDHOLDER and SHOP ACTIONS

A cardholder performs the following procedure:

1. Computes HEGSO $= h(e_S(\text{GSO}))$ – hash value of the encryption of GSO.
2. Computes HEPI $= h(e_B(\text{PI}))$ – hash value of the encryption of the payment instructions.
3. Computes HPO $= h(\text{HEPI}||\text{HEGSO})$ – hash values of the payment order.
4. Signs HPO by computing "dual signature" DS $= d_C(\text{HPO})$.
5. Sends $e_S(\text{GSO})$, DS, HEPI, and $e_B(\text{PI})$ to shop.

Shop performs the following procedure:

1. Calculates $h(e_S(\text{GSO})) =$ HEGSO;
2. Calculates $h(\text{HEPI}||h(e_S(\text{GSO})))$ and $e_C(\text{DS})$. If they are equal, shop has verified cardholder signature;
3. Computes $d_S(e_S(\text{GSO}))$ to get GSO.
4. Sends HEGSO, $e_B(\text{PI})$, and DS to the bank.

BANK and SHOP ACTIONS

Bank has received HEGSO, $e_B(\mathsf{PI})$, and DS and performs the following actions.

1. Computes $h(e_B(\mathsf{PI}))$ – what should be equal to HEPI.
2. Computes $h(h(e_B(\mathsf{PI}))||\mathsf{HEGSO})\{=\ \mathsf{HPO}\}$ and verifies whether $e_C(\mathsf{DS}) = \mathsf{HPO}$ – that way cardholder signature is verified.
3. Computes $d_B(e_B(\mathsf{PI}))$ to obtain PI;
4. Returns an encrypted (with e_S) digitally signed authorization to shop, guaranteing the payment.

Shop completes the procedure by encrypting, with e_C the receipt to cardholder, indicating that transaction has been completed.

It is easy to verify that the above protocol fulfills basic requirements concerning security, privacy and integrity.

DIGITAL MONEY

Is it possible to have electronic money?

It seems that not, because copies of digital information are indistinguishable from their origin and tehrefore one could hardly prevent double spending,....

T. Okamoto and K. Ohta formulated six properties digital money system should have.

1. One should be able to send e-money through e-networks.
2. It should not be possible to copy and reuse e-money.
3. Transactions using e-money should be done **off-line** – that is no communication with central bank is needed during translation.
4. One should be able to sent e-money to anybody.
5. An e-coin could be divided into e-coins of smaller values.

Several system of e-money have been created that satisfy all or most of the above requirements.

BLIND SIGNATURES – basic setting

So called “blind digital signatures” allow the signer (bank) to sign a message (money) without seeing who is the receiver of the messahe (e-money).

Scenario: Customer Bob would like to give e-money to Shop. E-money are signed by a Bank. Shop must be able to verify Bank's signature. Later, when Shop sends e-money to Bank, Bank should not be able to recognize that it signed these e-money for Bob. bank has therefore to sign money blindly.

Bob can obtain a blind signature for a message m from Bank by executing the Shnorr blind signature protocol described on next slide.

Basic setting

Bank chooses large primes $p, q|(p-1)$ and an $g \in \mathbf{Z}_p$ of order q.

Let $h : \{0,1\}^* \rightarrow \mathbf{Z}_p$ be a collision-free hash function.

Bank's secret will be a randomly chosen $x \in \{0, \ldots, p-1\}$.

Public information: $(p, q, g, y = g^x)$.

BLIND SIGNATURES – protocols

Shnorr's simplified identification protocol in which Bank proves its identity by proving that it knows x.

- Bank chooses a random $r \in \{0, \ldots, q-1\}$ and send $a = g^r$ to Bob. $\{$Bank commits itself to $r\}$.
- Bob send to Bank a random $c \in \{0, \ldots, q-1\}$ $\{$a challenge$\}$.
- Bank sends to Bob $b = r - cx$.
- Bob accepts the proof iff $a = g^b y^c$.

Transfer of the identification scheme to a signature scheme:
Bob does not choose c randomly but as $c = h(m||a)$, where m is message to sign.

Signature: (c, b); Verification rule: $a = g^b y^c$.

Shnorr's blind signature scheme

- Bank sends to Bob $\bar{a} = g^{\bar{r}}$ with random $\bar{r} \in \{0, \ldots, q-1\}$.
- Bob choses random $u, v, w \in \{0, \ldots, q-1\}$, $u \neq 0$, Computes $a = \bar{a}^u g^v y^w$, $c = h(m||a)$, $\bar{c} = (c - w)u^{-1}$ and sends $\bar{c}$ to Bank.
- Bank sends to Bob $\bar{b} = \bar{r} - \bar{c}x$.

Bob verifies whether $\bar{a} = g^{\bar{b}} y^{\bar{c}}$, computes $b = u\bar{b} + v$ and gets blind signature $\sigma(m) = (c, b)$ of m.

Verification condition for the blind signature: $c = h(m||g^b y^c)$.