Short Signature and Universal Designated Verifier Signature Without Random Oracles

Rui Zhang[1], Jun Furukawa[2], and Hideki Imai[1]

[1] The University of Tokyo
zhang@imailab.iis.u-tokyo.ac.jp, imai@iis.u-tokyo.ac.jp
[2] NEC Corporation
j-furukawa@ay.jp.nec.com

Abstract. We propose the first universal designated verifier signature (UDVS) scheme whose security can be proven without random oracles, whereas the security of all previously known UDVS schemes are proven only when random oracles are assumed. To achieve our goal, we present a new short signature scheme without random oracles, which is a variant of BB04 scheme [4]. We also give new security definitions to UDVS. We note that our weakest security definitions are even stronger than any of previously known security definitions: We allow adversaries to behave more adaptively in oracle accessing and we also consider adaptive chosen public key attacks. The security of our UDVS scheme is then proven according to the new security definitions.

1 Introduction

Short signatures and related assumptions. Short signature was recently proposed by Boneh, Lynn and Shacham (BLS01) [5] with the good property that its size is only about half of DSA signatures with almost the same level of security. Security of BLS01 scheme was based on Gap Diffie-Hellman assumption in random oracle model [3, 8]. Later, Boneh and Boyen proposed another signature scheme (BB04) which is almost as short as DSA and the security of this scheme relies on Strong Diffie-Hellman (SDH) assumption [4, 14] over bilinear group pairs without random oracles. SDH assumption can be informally stated as follows: for $\mathbb{G}=<g>$ of large prime order p, given $g, g^x, g^{(x^2)}, ..., g^{(x^q)} \in \mathbb{G}$, there is no probabilistic polynomial time algorithm able to compute $(c, g^{1/(x+c)})$, for arbitrary $c \in Z_p^*$. The SDH assumption is analogous to the Strong RSA assumption, which has already been used to construct signature schemes existential unforgeable against chosen message attack without random oracles [7, 9].

Universal designated verifier signatures. A normal digital signature has publicly verifiable property just as its real world counterpart. A verifier of a signature can convince any third party the fact by presenting a digital signature on a message. However, in the real world, sometimes it is desirable that a verifier should not present the signatures to other parties, such as certificates for hospital records, income summary, etc.

J. Ioannidis, A. Keromytis, and M.Yung (Eds.): ACNS 2005, LNCS 3531, pp. 483–498, 2005.

Universal designated verifier signature (UDVS), introduced by Steinfeld et al [15, 16], is an important tool to protect the privacy of the signature holder from dissemination of signatures by verifiers. UDVS schemes are signature schemes with additional functionality where any holder of the signature alone can transform the signature to a non-interactive proof statement for a desired *designated verifier* using the knowledge of the signature, such that the designated verifier can verify the message is signed by the signer but cannot prove the same fact to a third party, since he can also produce such a proof statement using his secret key.

UDVS can be viewed as an application of general *designated verifier proofs*, introduced by Jakobsson, Sako and Impagliazzo [12], where a prover designates a non-interactive proof statement to a designated verifier, who can simulate this proof with his secret key thus cannot transfer it to a third party. We refer to [15, 16] for more related work and applications of UDVS.

The good properties of UDVS make it an important tool to prevent dissemination of digital signatures in user certification systems. It is thus desirable to have rigorous model and corresponding formal analysis regarding UDVS schemes. However, there is still no provably secure UDVS scheme ever reported yet, except those in the *random oracle* model [15, 16].

Random oracle model is a formal model in analyzing cryptographic schemes, where a hash function is considered as a black-box that contains a random function. However, many impossibility results, e.g. [6], have shown that security in the random oracle model does not imply the security in the real world in that a scheme can be secure in the random oracle model and yet be broken without violating any particular intractability assumption, and without breaking the underlying hash functions. Consequently, to design a provable secure UDVS based on standard intractability assumptions is both of theoretical and practical importance.

1.1 Our Contribution

Secure signature scheme without random oracles. We present a new digital signature scheme which is a variant of BB04 scheme, and prove its security based on the Strong Diffie-Hellman assumption [4, 14]. This signature scheme does not resort to random oracles. We then construct our UDVS without random oracles using this new signature scheme.

Refined security definitions on UDVS. Much work on modeling UDVS was previously done in the two papers of Steinfeld et al [15, 16], where two security definitions are introduced, namely, "DV-Unforgeability" and "PR-Privacy". However, the former requirement is weak in the sense that an adversary is not allowed to adaptively choose a designated verifier whom the adversary tries to forge DV-signatures with respect to. The latter requirement is weak in the sense that an adversary is not allowed to access designation oracle with respect to the message that the adversary chooses as a target before this target message is cho-

sen. Hence, adversaries in the previous models are not allowed to fully-adaptively probe "weak" target verifier and "weak" target messages in their attacks.

In our refined security definitions, we give adversaries more freedom to select target verifiers and target messages. We also allow them to additionally access to key registration oracle and designation oracle. These abilities reflect those of real world adversaries. Moreover, we consider strong unforgeability (sEUF) in the sense of [1] for DV-signatures, though we also introduce a commonly-adopted weak version (wEUF), which suffices in many applications. We emphasize that our weak version of definition on DV-unforgeability is still strictly stronger than those given in [15, 16].

First secure UDVS scheme without random oracles. All previous known UDVS schemes are only secure in the random oracle model. We give the first provable secure construction of UDVS without random oracles. This also answers an open question raised in [15], where Steinfeld et al wondered how to construct UDVS schemes without random oracles. Furthermore, our scheme is analyzed in the new model, which is strictly stronger. The security of our UDVS is proven under both SDH and a natural extended version of the "KEA1" assumption [2, 11] in bilinear groups.

2 Preliminary

Conventions. Let $x \leftarrow X$ denote x is uniformly selected from distribution X. If X is an algorithm, $x \leftarrow X$ denotes x is set to the output of X. $x \leftarrow y$ denotes y is assigned to x. We say a function $f : \mathbb{N} \rightarrow \mathbb{R}$ is *negligible*, if for any $c > 0$, there exits $k_0 \in \mathbb{N}$, such that $|f(k)| < k^{-c}$ holds for any $k > k_0$, denoted as $\mathsf{negl}(k)$.

2.1 Signature Schemes

Syntax. A signature scheme consists of three algorithms: $\mathcal{SIG} = (\mathsf{KG}, \mathsf{Sig}, \mathsf{Ver})$. $\mathsf{KG}(1^k)$ is the key generation algorithm, with internal random coin flipping, outputs a pair of keys (pk, sk), where pk is the public verification key and sk is the secret signing key. The signing algorithm S takes sk and a message m from the associated message space $\mathcal{M}$, with possible internal random coin flipping, outputs a signature s, denoted as $s \leftarrow \mathsf{Sig}_{sk}(m)$. Ver is the deterministic verification algorithm, takes pk, a message m, and s as input, outputs `acc` for "accepted" or `rej` for "rejected" as the verification result. The subscript of pk or sk may be omitted in clear contexts. We also require that $\mathsf{Ver}(m, \mathsf{Sig}(m)) = \mathtt{acc}$ for all $m \in \mathcal{M}$.

Existential unforgeability. The widely-accepted security definition for digital signature is existential unforgeability against adaptive chosen message attack [10]. But we here consider a stronger version of it [1], called *strong existential unforgeability against adaptive chosen message attack* (sEUF-CMA).

Setup. KG is run, generates a public/secret key pair (pk, sk). pk is given to the adversary $\mathcal{A}$ and sk is given to a challenger.

Training. The adversary $\mathcal{A}$ requests signatures on at most q_s messages $m_1, m_2, ..., m_{q_s} \in \mathcal{M}$ chosen adaptively by itself. The challenger responds with the corresponding signature $s_i = \mathsf{Sig}(m_i)$, $i = 1, ..., q_s$.

Forge. Eventually, the adversary $\mathcal{A}$ outputs a pair (m, s) and wins the game if $(m, s) \notin \{(m_1, s_1), ..., (m_{q_s}, s_{q_s})\}$ and $\mathsf{Ver}(m, s) = \mathtt{acc}$.

For comparison, we also describe weak existentially unforgeability (wEUF-CMA), where the adversary is considered to win the game only if it can output $m \notin \{m_1, ..., m_{q_s}\}$ and $\mathsf{Ver}(m, s) = \mathtt{acc}$. However, we consider *only* strong existential unforgeability (sEUF) in the rest of the paper.

Let Adv be the probability that the adversary wins the above game, which is taken over the coin toss made by $\mathcal{A}$ and the Challenger.

Definition 1. *An adversary (q_s, t, ϵ)-breaks the signature scheme if $\mathcal{A}$ makes at most q_s signature queries, runs in time at most t and* Adv *is at least ϵ. A signature scheme is (q_s, t, ϵ)-strong existentially unforgeable if under an adaptive chosen message attack if no probabilistic polynomial time adversary (q_s, t, ϵ)-breaks it.*

2.2 Bilinear Groups

Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two cyclic groups of prime order p, where Computational Diffie-Hellman problem (CDH) is considered hard. Let g_1 be a generator of $\mathbb{G}_1$ and g_2 be a generator of $\mathbb{G}_2$. A bilinear map is a map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ such that $|\mathbb{G}_1| = |\mathbb{G}_2| = |\mathbb{G}_T|$ with following properties:

1. Bilinear: for all $u \in \mathbb{G}_1$ and $v \in \mathbb{G}_2$ and $a, b \in \mathbb{Z}$, $e(u^a, v^b) = e(u, v)^{ab}$.
2. Non-degenerate: $e(g_1, g_2) \neq 1$.

Two groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of prime order p are bilinear map group pair if there is an additional group $\mathbb{G}_T$ with $|\mathbb{G}_1| = |\mathbb{G}_2| = |\mathbb{G}_T|$, such that there exist a bilinear map and an efficiently computable isomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$. Especially e, ψ, and group operations in $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ can be computed efficiently. Joux and Nguyen [13] showed that an efficiently computable bilinear map e provides an algorithm for solving the Decisional Diffie-Hellman problem (DDH) in $(\mathbb{G}_1, \mathbb{G}_2)$. One can set $\mathbb{G}_1 = \mathbb{G}_2$, however, above notations indicates more general cases where $\mathbb{G}_1 \neq \mathbb{G}_2$ so that certain families of elliptic curves can be used to obtain shorter representations for the group element of $\mathbb{G}_1$.

2.3 Strong Diffie-Hellman Assumption

Definition 2 ((q, t, ε)-Strong Diffie-Hellman Assumption). *Let g_1 and g_2 be as above with $g_1 = \psi(g_2)$. We say a probabilistic polynomial time (PPT) adversary $\mathcal{A}$ (q, t, ε)-breaks the Strong Diffie Hellman problem in $(\mathbb{G}_1, \mathbb{G}_2)$ if after given q-tuples of $g_2^{(x^i)}$ with $1 \leq i \leq q$ and running time t, has the probability ε of*

outputting a pair $(c, g_1^{1/(x+c)})$ *where* $c \in Z_p^*$. *Here the probability is taken over the random choice of generator* g_2, *the choice of* $x \in Z_p^*$, *and internal random coins of* $\mathcal{A}$. *The* (q, t, ε)*-Strong Diffie-Hellman assumption holds if no PPT algorithm solves the SDH problem. For simplicity, we sometimes write SDH assumption.*

3 Short Signature Scheme Without Random Oracles

3.1 The Proposed Scheme

There are many applications for digital signature schemes with small size [5]. Our new short signature scheme is a variant of BB04 scheme [4]. We highlight it here not only because this is an important building block for our UDVS scheme but also it may admit many other possible applications.

Let $(\mathbb{G}_1, \mathbb{G}_2)$ be bilinear groups where $|\mathbb{G}_1| = |\mathbb{G}_2| = p$ for some large prime p. m is the message to be signed and is encoded as an element of Z_p^*.

KG: Pick a random generator $g_2 \in \mathbb{G}_2$ and set $g_1 = \psi(g_2)$. Pick $x, y \leftarrow Z_p^*$, and compute $u \leftarrow g_2^x \in \mathbb{G}_2$ and $v \leftarrow g_2^y \in \mathbb{G}_2$. For fast verification, also compute $z \leftarrow e(g_1, g_2) \in \mathbb{G}_T$. The public key is (g_1, g_2, u, v, z) and the secret key is (x, y).

Sig: Given a secret key $(x, y) \in (Z_p^*)^2$ and a message $m \in Z_p^*$, pick $r \leftarrow Z_p^*$. If $x + r + ym = 0 \bmod p$, try again with a different random r. Compute $\sigma \leftarrow g_1^{1/(x+r+ym)} \in \mathbb{G}_1$. The signature is (σ, r).

Ver: Give the public key (g_1, g_2, u, v, z), a message $m \in Z_p^*$, and a signature (σ, r), verify if

$$e(\sigma, u \cdot g_2^r \cdot v^m) = z$$

Output `acc` if the equality holds; otherwise, `rej`.

The size of the signature is the same as that of the BB04 scheme and comparable to DSA. The key generation algorithm, the signing algorithm and the verification algorithm take exactly the same amount of time as those of BB04 scheme. One can also use pre-computation to speed up the online computation.

3.2 Security Analysis

The security of the short signature scheme in section 3.1 is guaranteed by following theorem:

Theorem 1. *If there exists a forging algorithm* $\mathcal{F}$ *that* (q_s, t, ϵ)*-breaks the strong existential unforgeability of the above signature scheme, one can build an algorithm that* (q, t', ϵ')*-breaks the SDH assumption, where* $q_s < q$, $t' \leq (t - \Theta(q^2 T))$ *and* $\epsilon' \geq 2(\epsilon - q_s/p)$.

The proof is given Appendix A.

Hash Variants. A standard argument shows that if a collision-free hash function is applied to m, whose output can be encoded as an element of Z_p^*, one can still prove the security against ($\mathsf{sEUF\text{-}CMA}$) of the resulting signature scheme without random oracles. On the other hand, if we assume random oracles, we can have much efficient schemes.

4 Model of UDVS

Syntax of UDVS. A universal designated verifier signature (UDVS) scheme $\mathcal{UDVS} = (\mathsf{CPG}, \mathsf{SKG}, \mathsf{VKG}, \mathsf{S}, \mathsf{PV}, \mathsf{DS}, \mathsf{DV}, \mathsf{P_{KR}})$.

1. **Common Parameter Generation** CPG – a probabilistic algorithm, given a security parameter k, outputs a string cp consisting of common scheme parameters (publicly shared by all users).
2. **Signer Key Generation** SKG – a probabilistic algorithm, on input a common parameter string cp, outputs a secret/public key-pair $(sk_\mathsf{s}, pk_\mathsf{s})$ for *Signer*.
3. **Verifier Key Generation** VKG – a probabilistic algorithm, on input a common parameter string cp, outputs a secret/public key-pair $(sk_\mathsf{v}, pk_\mathsf{v})$ for *Verifier*.
4. **Signing** S – possibly a probabilistic algorithm, on input Signer's secret key sk_s and a message m, outputs Signer's public verifiable (PV) signature s.
5. **Public Verification** PV – a deterministic algorithm, on input Signer's public key pk_s and message/PV-signature pair (m, s), outputs verification result $d \in (\mathtt{acc}, \mathtt{rej})$.
6. **Designation** DS – possibly a probabilistic algorithm, on input Signer's public key pk_s, Verifier's public key pk_v, and a message/PV-signature pair (m, s), outputs Designated-Verifier (DV) signature $\widehat{s}$.
7. **Designated Verification** DV – a deterministic algorithm, on input Signer's public key pk_s, Verifier's secret key sk_v, and messge/DV-signature pair $(m, \widehat{s})$, outputs verification decision acc or rej.
8. **Verifier Key-Registration** $\mathsf{P_{KR}}(\mathsf{KR}, \mathsf{V})$ – a protocol between a Key Registration Authority KR and a Verifier V. The verifier registers a verifier's public key. On common input cp, KR and V interact with messages sent each other. At the end of the protocol, KR outputs a pair $(pk_\mathsf{v}, Auth)$, where pk_v is the public key of V and $Auth \in \{\mathtt{acc}, \mathtt{rej}\}$ indicates whether or not the key-registration is successful.

Note that the key registration is necessary in practice because the signature receiver can easily convince a third party the validity of a signature by claiming he is the owner of the third party's public key and presents the UDVS to the third party.

4.1 Enhanced Security Notions

Strong DV-Unforgeability. Without designation, a UDVS scheme reduces to a normal signature scheme. There are two types of unforgeability to consider:

Public Verifiable signature unforgeability (PV-unforgeability), the security for the signer, which states that anyone should not be able to forge a PV-signature of the signer. Designated Verifier signature unforgeability (DV-unforgeability), the security for the designated verifier, which states that for any message, an adversary without a PV-signature should be unable to convince a designated verifier of holding such a PV-signature. DV-unforgeability always implies PV-unforgeability, because anyone able to forge a PV-signature can transform it into a DV-signature. Thus it is enough to consider only DV-unforgeability.

We consider strong form of existential unforgeability in the sense of [1]. In our model, a forger $\mathcal{F}$ succeeds in breaking the scheme if it can outputs a valid new pair of message/DV-signature, whereas in [16], a forger $\mathcal{F}$ is considered to succeed only if it is able to forge the DV-signature on a "new message".

We call this security definition *strong existential unforgeability for designated verifier against adaptive chosen public key and chosen message attack* (sEUF-DV-CPKMA). We define it via the following game:

Definition 3. *Let* $\mathcal{UDVS} = (\mathsf{CPG}, \mathsf{SKG}, \mathsf{VKG}, \mathsf{S}, \mathsf{PV}, \mathsf{DS}, \mathsf{DV}, \mathsf{P_{KR}})$ *be a UDVS scheme.* $\mathcal{F}$ *is a forger attacking the DV-unforgeability of* $\mathcal{UDVS}$ *that plays the following game with a challenger.*

Setup. *The key generation algorithms are run,* $cp \leftarrow \mathsf{CPG}(1^k)$, $(pk_{\mathsf{s}}, sk_{\mathsf{s}}) \leftarrow \mathsf{SKG}(cp)$, $(pk_{\mathsf{v}_i}, sk_{\mathsf{v}_i}) \leftarrow \mathsf{PKG}_i(cp)$ *for* $1 \leq i \leq n$. *All public keys are given to* $\mathcal{F}$ *and the challenger. All secret keys are given to the challenger. The challenger maintains two lists:* $\mathbf{M}$ *and* $\widehat{\mathbf{S}}$, *initially empty. The challenger additionally maintains a list* $\mathbf{L}$ *which consists of all the public keys* $\{pk_{\mathsf{v}_i}\}$ *for* $1 \leq i \leq n$, *which are assumed to be already registered.*

Training. $\mathcal{F}$ *may adaptively issue* q_s *times signing queries,* q_d *times designation queries (see below),* q_v *times Designated Verification queries, and up to* $n-1$ *times of key registration queries to the challenger. However, once* pk_{v_i} *is queried by* $\mathcal{F}$, *the challenger neglects further designated verification quires with respect to verifier* V_i*'s public key* pk_{v_i}.

- *On a Signing query by* $\mathcal{F}$ *on* m, *the challenger returns the corresponding signature* $s = \mathsf{S}(sk_{\mathsf{s}}, m)$ *to* $\mathcal{F}$ *and adds* m *to* $\mathbf{M}$.
- *On Designation query by* $\mathcal{F}$ *on* m *and* pk_{v}, *the challenger first computes the corresponding PV-signature* $s = \mathsf{S}(sk_{\mathsf{s}}, m)$, *compute the* $\widehat{s} = \mathsf{DS}(pk_{\mathsf{s}}, pk_{\mathsf{v}}, m, s)$. *The challenger then adds* $(m, \widehat{s})$ *to* $\widehat{\mathbf{S}}$, *and returns* $\widehat{s}$ *to* $\mathcal{F}$.
- *On a Designated Verification oracle query by* $\mathcal{F}$ *on* $(m, \widehat{s})$ *on* pk_{v_i}, *the challenger runs the designated verification algorithm* $\mathsf{DV}(pk_{\mathsf{s}}, sk_{\mathsf{v}_i}, m, \widehat{s})$ *and returns the corresponding verification result to* $\mathcal{F}$.
- *On a Key Registration query by* $\mathcal{F}$ *on* pk_{v}, *the challenger sends corresponding secret key* sk_{v} *to* $\mathcal{F}$, *and deletes* pk_{v} *from* $\mathbf{L}$.

Forge. *Denote* $\mathcal{F}$*'s running time as* t. $\mathcal{F}$ *outputs* $(m^*, \widehat{s^*})$ *and wins the game if:* $\mathsf{DV}(pk_{\mathsf{s}}, sk_{\mathsf{v}^*}, m^*, \widehat{s^*}) = \mathtt{acc}$ *with* $(m^*, \widehat{s^*}) \notin \widehat{\mathbf{S}}$, *where* $pk_{\mathsf{v}^*} \in \mathbf{L}$, $m^* \in \mathcal{M}$ *is a valid message from the message space.*

Suppose q_s, q_d, q_v and t are all polynomially bounded. A UDVS signature is secure against sEUF-DV-CPKMA *if no any probabilistic polynomial time $\mathcal{F}$, can win the above game with non-negligible probability. The probability is taken over coin toss of key generation algorithms, $\mathcal{F}$ and the challenger.*

An important refinement to [15, 16] is that we allow adversaries to adaptively corrupt designated verifiers and adaptively choose the target designated verifier, which reflects more essence of real world adversaries. Moreover, our DV-unforgeability is defined in the sense of strong unforgeability.

One can weaken this definition with minor changes on the requirement of the forgery: for $pk_{\mathsf{v}^*} \in \mathbf{L}$, $\mathcal{F}$ wins the game if $\mathcal{F}$ outputs $\mathsf{DV}(pk_{\mathsf{s}}, sk_{\mathsf{v}^*}, m^*, \widehat{s^*}) = \mathtt{acc}$, with $m^* \notin \mathbf{M}$ and $(m^*, \cdot) \notin \widehat{\mathbf{S}}$. We call the resulting definition *weak existential unforgeability against chosen public key and chosen message attack* (wEUF-DV-CPKMA). We note that even this weaker definition guarantees stronger security than those considered in [15, 16].

Remark 1. We allow the adversary to make designation queries which captures the attack scenario where a real world adversary may obtain $(m, \widehat{s})$ without knowing corresponding PV-signature s. Though in previous models the adversary is not allowed to make designation queries, it turns out that previous model suffice in some sense when weak existential unforgeability of DV-signatures is considered, because given the signing oracle access, the adversary can simulate the designation oracle anyway. But we emphasize that our modeling approach is more general.

Non-transferability (Privacy Notion)

Definition 4. *$\mathcal{A}$ is an attacker that tries to brag about its interaction with the signature holder. $\mathcal{S}$ is a simulator that simulates the output of $\mathcal{A}$. $\mathcal{S}$ is able to access $\mathcal{A}$ as a black-box. $\mathcal{D}$ is a distinguisher that tries to distinguish whether a given output is of $\mathcal{A}$ or of $\mathcal{S}$. The run time of $\mathcal{S}$ does not include the run time of $\mathcal{A}$ that was black-box accessed by $\mathcal{S}$.*

Setup. $cp \leftarrow \mathsf{CPG}(1^k)$. $(pk_{\mathsf{s}}, sk_{\mathsf{s}}) \leftarrow \mathsf{SKG}(cp)$. KR *is a Key Registration oracle, who maintains a list of verifier's public keys, initially empty.*

Training. *$\mathcal{A}$ and $\mathcal{S}$ are allowed to have the following resources:*

- *$\mathcal{A}$ and $\mathcal{S}$ are allowed to access Signing oracle $\mathsf{S}(sk_{\mathsf{s}}, \cdot)$ up to q_s times and q'_s times, respectively. However, after the challenge message m^* is output, they may not access to Signing oracle S with respect to this challenge message.*
- *$\mathcal{S}$ and $\mathcal{A}$ can output the challenge message m^* at arbitrary time but for only once.*
- *$\mathcal{A}$ and $\mathcal{S}$ are allowed to access KR up to q_k and q'_k times, respectively.*
- *$\mathcal{A}$ and $\mathcal{S}$ are allowed to access $\mathcal{D}$ up to q_c and q'_c times, respectively.*
- *$\mathcal{A}$ is allowed to access to designation oracles $\mathsf{DS}(pk_{\mathsf{v}_i}, \cdot)$ up to q_d times as long as pk_{v_i} is correctly registered.*
- *$\mathcal{S}$ is NOT allowed to access DS.*

Guess. *Denote the running time of $\mathcal{A}$, $\mathcal{S}$ are t, t', respectively. Finally, $\mathcal{A}$ and $\mathcal{S}$ return to $\mathcal{D}$ their outputs with respect to m^*. $\mathcal{D}$ decides whether this output is of $\mathcal{A}$ or of $\mathcal{D}$.*

We say a UDVS scheme is unconditionally non-transferable against adaptive chosen public key attack and chosen message attack *(*$\mathsf{NT\text{-}CPKMA}$*), if there exists $\mathcal{S}$ such that for every $\mathcal{A}$, every computationally unbounded $\mathcal{D}$ distinguishes outputs of $\mathcal{A}$ and $\mathcal{S}$ on any challenge message m^* with only probability* $\mathsf{negl}(k)$*, where the probability is taken over the coin toss of key generation algorithms,* S*, $\mathcal{A}$, $\mathcal{S}$ and $\mathcal{D}$.*

Our definition on non-transferability extends [16] in the sense that $\mathcal{A}$ is able to access to Designation oracle with respect to any message (including the challenge message) before the challenge message is determined. This helps the adversary adaptively choose the challenge message.

5 UDVS Without Random Oracles

5.1 The Proposed Scheme

We give our scheme below. For simplicity, we omit the verifier key registration protocol. As in practice this is needed to run only once and may be executed interactively using zero-knowledge proof of knowledge of his secret key to KR or let the verifier send his secret key to the KR directly.

1. **Common Parameter Generation** CPG: Choose a bilinear group pair which is denoted by a description string Str_D: $(\mathbb{G}_1, \mathbb{G}_2)$ of prime order $|\mathbb{G}_1| = |\mathbb{G}_2| = p$ with a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ and an isomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$. Choose a random generator $g_2 \in \mathbb{G}_2$ and compute $g_1 = \psi(g_2) \in \mathbb{G}_1$. Then the common parameter is $cp = (Str_D, g_1, g_2)$.
2. **Signer Key Generation** SKG: Given cp, pick random $x_1, y_1 \leftarrow Z_p^*$, compute $u_1 = g_2^{x_1}$ and $v_1 = g_2^{y_1}$. Specially, for speeding up the verification, one may also compute $z \leftarrow e(g_1, g_2) \in \mathbb{G}_T$. The public key is $pk_{\mathsf{s}} = (cp, u_1, v_1, z)$, the secret key is $sk_{\mathsf{s}} = (x_1, y_1)$.
3. **Verifier Key Generation** VKG: Given cp, pick random $x_3, y_3 \leftarrow Z_p^*$. Compute $u_3 = g_2^{x_3}$ and $v_3 = g_2^{y_3}$. The public key is $pk_{\mathsf{v}} = (cp, u_3, v_3)$ and the secret key is $sk_{\mathsf{v}} = (x_3, y_3)$.
4. **Signing** S: Given the signer's secret key (cp, x_1, y_1) and a message m, select $r \leftarrow Z_p^*$. If $x_1 + r + my_1 = 0 \bmod p$, restart. Compute $\sigma = g_1^{1/(x_1+r+my_1)}$ and output $s = (\sigma, r)$ as the PV-signature.
5. **Public Verification** PV: Given the signer's public key (cp, u_1, v_1, z), and a message/PV-signature pair (m, s), output `acc` only if $e(\sigma, u_1 \cdot g_2^r \cdot v_1^m) = z$; otherwise output `rej`.
6. **Designation** DS: Given the signer's public key (cp, u_1, v_1), a verifier's public key (cp, u_3, v_3) and a message/PV-signature pair (m, s), where $s = (\sigma, r)$, let $h = g_2^r$ and compute $d = e(\psi(u_3), v_3^r) \in \mathbb{G}_T$. Then the DV-signature is $\widehat{s} = (\sigma, h, d)$.

7. **Designated Verification** DV: Given a signer's public key (cp, u_1, v_1), a verifier's secret key (x_3, y_3), and message/DV-signature pair $(m, \widehat{s})$, output `acc` only if following two equations hold simultaneously:

$$z = e(\sigma, u_1 \cdot h \cdot v_1^m) \qquad \text{and} \qquad d = e(\psi(u_3), h^{y_3})$$

Otherwise, output `rej`.

Remark 2. In fact, the public key u_3 of the designated verifier can be replaced by any *certified* random generator of group $\mathbb{G}_1$. However, we consider scenarios where a designated verifier may become a signer himself, who in turn needs to have two public keys.

5.2 Security Analysis

The correctness of the scheme is obvious. The following two theorems guarantee that the proposed scheme are both strong existentially unforgeable against adaptive chosen public key attack and chose message attack for designated verifier and unconditionally non-transferable.

Theorem 2 (Strong Unforgeability). *Above UDVS scheme achieves* sEUF-CPKMA*-security, provided that SDH assumption and the following assumption hold in bilinear groups.*

Assumption 1 (Knowledge of Exponent Assumption [2, 11]) *Suppose that an adversary $\mathcal{A}$ is given a pair (g, h) which is randomly chosen from uniform distribution of $\mathbb{G}^2$ and that $\mathcal{A}$ is able to generate a pair $(x, y) \in \mathbb{G}^2$ where $\log_g h = \log_x y$, then there exists an extractor that extracts $\log_g h$ for $\mathcal{A}$.*

In fact, one can easily prove that an extended version of above assumption holds in generic bilinear groups, namely, for $(g, h) \in_R {\mathbb{G}_2}^2$ and $(f_1, f_2) \in_R \mathbb{G}_1 \times \mathbb{G}_2$, $\mathcal{A}$ generates $y \in \mathbb{G}_T$, such that $\log_g h = \log_{e(f_1, f_2)} y$, then there exists an extractor that extracts $\log_g h$ for $\mathcal{A}$.

Proof. Suppose that there exists an adversary $\mathcal{A}$ that breaks sEUF-CPKMA of the scheme and (σ, h, d) is the DV-signature that $\mathcal{A}$ forged. Then, from the knowledge of exponent assumption, there exists an extractor that is able to extract r from $\mathcal{A}$ such that $h = g_2^r$ holds and the resulting (σ, r) is a successful forgery of the proposed short signature scheme. Therefore, combined with Theorem 1, it is easy to see the proposed UDVS scheme achieves sEUF-CPKMA.

We have introduced two types of existential unforgeabilities, namely weak EUF and strong EUF, though usually it is considered wEUF is enough for most of the applications. We emphasize that even adopting a probabilistic signing algorithm, one can still achieve strong DV-unforgeability.

Theorem 3 (Non-transferability). *Above UDVS scheme achieves unconditional* NT-CPKMA*-security.*

Proof. $\mathcal{A}$ and $\mathcal{S}$ are able to access Key Registration oracle KR and Signing oracle S. $\mathcal{A}$ is able to additionally access Designation oracle and Designated Verification oracle. $\mathcal{S}$ is able to access $\mathcal{A}$ as a black-box.

The following is how $\mathcal{S}$ simulate an output of $\mathcal{A}$. $\mathcal{S}$ invokes $\mathcal{A}$ by feeding it a random tape.

1. Suppose that $\mathcal{A}$ accesses to KR for a public key u'_3, v'_3. Then $\mathcal{S}$ interacts, as KR, with $\mathcal{A}$. If $\mathcal{A}$, as KR, accepts Key Registration, $\mathcal{S}$ rewinds $\mathcal{A}$ and obtains its corresponding secret key x'_3, y'_3.
2. Suppose that $\mathcal{A}$ accesses to the Signing oracle with respect to u'_1, v'_1 for message m'. $\mathcal{S}$ access to the Signing oracle with respect to u'_1, v'_1 for message m' and obtains signature σ'. Then $\mathcal{S}$ sends this σ' to $\mathcal{A}$.
3. Suppose that $\mathcal{A}$ accesses to the Designation oracle with respect to u'_1, v'_1 for message m' as an designated verifier whose public key is (u'_3, v'_3). (u'_3, v'_3) is the public key that $\mathcal{A}$ has once registered to KR. Recall that $\mathcal{S}$ is not allowed to query S on m^*, However, $\mathcal{S}$ can still generate DV-signature as follows:

$$s \in_R Z_p^* \qquad \sigma = g_2^s$$
$$h = g_2^{1/s} u^{-1} v^{-m} \qquad d = e(g_1, h)^{x'_3 y'_3}$$

4. Suppose that $\mathcal{A}$ outputs its output to distinguisher $\mathcal{D}$, $\mathcal{S}$ outputs it to $\mathcal{D}$.

Since $\mathcal{S}$ has perfectly simulated Signing oracle, KR, Designation oracle to $\mathcal{A}$, $\mathcal{S}$ finally obtains the output that is perfectly indistinguishable from the output of $\mathcal{A}$. The simulation of $\mathcal{S}$ is perfect. This completes the proof.

References

1. J.H. An, Y. Dodis, and T. Rabin. On the security of joint signature and encryption. In *Eurocrypt'02*, volume 2332 of *LNCS*, pages 83–107, Springer-Verlag, 2002.
2. M. Bellare and A. Palacio. The Knowledge-of-Exponent Assumptions and 3-Round Zero-Knowledge Protocols. In *CRYPTO '04*, volume 3152. Springer-Verlag, 2004.
3. M. Bellare and P. Rogaway. Random Oracles are Practical: A Paradigm for Designing Efficient Protocols. In *ACM Conference on Computer and Communication Security*, volume 62-73, 1993.
4. D. Boneh and X. Boyen. Short Signatures Without Random Oracles. In *Eurocrypt'04*, volume 3027 of *LNCS*, pages 56–73. Springer-Verlag, 2004.
5. D. Boneh, B. Lynn, and H. Shacham. Short Signatures from the Weil Pairing. In *Asiacrypt'01*, volume 2248 of *LNCS*, pages 514–532. Springer-Verlag, 2001.
6. R. Canetti, O. Goldreich, and S. Halevi. The Random Oracle Methodology, Revisited. In *STOC'98*, pages 209–218. ACM Press, 1998.
7. R. Cramer and V. Shoup. Signature Schemes Based on the Strong RSA Assumption. *ACM Transactions on Information and System Security*, 3(3):161–185, 2000.
8. A. Fiat and A. Shamir. How to prove yourself: practical solutions to identification and signature problem. In *Crypto'86*, volume 263 of *LNCS*, pages 186–194. Springer-Verlag, 1987.

9. R. Gennaro, S. Halevi, and T. Rabin. Secure Hash-and-Sign Signatures without the Random Oracle. In *Eurocrypt'99*, volume 1592 of *LNCS*, pages 123–139. Springer-Verlag, 1999.
10. S. Goldwasser, S. Micali, and R.L. Rivest. A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks. *SIAM J. Comput.*, 17(2):281–308, 1988.
11. S. Hada and T. Tanaka. On the Existence of 3-round Zero-Knowledge Protocols. In *CRYPTO '98*, LNCS, pages 408–423, 1998.
12. M. Jakobsson, K. Sako, and R. Impagliazzo. Designated Verifier Proofs and Their Applications. In *Eurocrypt'96*, volume 1070 of *LNCS*, pages 143–154. Springer-Verlag, 1996.
13. A. Joux and K. Nguyen. Separating Decisional Diffie-Hellman from Diffie-Hellman in Cryptographic Groups. Cryptology ePrint Archive, Report 2001/003, `http://eprint.iacr.org/2001/003/`, 2001.
14. S. Mitsunari, R. Sakai, and M. Kasahara. A New Trator Tracing. *IEICE Trans. Fundamentals*, E85A(2):481–484, 2002.
15. R. Steinfeld, L. Bull, H. Wang, and J. Pieprzyk. Universal designated-verifier signatures. In *Asiacrypt'03*, volume 2894 of *LNCS*, pages 523–542. Springer-Verlag, 2003.
16. R. Steinfeld, H. Wang, and J. Pieprzyk. Efficient Extension of Standard Schnorr/RSA Signatures into Universal Designated-Verifier Signatures. In *PKC'04*, volume 2947 of *LNCS*, pages 86–100. Springer-Verlag, 2004.

A Proof of Theorem 1

Actually, we do be able to give another proof in the full version of this paper, however, we believe the proof adopted here is much easier to understand.

Theorem 1 is proved via following two lemmas. The first lemma, actually proves a basic scheme is existentially unforgeable against *weak chosen message attack*. The second lemma then shows the security of the basic scheme implies the full scheme.

Weak Chosen Message Attack (wCMA). First, the adversary outputs a list of queries q_s messages $m_1, ..., m_{q_s} \in \mathcal{M}$. Then KG is run, generating the public/secret key pair (pk, sk). pk is given to the adversary $\mathcal{F}$ and sk is given to the Challenger. The Challenger sends $\mathcal{A}$ signatures $s_i = \mathsf{Sig}(m_i)$ for $i = 1, ..., q_s$. At last, $\mathcal{A}$ outputs a pair (m^*, s^*), and wins the game if $m^* \notin \{m_1, ..., m_{q_s}\}$ and $\mathsf{Ver}(m^*, \sigma^*) = \mathtt{acc}$. Define $\mathsf{Adv}_{\mathcal{A}}$ to be the probability that $\mathcal{A}$ wins above game, taken over the coin toss of $\mathcal{A}$ and the Challenger. Then $\mathcal{A}$ (q_s, t, ϵ)-weakly breaks a signature scheme if $\mathcal{A}$ runs in time at most t, makes at most q_s Sig queries, and $\mathsf{Adv}_{\mathcal{A}}$ is at least ϵ. A signature scheme is (q_s, t, ϵ)-weakly existentially unforgeable under a weak chosen message attack if no PPT $\mathcal{A}$ (q_s, t, ϵ)-weakly breaks it.

Basic scheme Let $(\mathbb{G}_1, \mathbb{G}_2)$ be bilinear group pair where $|\mathbb{G}_1| = |\mathbb{G}_2| = p$ for some prime p. m is encoded as an element of Z_p^*.

KG: Pick a random generator $g_2 \in \mathbb{G}_2$ and set $g_1 = \psi(g_2)$. Pick $x \leftarrow Z_p^*$, and compute $u \leftarrow g_2^x \in \mathbb{G}_2$. For fast verification, also compute $z \leftarrow e(g_1, g_2) \in \mathbb{G}_T$. The public key is (g_1, g_2, u, z).

Sig: Given a secret key $x \in Z_p^*$ and a message $m \in Z_p^*$, compute $\sigma \leftarrow g_1^{1/(x+m)} \in \mathbb{G}_1$. Here $1/(x+m)$ is computed modulo p. If $x + m = 0$, we set $\sigma \leftarrow 1$. The signature is σ.

Ver: Give the public key (g_1, g_2, u, z), a message $m \in Z_p^*$, and a signature σ, verify if $e(\sigma, u \cdot g_2^m) = z$. If the equality holds, output `acc`. If σ=1 and $u \cdot g_2^m = 1$ also output `acc`. Otherwise, output `rej`.

Lemma 1. *Suppose (q, t', ϵ)-SDH assumption holds in $(\mathbb{G}_1, \mathbb{G}_2)$. The basic signature scheme above is (q_s, t, ϵ)-secure against existential forgery under a weak chosen message attack, where*

$$q_s < q \qquad \textit{and} \qquad t \leq t' - \Theta(q^2 T)$$

Proof. Assume $\mathcal{A}$ is a forger that (q_s, t, ϵ)-breaks the signature scheme. We construct an algorithm $\mathcal{B}$ that, by interacting with $\mathcal{A}$, solves the SDH problem in time t' with advantage ϵ. Algorithm $\mathcal{B}$ is given a random instance $(g_1, g_2, A_1, ..., A_q)$ of the SDH problem, where $A_i = g^{(x_i)} \in \mathbb{G}_2$ for $i = 1, ..., q$ and for some unknown $x \in Z_p^*$. For convenience we set $A_0 = g_2$. Algorithm $\mathcal{B}$'s goal is to produce a pair $(c, g_1^{1/(x+c)})$ for some $c \in Z_p^*$. Algorithm $\mathcal{B}$ does so by interacting with the forger $\mathcal{A}$ as follows:

Query. Algorithm $\mathcal{A}$ outputs a list of distinct q_s messages $m_1, ..., m_{q_s} \in Z_p^*$, where $q_s < q$. Since $\mathcal{A}$ must reveal its queries up front, we may assume that $\mathcal{A}$ outputs exactly $q - 1$ messages to be signed (if the actual number is less, we can always virtually reduce the value of q so that $q = q_s + 1$).

Response. $\mathcal{B}$ must respond with a public key and signatures on the $q - 1$ messages from $\mathcal{A}$. Let $f(y)$ be the polynomial $f(y) = \prod_{i=1}^{q-1}(y + m_i)$. Expand $f(y)$ and write $f(y) = \sum_{i=0}^{q-1} \alpha_i y^i$ where $\alpha_0, ..., \alpha_{q-1} \in Z_p^*$ are the coefficients of the polynomial $f(y)$. Compute:

$$g_2' \leftarrow \prod_{i=0}^{q-1} A_i^{\alpha_i} = g_2^{f(x)} \qquad \text{and} \qquad h \leftarrow \prod_{i=1}^{q} A_i^{\alpha_{i-1}} = g_2^{xf(x)} = {g_2'}^x$$

Also, let $g_1' = \psi(g_2')$. and $z' = e(g_1', g_2')$. The public key given to $\mathcal{A}$ is (g_1', g_2', h, z'), which has the correct distribution. Next, for each $i = 1, ..., q-1$, Algorithm $\mathcal{B}$ must generate a signature σ_i on m_i. To do so, let $f_i(y)$ be the polynomial $f_i(y) = f(y)/(y + m_i) = \prod_{j=1, j\neq i}^{q-1}(y + m_j)$. As before, we expand f_i and write $f_i(y) = \sum_{j=0}^{q-2} \beta_j y_j$. Compute

$$S_i \leftarrow \prod_{j=0}^{q-2} A_j^{\beta_j} = g_2^{f_i(x)} = {g_2'}^{1/(x+m_i)} \in \mathbb{G}_2$$

Observe that $\sigma_i = \psi(S_i) \in \mathbb{G}_1$ is a valid signature on m under the public key (g_1', g_2', h, z'). Algorithm $\mathcal{B}$ gives $\mathcal{A}$ the $q - 1$ signatures $\sigma_1, ..., \sigma_{q-1}$.

Output. Algorithm $\mathcal{A}$ returns a forgery (σ^*, m^*) such that $\sigma^* \in \mathbb{G}_1$ is a valid signature on $m \in Z_p^*$ and $m^* \notin \{m_1, ..., m_{q-1}\}$ since there is only one valid signature per message. In other words, $e(\sigma^*, h \cdot {g_2'}^{m^*}) = e(g_1', g_2')$. Since $h = {g_2'}^{x}$ we have that $e(\sigma^*, {g_2'}^{x+m^*}) = e(g_1', g_2')$ and therefore

$$\sigma^* = {g_1'}^{1/(x+m^*)} = g_1^{f(x)/(x+m^*)} \tag{1}$$

Using long division we write (1) the polynomial f as $f(y) = \gamma(y)(y + m^*) + \gamma_{-1}$ for some polynomial $\gamma(y) = \sum_{i=0}^{q-2} \gamma_i y^i$ and some $\gamma_{-1} \in Z_p^*$. Then the rational fraction $f(y)/(y+m^*)$ in the exponent on the right side of Equation (1) can be written as

$$f(y)/(y + m^*) = \frac{\gamma_{-1}}{y + m^*} + \sum_{i=0}^{q-2} \gamma_i y^i$$

hence

$$\sigma^* = g_1^{\frac{\gamma_{-1}}{y+m^*} + \sum_{i=0}^{q-2} \gamma_i y^i}.$$

Note that $\gamma_{-1} \neq 0$, since $f(y) = \prod_{i=1}^{q-1}(y + m_i)$ and $m^* \notin \{m_1, ..., m_{q-1}\}$, thus $(y + m^*)$ does not divide $f(y)$. Then algorithm $\mathcal{B}$ computes

$$w = \left(\sigma^* \cdot \prod_{i=0}^{q-2} \psi(A_i)^{-\gamma_i}\right)^{1/\gamma_{-1}} = g_1^{1/(x+m^*)}$$

and returns (m^*, w) as the solution to the SDH instance.

The claimed bounds are obvious by construction of the reduction.

Lemma 2. *Suppose the basic scheme of Lemma 1 is (q_s, t', ϵ)-weakly secure. The full signature scheme is (q_s, t, ϵ)-secure against existential forgery under a chosen message attack, where*

$$\epsilon \geq 2(\epsilon + q_s/p) \qquad \textit{and} \qquad t \leq t' - \Theta(q_s T)$$

Proof. We explain informally our strategy. As mentioned above, for a forger $\mathcal{A}$ that breaks the full scheme, a forger $\mathcal{B}$ the basic scheme can be built as follows. $\mathcal{B}$ sends a list of randomly chosen message $M_1, ..., M_{q_s}$ and queries to its challenger, which returns the corresponding signatures $\sigma_1, ..., \sigma_{q_s}$. Then $\mathcal{B}$ is given the public key $u(= g_2^x)$ for some $x \in Z_p^*$. $\mathcal{B}$ chooses random secret key $y \in Z_p^*$, computes the public key $v = g_2^y$ and completes the public key for the full scheme with u and v. For q_s chosen message queried m_i by $\mathcal{A}$, $\mathcal{B}$ could compute an $r_i \in Z_p^*$ and such that $r_i + y m_i = M_i$. Then (σ_i, r_i) is a valid signature on m_i. Eventually, when $\mathcal{A}$ outputs a forgery (m^*, σ^*, r^*), $\mathcal{B}$ "switches" this forgery to $(M^* = r^* + y m^*, \sigma^*)$, which is a valid forgery for the full scheme. Next we give the detailed analysis and lower bound $\mathcal{B}$'s success probability. Denote two types of forger $\mathcal{A}$ as:

Type-1 Forger which either makes query for $m = -x$, or outputs a forgery where $m \notin \{M_1, ..., M_{q_s}\}$.

Type-2 Forger which both never makes query for message $m = -x$, and outputs a forgery where $r^* + ym^* \in \{M_1, ..., M_{q_s}\}$.

We describe $\mathcal{B}$ as follows:

Setup. $\mathcal{B}$ queries its challenger with a list of messages $M_1, ..., M_{q_s} \in Z_p^*$. The challenger responds with a valid public key (g_1, g_2, u, z) and signatures $\sigma_1, ..., \sigma_{q_s} \in \mathbb{G}_1$ on these messages, where $e(\sigma_i, g_2^{M_i} u) = e(g_1, g_2) = z$ for $i = 1, ..., q_s$. $\mathcal{B}$ chooses a random $y \in Z_p^*$ and chooses a bit $b \in \{1, 2\}$ randomly. If $b_{mode} = 1$, $\mathcal{B}$ gives $\mathcal{A}$ the public key $PK_1 = (g_1, g_2, u, g_2^y, z)$. If $b_{mode} = 2$, $\mathcal{B}$ gives $\mathcal{A}$ the public key $PK_2 = (g_1, g_2, g_2^y, u, z)$. In $\mathcal{A}$'s view, both PK_1 and PK_2 are valid public keys for the full scheme (g_1, g_2, U, V, z).

Signature Query. In order to respond, $\mathcal{B}$ maintains a H-list of three data entries: (m_i, r_i, W_i) and a counter l initiated to 0. If receiving a signature for m, it increases l by 1, and checks if $l > q_s$. If $l > q_s$, it neglects further queries by $\mathcal{A}$ and terminate $\mathcal{A}$. Otherwise, it checks whether $g_2^{-m} = u$. If so, the $\mathcal{B}$ just obtains the private key of its challenger. It can forge any number of signatures for its target. In this case, it terminates successfully.

Otherwise, if $b_{mode} = 1$, set $r_l = M_i - ym \in Z_p^*$. If $r_l = 0$, $\mathcal{B}$ reports failure and aborts. Otherwise, $\mathcal{B}$ returns (σ_l, r_l) as answer. This is a valid signature on m for PK_1 because r_l is uniform in Z_p^* and

$$e(\sigma_l, U \cdot g_2^{r_l} \cdot V^m) = e(\sigma_l, u \cdot g_2^{M_i - ym} \cdot g_2^{ym}) = e(\sigma_l, u \cdot g_2^{M_l}) = e(g_1, g_2) = z$$

if $b_{mode} = 2$, set $r_l = mM_l - y \in Z_p^*$. If $r_l = 0$, $\mathcal{B}$ reports failure and aborts. Otherwise, $\mathcal{B}$ returns $(\sigma_l^{1/m}, r_l)$ as answer. This is a valid signature on m for PK_2 because r_l is uniform in Z_p^* and

$$e(\sigma_l^{1/m}, U \cdot g_2^r \cdot V^m) = e(\sigma_l^{1/m}, g_2^y \cdot g_2^{mM_l - y} \cdot u^m) = e(\sigma_l, u \cdot g_2^{M_l}) = e(g_1, g_2)$$

$\mathcal{B}$ further saves tuple $(m, r_l, g_2^{r_l} \cdot V^m)$ to H-list.

Output. Eventually, $\mathcal{A}$ returns a valid forgery (m^*, σ^*, r^*). Note that by adding dummy queries, we may assume that $\mathcal{A}$ makes exactly q_s queries. Let $W^* \leftarrow g_2^{m^*} V^{r^*}$. Then according to two types of forger $\mathcal{A}$, we denote the following events as

F1. No tuple of the form $(\cdot, \cdot, W)$ appears on the H-list, or $\mathcal{A}$ has queried on m such that $u = g^{-m}$.

F2. The H-list contains at least one tuple (m_j, r_j, W_j) such that $W_j = W^*$.

Denote $E1$ to be the event $b_{mode} = 1$ and denote $E2$ to be the event $b_{mode} = 2$. We claim that $\mathcal{B}$ can succeed in breaking the basic scheme if $(E1 \wedge F1) \vee (E2 \wedge F2)$ happens.

– (Case $E1 \wedge F1$). If $u = g^{-m}$, then $\mathcal{B}$ has already recovered the secret key of its challenger, $\mathcal{B}$ can forge signature on any message of his choice. Let $M^* = r + ym^*$, then from the definition of $F1$, (M^*, σ^*) is a valid message/signature pair and it is not queried to $\mathcal{B}$'s challenger before.

– (Case $E2 \wedge F2$). Since $V = u$, then we know that there exists a pair $g_2^{r_j} u^{m_j} = g_2^{r^*} u^{m^*}$. Since $(m^*, r^*) \neq (m_j, r_j)$, otherwise it is not regarded as a forgery, $m^* \neq m_j$ and $r^* \neq r_j$. Write $u = g_2^{\tau}$, $\mathcal{B}$ can compute $\tau = (r_j - r^*)/(m^* - m_j)$ which also enables $\mathcal{B}$ to recover the secret key of its challenger.

Since $E1$ and $F1$ are independent with uniform distribution, $Pr[E1 \vee E2] = 1$ and $Pr[F1 \vee F2] = 1$, the probability that $\mathcal{B}$ succeeds is

$$\Pr[E1 \wedge F1) \vee (E2 \wedge F2)] = 1/2$$

Then all left to do is lower-bounds $\mathcal{B}$'s abort probability. From above description of $\mathcal{B}$ we know that if $E1 \wedge F1$ happens, $\mathcal{B}$ aborts only if $r_l = 0$, i.e., $m_l = M_l$, this happens with probability at most q_s/p. If $E2 \wedge F2$ happens, $\mathcal{B}$ does not abort. Then $\mathcal{B}$ succeeds with probability at least $\epsilon/2 - q_s/p$. This completes the proof.