A Transitive Signature Scheme for Directed Trees

Richard McPherson and abhi shelat

1 Introduction

Digital signatures are a way of cryptographically signing data to verify its integrity and authenticity [2]. In most digital signature schemes, the author (let's call her Alice) signs a messages using a secret key and then publishes the signature and message. A reader can then verify Alice as the author of the complete message without being able to forge a signature of Alice's on any new message. Digital signatures are akin to written signatures, except for digital documents.

In mathematics a graph is composed of two sets. One, $V = \{u, v, w, \ldots\}$, is a collection of vertices and the other, $E = \{(u, v), (v, w)\}$, is a collection of edges or relations between the vertices. Graphs can be directed, where edge (u, v) means that the edge goes directionally from u to v, or undirected, where (u, v) means the edge goes from both u and v. Graphs are often used to show relations. A directed graph can show levels of authority such as in the armed forces, where (u, v) means that u has authority over v. An undirected graph can be used to show administrative permissions, where if u is connected to v then they share the same permissions.

Graphs often have a transitive nature in that if there is a path from u to v and from v to w, then there exists a path from u to w. In 2002 Micali and Rivest proposed a new type of digital signature schemes for graphs [6]. If given valid signatures of (u, v) and (v, w) a scheme were able to generate a valid signature of (u, w), then only a minimum number of edges would need to be signed that would then be able to produce the original graph as the signed graph's transitive closure.

Micali and Rivest defined a transitive signature scheme with two properties. First it was to be a signature scheme for vertexes and edges that is it is computationally hard to forge a digital signature of a new vertex or edge in an adaptive fashion. Second it was to allow the transitive closure of previously signed edges to be generated by any observer [6]. They presented a scheme for undirected graphs and left the directed case as an open problem.

Following the Micali and Rivest paper, many new undirected signature schemes were presented based on factoring [3, 1, 12, 9], discrete logs [1, 5], and Diffie-Hellman groups [1, 8]. Sujing proved that a security assumption of non-adaptive security is strong enough to cover the initial adaptive chosen message security [9].

The question of whether a directed signature scheme exists for all graphs is still an open problem. In 2003, Hohenberger showed that signatures for edges in such a scheme would form an Abelian trapdoor group with infeasible inversion, a group not known to exist [4]. Yi proposed a scheme for directed trees in 2007 and Neven proposed another scheme in 2008 [11, 7]. In 2009, Xu publish another signature scheme for directed trees that stays constant in size with applications of composed signatures, something the previous schemes had not done [10].

We propose a new digital signature scheme for directed trees that is provably secure under non-adaptive chosen message attacks in the random oracle model. While it has a higher signature size and cost than Xu's $\mathcal{DDTS}$ scheme, it has a a lower verification and composition cost [10]. The signature is constant in size, though we are only able to prove the security of our scheme using a weaker notion.

2 Definitions

Formally, we follow Yi with definition to a transitive signature scheme and correctness [11]:

A directed transitive signature scheme $\mathcal{DTS}$=(TKG, TSign, TVf, Comp) is composed of four polynomial-time algorithsm.

1. TKG is the randomized key generation algorithm. It takes input $1^k, k \in \mathbb{N}$ and outputs a pair (tpk, tsk), where tpk is the public key and tsk is the secret key.
2. TSign is the signing algorithm. It takes as input the secret key tsk and nodes $i, j \in \mathbb{N}$ and returns an original signature $\sigma_{i,j}$ of edge (i, j) relative to tsk.
3. TVf is the deterministic verification algorithm. It takes as input the public key tpk, nodes $i, j \in \mathbb{N}$, and a candidate signature σ and returns 1 if the signature is valid for the inputs and 0 otherwise.
4. Comp is the deterministic composition algorithm which takes the public key tpk, nodes $i, j, k \in \mathbb{N}$ and values $\sigma_{i,j}$, $\sigma_{j,k}$ and returns either a composed signature $\sigma_{i,k}$ of the edge (i, k) or $\perp$ to indicate failure.

Definition 2.1. A $\mathcal{DTS}$ is considered correct if for every algorithm A and every $k \in \mathbb{N}$, the output of the experiment of Figure 1 is true with zero probability.

We have modified Yi's security definition slightly, making it weaker. F now gives the edge on which it will forge and all the signatures it requests at the start of the experiment. We hope to strengthen the security definition in the future.

We define the experiment in Figure 2 as $\mathbf{Exp}_{\mathcal{DTS},F}(k)$ for any algorithm F and $k \in \mathbb{N}$. Let the advantage of F be defined as follows

$$\mathbf{Adv}_{\mathcal{DTS},F}(k) = \text{Prob}[\mathbf{Exp}_{\mathcal{DTS},F}(k) = 1]$$

Definition 2.2. A $\mathcal{DTS}$ is considered secure if $\mathbf{Adv}_{\mathcal{DTS},F}(k)$ is negligible for any adversary F that runs in time polynomial to the security parameter k.

```
$(tpk, tsk) \leftarrow \mathrm{TKG}(1^k)$
$S \leftarrow \emptyset$; $Legit \leftarrow true$; $NotOK \leftarrow false$
Run A with its oracles until it halts, replying to its oracle queries as follows:
If A makes TSign query $i, j$ then
  If $[(i = j) \vee (\{i, j\} \in V)]$ then $Legit \leftarrow false$
  Else
    Let $\sigma_{i,j}$ be the output of the TSign oracle
    $S \leftarrow S \cup \{(i, j, \sigma_{i,j})\}$
    If TVf$(tpk, i, j, \sigma_{i,j}) = 0$ then $NotOK \leftarrow true$
If A makes Comp query on $(i, j, k), \sigma_{i,j}, \sigma_{j,k}$ then
  If $[(i, j, k$ are no all distinct$) \vee ((i, j, \sigma_{i,j}) \notin S) \vee ((j, k, \sigma_{j,k}) \notin S)]$
  Then $Legit \leftarrow false$
  Else
    Let $\sigma_{i,k}$ be the output of the Comp oracle
    $S \leftarrow S \cup \{(i, k, \sigma_{i,k})\}$
    If (TVf$(tpk, i, k, \sigma_{i,k}) = 0$) then $NotOK \leftarrow true$
When A halts, it outputs $(Legit \wedge NotOK)$
```

Figure 1: An experiment to define the correctness of a $\mathcal{DTS}$

```
$(tsk, tpk) \leftarrow \mathrm{TKG}(1^k)$
$(i', j') \leftarrow F(tpk, \perp)$
$P = ((i_1, j_1), (i_2, j_2), \ldots) \leftarrow F(tpk, \perp)$
$S = \{(i, j, \sigma_{i,j})\} \leftarrow$ TSign$(tsk, i, j)$ s.t. $(i, j) \in P$
$(i', j', \sigma'_{i',j'}) \leftarrow$ F$(tpk, S)$
Let $E = \{(i, j) | \exists (i, j, \sigma_{i,j}) \in S\}$, $V = \{i | (\exists (i, j) \in E) \vee (\exists (j, i) \in E)\}$
Let $G = (V, E)$
If $(i', j', \sigma'_{i',j'}) \in S \vee$ TVf$(i', j', \sigma'_{i',j'}) = 0$ then output 0
Else output 1
```

Figure 2: An experiment to define the security of a $\mathcal{DTS}$

3 Algorithm

KeyGen(1^λ) : A bilinear group $\mathbb{G}$ with respect to $\mathbf{e} : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}$ of prime order $p > 2^\lambda$ is selected with a generator g. Let $H_s : \{0,1\}^* \rightarrow \mathbb{G}$, and $H_e : \{0,1\}^* \rightarrow \mathbb{G}$ denote two hash functions. Choose a random $\alpha \leftarrow \mathbb{Z}_p$. The secret key, tsk, is: (α) and the public key, tpk, is: $(H_s, H_e, g, \mathbf{e}(g,g)^\alpha)$.

SignVertex(v_i) : Any secure signature scheme can be used to sign vertices.

SignEdge($tsk, (v_i, v_j)$) : If not previously defined, generate random numbers $x_i, x_j, r_{i,j}, r'_{i,j} \leftarrow \mathbb{Z}_p$.

A signature is composed of the following values

$$S_{i,j} := g^\alpha g^{-x_i} H_s(v_i)^{r_{i,j}} \qquad \widetilde{S_{i,j}} := g^{r_{i,j}}$$

$$A_{i,j} := g^{x_i} g^{-x_j}$$

$$E_{i,j} := g^{x_i} H_e(v_j)^{r'_{i,j}} \qquad \widetilde{E_{i,j}} := g^{r'_{i,j}}$$

$$B_{i,j} := S_{i,j} E_{i,j} = g^\alpha H_s(v_i)^{r_{i,j}} H_e(v_j)^{r'_{i,j}}$$

VerifyEdge($tpk, B_{i,j}, \widetilde{S_{i,j}}, \widetilde{E_{i,j}}$) : Check that

$$e(B,g) = e(g,g)^\alpha \cdot e(H_s(v_i), \widetilde{S_{i,j}}) \cdot e(H_e(v_j), \widetilde{E_{i,j}})$$

Compose Signature Given signatures for (v_i, v_j) and (v_j, v_k) a valid signature for (v_i, v_k) can be constructed in the following manner

$$S_{i,k} := S_{i,j} \qquad \widetilde{S_{i,k}} := \widetilde{S_{i,j}}$$

$$A_{i,k} := A_{i,j} A_{j,k}$$

$$E_{i,k} := A_{i,j} E_{j,k} \qquad \widetilde{E_{i,k}} := \widetilde{E_{j,k}}$$

$$B_{i,k} := S_{i,k} E_{i,k}$$

4 Security Proof

Suppose an adversary $\mathcal{A}$ can produce a forgery with probability ϵ on the selective unforgeability game, then we can construct an adversary $\mathcal{B}$ that breaks the CDH assumption with probabilty ϵ. On input to the CDH challenge (g, g^a, g^b), $\mathcal{B}$ proceeds as follows:

4.1 Selective Disclosure

$\mathcal{A}$ first announces the edge (v_n, v_m) on which he will forge and a set of edges on which he requests valid signatures.

4.2 Setup

Let $\mathbf{e}(g, g)^\alpha := \mathbf{e}(g^a, g^b)$ which sets $\alpha = ab$ as the secret key. Have $\mathcal{B}$ give $\mathcal{A}$ the public key $(g, \mathbf{e}(g, g)^\alpha)$. We can divide the edges $\mathcal{A}$ has requested signatures on into two subsets. Let $\mathcal{E}'$ designate the edges that are connected to v_m and let $\mathcal{E}$ be the remaining edges. $\mathcal{B}$ will answer all of $\mathcal{A}$'s queries to the random oracles H_s and H_e and the signing oracle as specified below.

4.3 Queries

$\mathcal{A}$ may make any of the following queries which $\mathcal{B}$ will answers as follows:

1. $H_s(v)$: The random oracle is answered as follows. If the query has been made before, return the same responce as before. Otherwise, select a random $\delta \in \mathbb{Z}_p$ and return the response as:

$$H_s(v) = \begin{cases} g^{\delta} & \begin{array}{l} \text{if } v = v_m \\ \text{(the hash if of } v_m) \end{array} \\ g^{b\delta} & \begin{array}{l} \text{otherwise} \\ \text{(the hash is not of } v_m) \end{array} \end{cases}$$

2. $H_e(v)$: The random oracle is answered as follows. If the query has been made before, return the same responce as before. Otherwise, select a random $\lambda \in \mathbb{Z}_p$ and return the responce as:

$$H_e(v) = \begin{cases} g^{\lambda} & \begin{array}{c}\text{if } v = v_n \\ \text{(the hash is of } v_n\text{)}\end{array} \\ g^{b\lambda} & \begin{array}{c}\text{otherwise} \\ \text{(the hash is not of } v_n\text{)}\end{array} \end{cases}$$

3. NewEdgeSig$((v_i, v_j))$: An edge signature is composed of the following values:

   (a) If $(v_i, v_j) \in \mathcal{E}'$

   Start: If not previously defined, choose random numbers $x'_i, r_{i,j} \leftarrow \mathbb{Z}_p$ and set $x_i = x'_i + \alpha$

   i. If $v_i = v_m$

$$\begin{aligned} S_{i,j} &= g^{-x'_i} g^{\delta r_{i,j}} \\ &= g^{\alpha} g^{-x_i} H_s(v_i)^{r_{i,j}} \\ \widetilde{S_{i,j}} &= g^{r_{i,j}} \end{aligned}$$

   ii. Else, $v_i \neq v_m$

$$\begin{aligned} S_{i,j} &= g^{-x'_i} g^{b\delta r_{i,j}} \\ &= g^{\alpha} g^{-x_i} H_s(v_i)^{r_{i,j}} \\ \widetilde{S_{i,j}} &= g^{r_{i,j}} \end{aligned}$$

   Across: If not previously defined, choose random number x'_j and set $x_j = x_j = \alpha$

$$\begin{aligned} A_{i,j} &= g^{x'_i} g^{-x'_j} \\ &= g^{x_i} g^{-x_j} \end{aligned}$$

   End: If not previously defined, choose random number $s'_{i,j}$ and set $r'_{i,j} = -a/\lambda + s'_{i,j}$

$$\begin{aligned} E_{i,j} &= g^{x'_i} g^{b\lambda s'_{i,j}} \\ &= g^{x_i} H_e(v_i)^{r'_{i,j}} \\ \widetilde{E_{i,j}} &= g^{-a/\lambda + s'_{i,j}} \\ &= g^{r'_{i,j}} \end{aligned}$$

B:

i. If $v_i = v_m$

$$\begin{aligned} B_{i,j} &= g^{\delta r_{i,j}} g^{b\lambda s'_{i,j}} \\ &= g^{\alpha} g^{\delta r_{i,j}} g^{b\lambda r'_{i,j}} \\ &= g^{\alpha} H_s(v_i)^{r_{i,j}} H_e(v_j)^{r'_{i,j}} \end{aligned}$$

ii. Else, if $v_j \neq v_m$

$$\begin{aligned} B_{i,j} &= g^{b\delta r_{i,j}} g^{b\lambda s'_{i,j}} \\ &= g^{\alpha} g^{b\delta r_{i,j}} g^{b\lambda r'_{i,j}} \\ &= g^{\alpha} H_s(v_i)^{r_{i,j}} H_e(v_j)^{r'_{i,j}} \end{aligned}$$

(b) Else $(v_i, v_j) \in \mathcal{E}$

Start: If not previously defined, choose random numbers $x_i, s_{i,j} \leftarrow \mathbb{Z}_p$ and set $r_{i,j} = -a/\delta + s_{i,j}$

$$\begin{aligned} S_{i,j} &= g^{-x_i} g^{b\delta s_{i,j}} \\ &= g^{\alpha} g^{-x_i} H_s(v_i)^{r_{i,j}} \\ \widetilde{S_{i,j}} &= g^{-a/\delta + s_{i,j}} \\ &= g^{r_{i,j}} \end{aligned}$$

Across: If not previously defined, choose random number x_j

$$A_{i,j} = g^{x_i} g^{-x_j}$$

End: If not previously defined, choose random number $r'_{i,j}$

i. $v_j = v_n$

$$\begin{aligned} E_{i,j} &= g^{x_i} g^{\lambda r'_{i,j}} \\ &= g^{x_i} H_e(v_i)^{r'_{i,j}} \\ \widetilde{E_{i,j}} &= g^{r'_{i,j}} \end{aligned}$$

ii. $v_j \neq v_n$

$$\begin{aligned} E_{i,j} &= g^{x_i} g^{b\lambda r'_{i,j}} \\ &= g^{x_i} H_e(v_i)^{r'_{i,j}} \\ \widetilde{E_{i,j}} &= g^{r'_{i,j}} \end{aligned}$$

B:

i. If $v_j = v_n$

$$\begin{aligned} B_{i,j} &= g^{b\delta s_{i,j}} g^{b\lambda r'_{i,j}} \\ &= g^{\alpha} g^{\delta r_{i,j}} g^{b\lambda r'_{i,j}} \\ &= g^{\alpha} H_s(v_i)^{r_{i,j}} H_e(v_j)^{r'_{i,j}} \end{aligned}$$

ii. Else, if $v_i \neq v_n$

$$\begin{aligned} B_{i,j} &= g^{b\delta s_{i,j}} g^{b\lambda r'_{i,j}} \\ &= g^{\alpha} g^{b\delta r_{i,j}} g^{b\lambda r'_{i,j}} \\ &= g^{\alpha} H_s(v_i)^{r_{i,j}} H_e(v_j)^{r'_{i,j}} \end{aligned}$$

4.4 Response

Eventually $\mathcal{A}$ outputs a valid edge-signature pair

$$((v_m, v_n), (B, S_{m,n}, E_{m,n}, \widetilde{S_{m,n}}, \widetilde{E_{m,n}})).$$

Since the signature is valid, we know that

$$e(B, g) = e(g, g)^{\alpha} \cdot e(H_s(v_m), \widetilde{S_{m,n}}) \cdot e(H_e(v_n), \widetilde{E_{m,n}})$$

and that

$$\begin{aligned} B &= g^{\alpha} H_s(v_m)^{r_{m,n}} H_e(v_n)^{r'_{m,n}} \\ &= g^{\alpha} g^{\delta r_{m,n}} g^{\lambda r'_{m,n}} \end{aligned}$$

We can then find g^{α} in the following manner:

$$g^{\alpha} = B(\widetilde{S_{m,n}})^{-\delta}(\widetilde{E_{i,j}})^{-\lambda}$$

$\mathcal{B}$ thus breaks the CDH security assumption with probability ϵ. Therefore ϵ must be negligible and the construction is secure.

References

[1] Mihir Bellare and Gregory Neven. Transitive signatures: New schemes and proofs. *IEEE Transactions on Information Theory*, 51:2133–2151, 2005.

[2] Whitfield Diffie and Martin E. Hellman. New directions in cryptography, 1976.

[3] Dang Nguyen Duc, Han Kyusuk, Zeen Kim, and Kwangjo Kim. A new transitive signature scheme based on rsa-based security assumptions.

[4] Susan Rae Hohenberger. The cryptographic impact of groups with infeasible inversion. In *Masters thesis, MIT*, 2003.

[5] Zichen Li, Juanmei Zhang, and Dong Zheng. New transitive signature scheme based on discreted logarithm problem. In *Progress on Cryptography*, volume 769 of *The International Series in Engineering and Computer Science*, pages 113–122. Springer Netherlands, 2004.

[6] Silvio Micali and Ronald L. Rivest. Transitive signature schemes. In *CT-RSA '02: Proceedings of the The Cryptographer's Track at the RSA Conference on Topics in Cryptology*, pages 236–243, London, UK, 2002. Springer-Verlag.

[7] Gregory Neven. A simple transitive signature scheme for directed trees. *Theoretical Computer Science*, 396(1-3):277 – 282, 2008.

[8] Siamak Fayyaz Shahandashti, Mahmoud Salmasizadeh, and Javad Mohajeri. A provably secure short transitive signature scheme from bilinear group pairs. In *Security in Communication Networks*, volume 3352 of *Lecture Notes in Computer Science*, pages 60–76. Springer Berlin / Heidelberg, 2005.

[9] Zhou Sujing. Transitive signatures based on non-adaptive standard signatures. Technical report.

[10] Jia Xu. On directed transitive signature. Cryptology ePrint Archive, Report 2009/209, 2009.

[11] Xun Yi. Directed transitive signature scheme. In *Topics in Cryptology 2007*, volume 4377 of *Lecture Notes in Computer Science*, pages 129–144. Springer Berlin / Heidelberg, 2007.

[12] Huafei Zhu, Bao Feng, and Robert H. Deng. A transitive signature scheme provably secure against adaptive chosen-message attack, 2003.