Digital signature schemes

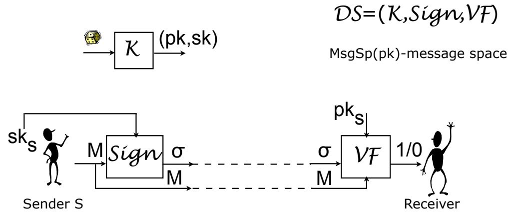
- Let's study the problem of data authentication and integrity in the asymmetric (public-key) setting.
- A sender needs to be assured that a message came from the legitimate sender and was not modified on the way.
- MACs solved this problem but for the symmetric-key setting.
- A digital signature scheme primitive is the solution to the goal of authenticity in the asymmetric setting.

1

Digital signature schemes

$\mathcal{DS}=(\mathcal{K},\mathcal{Sign},\mathcal{VF})$

MsgSp(pk)-message space



It is required that for every M∈MsgSp, every (pk,sk) that can be output by $\mathcal{K}$, if σ is output by $\mathcal{Sign}$, then $\mathcal{VF}$(pk,M,σ)=1

2

Digital signature schemes

- The signing algorithm can be randomized or stateful (but it does not have to be).
- The MsgSp is often $\{0,1\}^*$ for every pk.
- Note that the key usage in a digital signature scheme is reverse compared to an asymmetric encryption scheme:
  - in a digital signature scheme the holder of the secret key is a sender, and anyone can verify
  - in an asymmetric encryption scheme the holder of the secret key is a receiver and anyone can encrypt

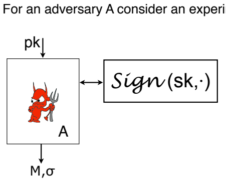
3

Security definition for digital signatures

Fix DS=(*K*,Sign,VF)

Run *K* to get (pk,sk)

For an adversary A consider an experiment $\mathbf{Exp}^{\text{uf-cma}}_{\mathcal{DS}}(A)$



Return 1 iff VF(pk,M,σ)=1 and M∈MsgSp(pk) that was was not queried to the signing oracle

The uf-cma advantage of A is defined as $\mathbf{Adv}^{\text{uf-cma}}_{\mathcal{DS}}(A) = \Pr\left[\mathbf{Exp}^{\text{uf-cma}}_{\mathcal{DS}}(A) = 1\right]$

The resources of A are its time-complexity, the number of queries and the total length of all queries and of the message in the forgery.

4

Plain RSA signature scheme

Algorithm $K(k)$
$((N,e)(N,p,q,d)) \xleftarrow{\$} K^{\$}_{rsa}(k)$
Return $((N,e)(N,p,q,d))$

Algorithm $\mathrm{Sign}_{N,p,q,d}(M)$
If $M \notin \mathbf{Z}_N^*$ then return $\perp$
$x \leftarrow M^d \bmod N$
Return x

Algorithm $\mathrm{VF}_{N,e}(M,x)$
If ($M \notin \mathbf{Z}_N^*$ or $x \notin \mathbf{Z}_N^*$) then return 0
If $M = x^e \bmod N$ then return 1 else return 0

- Is Plain RSA signature scheme secure?

Plain RSA is not secure

Forger $F_1^{\mathrm{Sign}_{N,p,q,d}(\cdot)}(N,e)$
Return $(1,1)$

Forger $F_2^{\mathrm{Sign}_{N,p,q,d}(\cdot)}(N,e)$
$x \xleftarrow{\$} Z_N^*$; $M \leftarrow x^e \bmod N$
Return (M,x)

Forger $F_3^{\mathrm{Sign}_{N,e}(\cdot)}(N,e)$
$M_1 \xleftarrow{\$} Z_N^* - \{1, M\}$; $M_2 \leftarrow M M_1^{-1} \bmod N$
$x_1 \leftarrow \mathrm{Sign}_{N,e}(M_1)$; $x_2 \leftarrow \mathrm{Sign}_{N,e}(M_2)$
$x \leftarrow x_1 x_2 \bmod N$
Return (M,x)

All adversaries (forgers) have uf-cma advantages 1 and are efficient.

Hash-then-invert paradigm

- We want to have an RSA-based signature scheme
  - that resists the attacks above
  - has a more flexible message space
  - provably secure
- An idea: let's hash the message first

Let Hash be a function family whose key space is the set of all moduli N that can be output by $K^{\$}_{rsa}$ s.t. $\mathsf{Hash}_N \colon \{0,1\}^* \rightarrow Z_N^*$

Algorithm $\mathrm{Sign}_{N,p,q,d}(M)$
$y \leftarrow \mathsf{Hash}_N(M)$
$x \leftarrow y^d \bmod N$
Return x

Algorithm $\mathrm{VF}_{N,e}(M,x)$
$y \leftarrow \mathsf{Hash}_N(M)$
$y' \leftarrow x^e \bmod N$
If $y = y'$ then return 1 else return 0

- What properties of the hash function do we need?
- If we have hash that “destroys” the algebraic structure and is collision resistant the obvious attacks do not apply.
- However, to prove security we need more:
  - we need to assume that the hash function is a random function
  - this is not a realistic assumption

Full-Domain-Hash (FDH) RSA signature scheme

- Let H: $\{0,1\}^* \rightarrow Z_N^*$ be a random function to which all parties have oracle access to
- FDH-RSA is a signature scheme $\mathcal{DS} = (\mathcal{K}_{\text{rsa}}, \text{Sign}, \text{VF})$

Algorithm $\text{Sign}_{N,p,q,d}^{H(\cdot)}(M)$	Algorithm $\text{VF}_{N,e}^{H(\cdot)}(M,x)$
$y \leftarrow H(M)$	$y \leftarrow H(M)$
$x \leftarrow y^d \bmod N$	$y' \leftarrow x^e \bmod N$
Return x	If $y = y'$ then return 1 else return 0

9

Security of the FDH-RSA scheme

- <u>Theorem</u>. Under the RSA assumption the FDH-RSA signature scheme is uf-cma secure in the random oracle (RO) model.
- <u>Proof</u>. Let K_{rsa} be an RSA generator and let DS be the FDH-RSA signature scheme. Let *F* be an adversary making at most q_{hash} queries to its hash oracle and at most q_{sign} queries to its signing oracle where $q_{hash} \geq q_{sign}$ +1. Then there exists an adversary I with comparable resources s.t.

$$\mathbf{Adv}_{\mathcal{DS}}^{\text{uf-cma}}(F) \;\leq\; q_{\text{hash}} \cdot \mathbf{Adv}_{\mathcal{K}_{\text{rsa}}}^{\text{ow-kea}}(I)$$

10

- *I* has to simulate for *F* the following experiment

Experiment $\mathbf{Exp}_{\mathcal{DS}}^{\text{uf-cma}}(F)$
$((N,e),(N,p,q,d)) \stackrel{\$}{\leftarrow} \mathcal{K}_{\text{rsa}}$
$H \stackrel{\$}{\leftarrow} \mathsf{Func}(\{0,1\}^*, \mathbf{Z}_N^*)$
$(M,x) \stackrel{\$}{\leftarrow} F^{H(\cdot),\text{Sign}_{N,p,q,d}^{H(\cdot)}(\cdot)}(N,e)$
If the following are true return 1 else return 0:
– $\text{VF}_{pk}^{H}(M,\sigma) = 1$
– M was not a query of A to its oracle

- *I* has to give *F* a public key and answer its hash and signing queries.
- *I* has to use F's forgery to invert its challenge.
- The idea: *I* guesses when *F* makes a hash query on a message in the future forgery, and gives its challenge to *F* as an answer to this hash query. Other hash and signing queries are answered differently (using a little trick).

11

Inverter $I(N,e,y)$
Initialize arrays $Msg[1 \ldots q_{\text{hash}}]$, $X[1 \ldots q_{\text{hash}}]$, $Y[1 \ldots q_{\text{hash}}]$ to empty
$j \leftarrow 0$; $i \stackrel{\$}{\leftarrow} \{1, \ldots, q_{\text{hash}}\}$
Run F on input (N,e)
If F makes oracle query (hash, M)
then $h \leftarrow H\text{-}Sim(M)$; return h to F as the answer
If F makes oracle query (sign, M)
then $x \leftarrow \text{Sign-}Sim(M)$; return x to F as the answer
Until F halts with output (M,x)
$y' \leftarrow H\text{-}Sim(M)$
Return x

$Msg[j]$ – The j-th hash query in the experiment
$Y[j]$ – The reply of the hash oracle simulator to the above, meaning the value playing the role of $H(Msg[j])$. For $j = i$ it is y.
$X[j]$ – For $j \neq i$, the response to sign query $Msg[j]$, meaning it satisfies $(X[j])^e \equiv Y[j] \pmod N$. For $j = i$ it is undefined.

12

We will make use of a subroutine *Find* that given an array A, a value v and index m, returns 0 if $v \notin \{A[1], \ldots, A[m]\}$, and else returns the smallest index l such that $v = A[l]$.

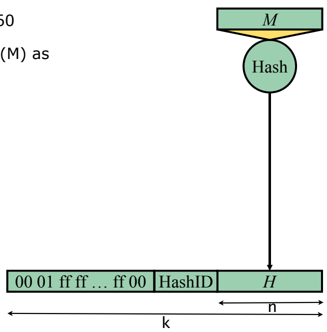
Subroutine $H\text{-}Sim(v)$
  $l \leftarrow Find(Msg, v, j)$; $j \leftarrow j+1$; $Msg[j] \leftarrow v$
  If $l = 0$ then
    If $j = i$ then $Y[j] \leftarrow y$
    Else $X[j] \stackrel{\$}{\leftarrow} Z_N^*$; $Y[j] \leftarrow (X[j])^e \bmod N$
    EndIf
    Return $Y[j]$
  Else
    If $j = i$ then abort
    Else $X[j] \leftarrow X[l]$; $Y[j] \leftarrow Y[l]$; Return $Y[j]$
    EndIf
  EndIf

Subroutine Sign-$Sim(M)$
  $h \leftarrow H\text{-}Sim(M)$
  If $j = i$ then abort
  Else return $X[j]$
  EndIf

In practice: RSA PKSC#1

- Fix a function Hash: $\{0,1\}^* \rightarrow \{0,1\}^n$ where $n \geq 128$
- E.g. for SHA1 n=160
- Define PKCS-HASH(M) as

PKCS-HASH(M):



- If Hash is collision resistant, so is PKCS-HASH.
- But hardness of computing the inverse of the RSA function on a random point in Z_N^* does not imply that on a point in S={PKCS-HASH(M): M{0,1}*}
- The are no attacks known, but it does not mean we should not be concerned.