Asymptotically Efficient Lattice-Based Digital Signatures [TCC 2008]

Vladimir Lyubashevsky Daniele Micciancio

M. Tibouchi, Lattice-Based Crypto Mini-Group, 2009-10-14

Outline

Outline

Efficiency Gap of Digital Signatures

- As has been long known, secure digital signatures exist based on one-way functions, just like MACs and secret-key encryption schemes.
- However, while symmetric cryptographic constructs are expected to run in time linear in the security parameter k, usual signature schemes have complexity at least $\Omega(k^2)$.

Efficiency Gap of Digital Signatures

- As has been long known, secure digital signatures exist based on one-way functions, just like MACs and secret-key encryption schemes.
- However, while symmetric cryptographic constructs are expected to run in time linear in the security parameter k, usual signature schemes have complexity at least $\Omega(k^2)$.

Outline

Lamport-Diffie One-time Signatures

- Let $f : Y \to Z$ be a one-way function. Lamport proposed the following signature scheme.
  - KeyGen(1^k): for $1 \leq i \leq k$, $j = 0, 1$, choose $y_{i,j} \in Y$ randomly, and let $z_{i,j} = f(y_{i,j})$. Then $\mathrm{sk} = (y_{i,j})$, $\mathrm{pk} = (z_{i,j})$.
  - Sign($m \in \{0,1\}^k$): if $m = (m_1, \ldots, m_k)$, the signature is $s = (y_{1,m_1}, \ldots, y_{k,m_k})$.
  - Verify($m \in \{0,1\}^k, s \in Y^k$): if $s = (s_1, \ldots, s_k)$, accept if and only if $f(s_i) = z_{i,m_i}$ for all i.
- This is a one-time secure signature scheme: an adversary who obtains a signature on any one message of his choice cannot forge a signature on another message. Each key pair can be used only once.
- Verification requires k applications of function f: complexity at least $\Omega(k^2)$.

Lamport-Diffie One-time Signatures

- Let $f : Y \to Z$ be a one-way function. Lamport proposed the following signature scheme.
  - KeyGen(1^k): for $1 \leq i \leq k$, $j = 0, 1$, choose $y_{i,j} \in Y$ randomly, and let $z_{i,j} = f(y_{i,j})$. Then $\mathrm{sk} = (y_{i,j})$, $\mathrm{pk} = (z_{i,j})$.
  - Sign($m \in \{0,1\}^k$): if $m = (m_1, \ldots, m_k)$, the signature is $s = (y_{1,m_1}, \ldots, y_{k,m_k})$.
  - Verify($m \in \{0,1\}^k, s \in Y^k$): if $s = (s_1, \ldots, s_k)$, accept if and only if $f(s_i) = z_{i,m_i}$ for all i.
- This is a one-time secure signature scheme: an adversary who obtains a signature on any one message of his choice cannot forge a signature on another message. Each key pair can be used only once.
- Verification requires k applications of function f: complexity at least $\Omega(k^2)$.

Lamport-Diffie One-time Signatures

- Let $f : Y \to Z$ be a one-way function. Lamport proposed the following signature scheme.
  - KeyGen(1^k): for $1 \le i \le k$, $j = 0, 1$, choose $y_{i,j} \in Y$ randomly, and let $z_{i,j} = f(y_{i,j})$. Then $\mathsf{sk} = (y_{i,j})$, $\mathsf{pk} = (z_{i,j})$.
  - Sign($m \in \{0,1\}^k$): if $m = (m_1, \ldots, m_k)$, the signature is $s = (y_{1,m_1}, \ldots, y_{k,m_k})$.
  - Verify($m \in \{0,1\}^k, s \in Y^k$): if $s = (s_1, \ldots, s_k)$, accept if and only if $f(s_i) = z_{i,m_i}$ for all i.
- This is a one-time secure signature scheme: an adversary who obtains a signature on any one message of his choice cannot forge a signature on another message. Each key pair can be used only once.
- Verification requires k applications of function f; complexity at least $\Omega(k^2)$.

Lamport-Diffie One-time Signatures

- Let $f : Y \rightarrow Z$ be a one-way function. Lamport proposed the following signature scheme.
  - KeyGen(1^k): for $1 \leq i \leq k$, $j = 0, 1$, choose $y_{i,j} \in Y$ randomly, and let $z_{i,j} = f(y_{i,j})$. Then $\mathsf{sk} = (y_{i,j})$, $\mathsf{pk} = (z_{i,j})$.
  - Sign($m \in \{0,1\}^k$): if $m = (m_1, \ldots, m_k)$, the signature is $s = (y_{1,m_1}, \ldots, y_{k,m_k})$.
  - Verify($m \in \{0,1\}^k, s \in Y^k$): if $s = (s_1, \ldots, s_k)$, accept if and only if $f(s_i) = z_{i,m_i}$ for all i.
- This is a one-time secure signature scheme: an adversary who obtains a signature on any one message of his choice cannot forge a signature on another message. Each key pair can be used only once.
- Verification requires k applications of function f: complexity at least $\Omega(k^2)$.

Lamport-Diffie One-time Signatures

- Let $f : Y \to Z$ be a one-way function. Lamport proposed the following signature scheme.
  - KeyGen(1^k): for $1 \leq i \leq k$, $j = 0, 1$, choose $y_{i,j} \in Y$ randomly, and let $z_{i,j} = f(y_{i,j})$. Then $\mathrm{sk} = (y_{i,j})$, $\mathrm{pk} = (z_{i,j})$.
  - Sign($m \in \{0,1\}^k$): if $m = (m_1, \ldots, m_k)$, the signature is $s = (y_{1,m_1}, \ldots, y_{k,m_k})$.
  - Verify($m \in \{0,1\}^k, s \in Y^k$): if $s = (s_1, \ldots, s_k)$, accept if and only if $f(s_i) = z_{i,m_i}$ for all i.
- This is a one-time secure signature scheme: an adversary who obtains a signature on any one message of his choice cannot forge a signature on another message. Each key pair can be used only once.
- Verification requires k applications of function f_k: complexity at least $\Omega(k^2)$.

Lamport-Diffie One-time Signatures

- Let $f : Y \rightarrow Z$ be a one-way function. Lamport proposed the following signature scheme.
  - KeyGen(1^k): for $1 \leq i \leq k$, $j = 0, 1$, choose $y_{i,j} \in Y$ randomly, and let $z_{i,j} = f(y_{i,j})$. Then $\text{sk} = (y_{i,j})$, $\text{pk} = (z_{i,j})$.
  - Sign($m \in \{0,1\}^k$): if $m = (m_1, \ldots, m_k)$, the signature is $s = (y_{1,m_1}, \ldots, y_{k,m_k})$.
  - Verify($m \in \{0,1\}^k, s \in Y^k$): if $s = (s_1, \ldots, s_k)$, accept if and only if $f(s_i) = z_{i,m_i}$ for all i.
- This is a one-time secure signature scheme: an adversary who obtains a signature on any one message of his choice cannot forge a signature on another message. Each key pair can be used only once.
- Verification requires k applications of function f_k: complexity at least $\Omega(k^2)$.

Lamport-Diffie One-time Signatures

- Let $f : Y \to Z$ be a one-way function. Lamport proposed the following signature scheme.
  - KeyGen(1^k): for $1 \leq i \leq k$, $j = 0, 1$, choose $y_{i,j} \in Y$ randomly, and let $z_{i,j} = f(y_{i,j})$. Then $\mathsf{sk} = (y_{i,j})$, $\mathsf{pk} = (z_{i,j})$.
  - Sign($m \in \{0,1\}^k$): if $m = (m_1, \ldots, m_k)$, the signature is $s = (y_{1,m_1}, \ldots, y_{k,m_k})$.
  - Verify($m \in \{0,1\}^k, s \in Y^k$): if $s = (s_1, \ldots, s_k)$, accept if and only if $f(s_i) = z_{i,m_i}$ for all i.
- This is a one-time secure signature scheme: an adversary who obtains a signature on any one message of his choice cannot forge a signature on another message. Each key pair can be used only once.
- Verification requires k applications of function f_k: complexity at least $\Omega(k^2)$.

Merkle Trees

- Merkle proposed a way to turn a one-time secure signature scheme into a secure (stateful) 2^h-time signature scheme.
- Idea: use 2^h different key pairs.

Merkle Trees

- Merkle proposed a way to turn a one-time secure signature scheme into a secure (stateful) 2^h-time signature scheme.
- Idea: use 2^h different key pairs. The secret key used to sign the i-th message can be chosen as $\mathrm{sk}_i = \mathrm{PRNG}_K(i)$, where PRNG is a pseudo-random number generator.
- The signer constructs a hash tree from the public keys pk_i and publishes the root. When signing a message, she gives the verifier the path to the root and the adjacent nodes to authenticate the corresponding public key.
- The resulting scheme is 2^h-time secure, provided that the hash function used in constructing the tree is collision resistant.
- Variants of this construction can be used to build secure signature schemes for messages of arbitrary length based on any one-time signature scheme.

Merkle Trees

- Merkle proposed a way to turn a one-time secure signature scheme into a secure (stateful) 2^h-time signature scheme.
- Idea: use 2^h different key pairs. The secret key used to sign the i-th message can be chosen as $\mathrm{sk}_i = \mathrm{PRNG}_K(i)$, where PRNG is a pseudo-random number generator.
- The signer constructs a hash tree from the public keys pk_i and publishes the root. When signing a message, she gives the verifier the path to the root and the adjacent nodes to authenticate the corresponding public key.
- The resulting scheme is 2^h-time secure, provided that the hash function used in constructing the tree is collision resistant.
- Variants of this construction can be used to build secure signature schemes for messages of arbitrary length based on any one-time signature scheme.

Merkle Trees

- Merkle proposed a way to turn a one-time secure signature scheme into a secure (stateful) 2^h-time signature scheme.
- Idea: use 2^h different key pairs. The secret key used to sign the i-th message can be chosen as $\mathsf{sk}_i = \mathsf{PRNG}_K(i)$, where PRNG is a pseudo-random number generator.
- The signer constructs a hash tree from the public keys pk_i and publishes the root. When signing a message, she gives the verifier the path to the root and the adjacent nodes to authenticate the corresponding public key.
- The resulting scheme is 2^h-time secure, provided that the hash function used in constructing the tree is collision resistant.
- Variants of this construction can be used to build secure signature schemes for messages of arbitrary length based on any one-time signature scheme.

Merkle Trees

- Merkle proposed a way to turn a one-time secure signature scheme into a secure (stateful) 2^h-time signature scheme.
- Idea: use 2^h different key pairs. The secret key used to sign the i-th message can be chosen as $\mathsf{sk}_i = \mathsf{PRNG}_K(i)$, where PRNG is a pseudo-random number generator.
- The signer constructs a hash tree from the public keys pk_i and publishes the root. When signing a message, she gives the verifier the path to the root and the adjacent nodes to authenticate the corresponding public key.
- The resulting scheme is 2^h-time secure, provided that the hash function used in constructing the tree is collision resistant.
- Variants of this construction can be used to build secure signature schemes for messages of arbitrary length based on any one-time signature scheme.

Merkle Trees

- Merkle proposed a way to turn a one-time secure signature scheme into a secure (stateful) 2^h-time signature scheme.
- Idea: use 2^h different key pairs. The secret key used to sign the i-th message can be chosen as $\mathrm{sk}_i = \mathrm{PRNG}_K(i)$, where PRNG is a pseudo-random number generator.
- The signer constructs a hash tree from the public keys pk_i and publishes the root. When signing a message, she gives the verifier the path to the root and the adjacent nodes to authenticate the corresponding public key.
- The resulting scheme is 2^h-time secure, provided that the hash function used in constructing the tree is collision resistant.
- Variants of this construction can be used to build secure signature schemes for messages of arbitrary length based on any one-time signature scheme.

Merkle Trees

- Merkle proposed a way to turn a one-time secure signature scheme into a secure (stateful) 2^h-time signature scheme.
- Idea: use 2^h different key pairs. The secret key used to sign the i-th message can be chosen as $\mathsf{sk}_i = \mathsf{PRNG}_K(i)$, where PRNG is a pseudo-random number generator.
- The signer constructs a hash tree from the public keys pk_i and publishes the root. When signing a message, she gives the verifier the path to the root and the adjacent nodes to authenticate the corresponding public key.
- The resulting scheme is 2^h-time secure, provided that the hash function used in constructing the tree is collision resistant.
- Variants of this construction can be used to build secure signature schemes for messages of arbitrary length based on any one-time signature scheme.

Outline

Main result

There exists a signature scheme such that the signature of an n-bit message is of length $\tilde{O}(k)$, and both signature and verification take time $\tilde{O}(k) + \tilde{O}(n)$.
The scheme is strongly unforgeable under chosen-message attack assuming that approximating SVP in ideal lattices of dimension k up to a factor $\tilde{O}(k^2)$ is hard in the worst case.
Remarks:

- Asymptotically, the scheme is optimally efficient up to polylogaritmic factors.
- It is not secure for practical parameter sizes.
- Lyubashevsky and Micciancio actually construct an efficient one-time signature scheme. The existence of a signature scheme follows, using efficient implementations of Merkle trees.

Main result

There exists a signature scheme such that the signature of an n-bit message is of length $\tilde{O}(k)$, and both signature and verification take time $\tilde{O}(k) + \tilde{O}(n)$.
The scheme is strongly unforgeable under chosen-message attack assuming that approximating SVP in ideal lattices of dimension k up to a factor $\tilde{O}(k^2)$ is hard in the worst case.
Remarks:

- Asymptotically, the scheme is optimally efficient up to polylogaritmic factors.
- It is not secure for practical parameter sizes.
- Lyubashevsky and Micciancio actually construct an efficient one-time signature scheme. The existence of a signature scheme follows, using efficient implementations of Merkle trees.

Main result

There exists a signature scheme such that the signature of an n-bit message is of length $\tilde{O}(k)$, and both signature and verification take time $\tilde{O}(k) + \tilde{O}(n)$.
The scheme is strongly unforgeable under chosen-message attack assuming that approximating SVP in ideal lattices of dimension k up to a factor $\tilde{O}(k^2)$ is hard in the worst case.
Remarks:

- Asymptotically, the scheme is optimally efficient up to polylogaritmic factors.
- It is not secure for practical parameter sizes.
- Lyubashevsky and Micciancio actually construct an efficient one-time signature scheme. The existence of a signature scheme follows, using efficient implementations of Merkle trees.

Main elements of the construction

- Messages are small elements $\mathbf{z}$ in a ring $R = \mathbb{Z}_p[x]/\langle f \rangle$, where f is a unitary polynomial of degree n, irreducible over $\mathbb{Z}$ (and $p \sim C \cdot n^3$ is not necessarily prime).
- The secret key is a pair of short vectors $(\hat{\mathbf{k}}, \hat{\mathbf{l}})$ in R^m ($m \sim \log_2 n$), chosen according to an appropriate distribution.
- The public key is $(h, h(\hat{\mathbf{k}}), h(\hat{\mathbf{l}}))$ where h is a random hash function of the form:

$$h(x_1, \ldots, x_m) = a_1 x_1 + \cdots + a_m x_m$$

For a random choice of the hash key $\hat{\mathbf{a}} = (a_1, \ldots, a_m)$ (among all vectors in R^m), the collision resistance of h is equivalent to the approximate SVP for ideal lattices.

Main elements of the construction

- Messages are small elements $\mathbf{z}$ in a ring $R = \mathbb{Z}_p[x]/\langle f \rangle$, where f is a unitary polynomial of degree n, irreducible over $\mathbb{Z}$ (and $p \sim C \cdot n^3$ is not necessarily prime).
- The secret key is a pair of short vectors $(\hat{\mathbf{k}}, \hat{\mathbf{l}})$ in R^m ($m \sim \log_2 n$), chosen according to an appropriate distribution.
- The public key is $(h, h(\hat{\mathbf{k}}), h(\hat{\mathbf{l}}))$ where h is a random hash function of the form:

$$h(x_1, \ldots, x_m) = a_1 x_1 + \cdots + a_m x_m$$

For a random choice of the hash key $\hat{\mathbf{a}} = (a_1, \ldots, a_m)$ (among all vectors in R^m), the collision resistance of h is equivalent to the approximate SVP for ideal lattices.

Main elements of the construction

- Messages are small elements $\mathbf{z}$ in a ring $R = \mathbb{Z}_p[x]/\langle f \rangle$, where f is a unitary polynomial of degree n, irreducible over $\mathbb{Z}$ (and $p \sim C \cdot n^3$ is not necessarily prime).
- The secret key is a pair of short vectors $(\hat{\mathbf{k}}, \hat{\mathbf{l}})$ in R^m ($m \sim \log_2 n$), chosen according to an appropriate distribution.
- The public key is $(h, h(\hat{\mathbf{k}}), h(\hat{\mathbf{l}}))$ where h is a random hash function of the form:

$$h(x_1, \ldots, x_m) = a_1 x_1 + \cdots + a_m x_m$$

  For a random choice of the hash key $\hat{\mathbf{a}} = (a_1, \ldots, a_m)$ (among all vectors in R^m), the collision resistance of h is equivalent to the approximate SVP for ideal lattices.

The one-time signature scheme

- KeyGen(1^n): sk $= (\hat{\mathbf{k}}, \hat{\mathbf{l}})$, picked randomly according to a distribution that gives smaller vectors more weight; pk $= (h, h(\hat{\mathbf{k}}), h(\hat{\mathbf{l}}))$, with the key of h chosen at random.
- Sign($\mathbf{z}$): $\hat{\mathbf{s}} = \hat{\mathbf{k}}\mathbf{z} + \hat{\mathbf{l}}$.
- Verify($\mathbf{z}, \hat{\mathbf{s}}$): accept if $\hat{\mathbf{s}}$ is small enough and $h(\hat{\mathbf{s}}) = h(\hat{\mathbf{k}})\mathbf{z} + h(\hat{\mathbf{l}})$.

Lyubashevsky and Micciancio show that this scheme (when made precise) is a (strongly) unforgeable one-time signature scheme, assuming the hardness of the stated approximate SVP problem.

The one-time signature scheme

- KeyGen(1^n): $\text{sk} = (\hat{\mathbf{k}}, \hat{\mathbf{l}})$, picked randomly according to a distribution that gives smaller vectors more weight; $\text{pk} = (h, h(\hat{\mathbf{k}}), h(\hat{\mathbf{l}}))$, with the key of h chosen at random.
- Sign($\mathbf{z}$): $\hat{\mathbf{s}} = \hat{\mathbf{k}}\mathbf{z} + \hat{\mathbf{l}}$.
- Verify($\mathbf{z}, \hat{\mathbf{s}}$): accept if $\hat{\mathbf{s}}$ is small enough and $h(\hat{\mathbf{s}}) = h(\hat{\mathbf{k}})\mathbf{z} + h(\hat{\mathbf{l}})$.

Lyubashevsky and Micciancio show that this scheme (when made precise) is a (strongly) unforgeable one-time signature scheme, assuming the hardness of the stated approximate SVP problem.

The one-time signature scheme

- KeyGen(1^n): sk $= (\hat{\mathbf{k}}, \hat{\mathbf{l}})$, picked randomly according to a distribution that gives smaller vectors more weight; pk $= (h, h(\hat{\mathbf{k}}), h(\hat{\mathbf{l}}))$, with the key of h chosen at random.
- Sign($\mathbf{z}$): $\hat{\mathbf{s}} = \hat{\mathbf{k}}\mathbf{z} + \hat{\mathbf{l}}$.
- Verify($\mathbf{z}, \hat{\mathbf{s}}$): accept if $\hat{\mathbf{s}}$ is small enough and $h(\hat{\mathbf{s}}) = h(\hat{\mathbf{k}})\mathbf{z} + h(\hat{\mathbf{l}})$.

Lyubashevsky and Micciancio show that this scheme (when made precise) is a (strongly) unforgeable one-time signature scheme, assuming the hardness of the stated approximate SVP problem.

The one-time signature scheme

- KeyGen(1^n): $\mathsf{sk} = (\hat{\mathbf{k}}, \hat{\mathbf{l}})$, picked randomly according to a distribution that gives smaller vectors more weight; $\mathsf{pk} = (h, h(\hat{\mathbf{k}}), h(\hat{\mathbf{l}}))$, with the key of h chosen at random.
- Sign($\mathbf{z}$): $\hat{\mathbf{s}} = \hat{\mathbf{k}}\mathbf{z} + \hat{\mathbf{l}}$.
- Verify($\mathbf{z}, \hat{\mathbf{s}}$): accept if $\hat{\mathbf{s}}$ is small enough and $h(\hat{\mathbf{s}}) = h(\hat{\mathbf{k}})\mathbf{z} + h(\hat{\mathbf{l}})$.

Lyubashevsky and Micciancio show that this scheme (when made precise) is a (strongly) unforgeable one-time signature scheme, assuming the hardness of the stated approximate SVP problem.

The one-time signature scheme

- KeyGen(1^n): $\mathsf{sk} = (\hat{\mathbf{k}}, \hat{\mathbf{l}})$, picked randomly according to a distribution that gives smaller vectors more weight; $\mathsf{pk} = (h, h(\hat{\mathbf{k}}), h(\hat{\mathbf{l}}))$, with the key of h chosen at random.
- Sign($\mathbf{z}$): $\hat{\mathbf{s}} = \hat{\mathbf{k}}\mathbf{z} + \hat{\mathbf{l}}$.
- Verify($\mathbf{z}, \hat{\mathbf{s}}$): accept if $\hat{\mathbf{s}}$ is small enough and $h(\hat{\mathbf{s}}) = h(\hat{\mathbf{k}})\mathbf{z} + h(\hat{\mathbf{l}})$.

Lyubashevsky and Micciancio show that this scheme (when made precise) is a (strongly) unforgeable one-time signature scheme, assuming the hardness of the stated approximate SVP problem.

Main points of the proof

- If some adversary, given a signature on a message $\mathbf{z}$ of his choice, can forge a signature $\hat{\mathbf{s}}'$ on $\mathbf{z}' \neq \mathbf{z}$, one can break the collision resistance of h, and hence solve approximate SVP.
- Indeed, we then have $h(\hat{\mathbf{s}}') = h(\hat{\mathbf{k}}\mathbf{z}' + \hat{\mathbf{l}})$. This is a collision, unless $\hat{\mathbf{s}}' = \hat{\mathbf{k}}\mathbf{z}' + \hat{\mathbf{l}}$.
- However, if the adversary can produce $\mathbf{z}'$ and $\hat{\mathbf{k}}\mathbf{z}' + \hat{\mathbf{l}}$, she can recover the signing key $(\hat{\mathbf{k}}, \hat{\mathbf{l}})$ from the result of the oracle query.
- But doing so is information theoretically impossible, because the information available to the adversary, namely $(h(\hat{\mathbf{k}}), h(\hat{\mathbf{l}}), \hat{\mathbf{k}}\mathbf{z} + \hat{\mathbf{l}})$ corresponds to exponentially many signing keys $(\hat{\mathbf{k}}, \hat{\mathbf{l}})$.
- If an adversary obtains a second signature on the message she queried, she also gets a collision on h, hence strong unforgeability.

Main points of the proof

- If some adversary, given a signature on a message $\mathbf{z}$ of his choice, can forge a signature $\hat{\mathbf{s}}'$ on $\mathbf{z}' \neq \mathbf{z}$, one can break the collision resistance of h, and hence solve approximate SVP.
- Indeed, we then have $h(\hat{\mathbf{s}}') = h(\hat{\mathbf{k}}\mathbf{z}' + \hat{\mathbf{l}})$. This is a collision, unless $\hat{\mathbf{s}}' = \hat{\mathbf{k}}\mathbf{z}' + \hat{\mathbf{l}}$.
- However, if the adversary can produce $\mathbf{z}'$ and $\hat{\mathbf{k}}\mathbf{z}' + \hat{\mathbf{l}}$, she can recover the signing key $(\hat{\mathbf{k}}, \hat{\mathbf{l}})$ from the result of the oracle query.
- But doing so is information theoretically impossible, because the information available to the adversary, namely $(h(\hat{\mathbf{k}}), h(\hat{\mathbf{l}}), \hat{\mathbf{k}}\mathbf{z} + \hat{\mathbf{l}})$ corresponds to exponentially many signing keys $(\hat{\mathbf{k}}, \hat{\mathbf{l}})$.
- If an adversary obtains a second signature on the message she queried, she also gets a collision on h, hence strong unforgeability.

Main points of the proof

- If some adversary, given a signature on a message $\mathbf{z}$ of his choice, can forge a signature $\hat{\mathbf{s}}'$ on $\mathbf{z}' \neq \mathbf{z}$, one can break the collision resistance of h, and hence solve approximate SVP.
- Indeed, we then have $h(\hat{\mathbf{s}}') = h(\hat{\mathbf{k}}\mathbf{z}' + \hat{\mathbf{l}})$. This is a collision, unless $\hat{\mathbf{s}}' = \hat{\mathbf{k}}\mathbf{z}' + \hat{\mathbf{l}}$.
- However, if the adversary can produce $\mathbf{z}'$ and $\hat{\mathbf{k}}\mathbf{z}' + \hat{\mathbf{l}}$, she can recover the signing key $(\hat{\mathbf{k}}, \hat{\mathbf{l}})$ from the result of the oracle query.
- But doing so is information theoretically impossible, because the information available to the adversary, namely $(h(\hat{\mathbf{k}}), h(\hat{\mathbf{l}}), \hat{\mathbf{k}}\mathbf{z} + \hat{\mathbf{l}})$ corresponds to exponentially many signing keys $(\hat{\mathbf{k}}, \hat{\mathbf{l}})$.
- If an adversary obtains a second signature on the message she queried, she also gets a collision on h, hence strong unforgeability.

Main points of the proof

- If some adversary, given a signature on a message $\mathbf{z}$ of his choice, can forge a signature $\hat{\mathbf{s}}'$ on $\mathbf{z}' \neq \mathbf{z}$, one can break the collision resistance of h, and hence solve approximate SVP.
- Indeed, we then have $h(\hat{\mathbf{s}}') = h(\hat{\mathbf{k}}\mathbf{z}' + \hat{\mathbf{l}})$. This is a collision, unless $\hat{\mathbf{s}}' = \hat{\mathbf{k}}\mathbf{z}' + \hat{\mathbf{l}}$.
- However, if the adversary can produce $\mathbf{z}'$ and $\hat{\mathbf{k}}\mathbf{z}' + \hat{\mathbf{l}}$, she can recover the signing key $(\hat{\mathbf{k}}, \hat{\mathbf{l}})$ from the result of the oracle query.
- But doing so is information theoretically impossible, because the information available to the adversary, namely $(h(\hat{\mathbf{k}}), h(\hat{\mathbf{l}}), \hat{\mathbf{k}}\mathbf{z} + \hat{\mathbf{l}})$ corresponds to exponentially many signing keys $(\hat{\mathbf{k}}, \hat{\mathbf{l}})$.
- If an adversary obtains a second signature on the message she queried, she also gets a collision on h, hence strong unforgeability.

Main points of the proof

- If some adversary, given a signature on a message $\mathbf{z}$ of his choice, can forge a signature $\hat{\mathbf{s}}'$ on $\mathbf{z}' \neq \mathbf{z}$, one can break the collision resistance of h, and hence solve approximate SVP.
- Indeed, we then have $h(\hat{\mathbf{s}}') = h(\hat{\mathbf{k}}\mathbf{z}' + \hat{\mathbf{l}})$. This is a collision, unless $\hat{\mathbf{s}}' = \hat{\mathbf{k}}\mathbf{z}' + \hat{\mathbf{l}}$.
- However, if the adversary can produce $\mathbf{z}'$ and $\hat{\mathbf{k}}\mathbf{z}' + \hat{\mathbf{l}}$, she can recover the signing key $(\hat{\mathbf{k}}, \hat{\mathbf{l}})$ from the result of the oracle query.
- But doing so is information theoretically impossible, because the information available to the adversary, namely $(h(\hat{\mathbf{k}}), h(\hat{\mathbf{l}}), \hat{\mathbf{k}}\mathbf{z} + \hat{\mathbf{l}})$ corresponds to exponentially many signing keys $(\hat{\mathbf{k}}, \hat{\mathbf{l}})$.
- If an adversary obtains a second signature on the message she queried, she also gets a collision on h, hence strong unforgeability.

Outline

Vector length

To define small elements in $R = \mathbb{Z}_p[x]/\langle f \rangle$ and short vectors in R^m, one introduces the infinity "norm":

- for $\mathbf{z} \in R$, $\|\mathbf{z}\|_\infty$ is the supremum of the absolute values of the coefficients of $\mathbf{z}$ considered as a polynomial in $\mathbb{Z}[x]$ of degree $< n$ with coefficients in $(-p/2, p/2]$;
- for vectors in R^m, we set $\|(\mathbf{z}_1, \ldots, \mathbf{z}_m)\|_\infty = \sup_j \|\mathbf{z}_j\|_\infty$;
- $\|\mathbf{a} + \mathbf{b}\|_\infty \leq \|\mathbf{a}\|_\infty + \|\mathbf{b}\|_\infty$;
- $\|\alpha \mathbf{a}\|_\infty \leq |\alpha| \cdot \|\mathbf{a}\|_\infty$ for $\alpha \in \mathbb{Z}$;
- $\|\mathbf{ab}\|_\infty \leq \phi n \|\mathbf{a}\|_\infty \|\mathbf{b}\|_\infty$ for some constant ϕ depending only on f. Some polynomials f of arbitrarily large degree can ensure a small value for ϕ (say $\phi \leq 2$).

Vector length

To define small elements in $R = \mathbb{Z}_p[x]/\langle f \rangle$ and short vectors in R^m, one introduces the infinity "norm":

- for $\mathbf{z} \in R$, $\|\mathbf{z}\|_\infty$ is the supremum of the absolute values of the coefficients of $\mathbf{z}$ considered as a polynomial in $\mathbb{Z}[x]$ of degree $< n$ with coefficients in $(-p/2, p/2]$;
- for vectors in R^m, we set $\|(\mathbf{z}_1, \ldots, \mathbf{z}_m)\|_\infty = \sup_j \|\mathbf{z}_j\|_\infty$;
- $\|\mathbf{a} + \mathbf{b}\|_\infty \leq \|\mathbf{a}\|_\infty + \|\mathbf{b}\|_\infty$;
- $\|\alpha \mathbf{a}\|_\infty \leq |\alpha| \cdot \|\mathbf{a}\|_\infty$ for $\alpha \in \mathbb{Z}$;
- $\|\mathbf{ab}\|_\infty \leq \phi n \|\mathbf{a}\|_\infty \|\mathbf{b}\|_\infty$ for some constant ϕ depending only on f. Some polynomials f of arbitrarily large degree can ensure a small value for ϕ (say $\phi \leq 2$).

Vector length

To define small elements in $R = \mathbb{Z}_p[x]/\langle f \rangle$ and short vectors in R^m, one introduces the infinity "norm":

- for $\mathbf{z} \in R$, $\|\mathbf{z}\|_\infty$ is the supremum of the absolute values of the coefficients of $\mathbf{z}$ considered as a polynomial in $\mathbb{Z}[x]$ of degree $< n$ with coefficients in $(-p/2, p/2]$;
- for vectors in R^m, we set $\|(\mathbf{z}_1, \ldots, \mathbf{z}_m)\|_\infty = \sup_j \|\mathbf{z}_j\|_\infty$;
- $\|\mathbf{a} + \mathbf{b}\|_\infty \leq \|\mathbf{a}\|_\infty + \|\mathbf{b}\|_\infty$;
- $\|\alpha\mathbf{a}\|_\infty \leq |\alpha| \cdot \|\mathbf{a}\|_\infty$ for $\alpha \in \mathbb{Z}$;
- $\|\mathbf{ab}\|_\infty \leq \phi n \|\mathbf{a}\|_\infty \|\mathbf{b}\|_\infty$ for some constant ϕ depending only on f. Some polynomials f of arbitrarily large degree can ensure a small value for ϕ (say $\phi \leq 2$).

Vector length

To define small elements in $R = \mathbb{Z}_p[x]/\langle f \rangle$ and short vectors in R^m, one introduces the infinity "norm":

- for $\mathbf{z} \in R$, $\|\mathbf{z}\|_\infty$ is the supremum of the absolute values of the coefficients of $\mathbf{z}$ considered as a polynomial in $\mathbb{Z}[x]$ of degree $< n$ with coefficients in $(-p/2, p/2]$;
- for vectors in R^m, we set $\|(\mathbf{z}_1, \ldots, \mathbf{z}_m)\|_\infty = \sup_j \|\mathbf{z}_j\|_\infty$;
- $\|\mathbf{a} + \mathbf{b}\|_\infty \leq \|\mathbf{a}\|_\infty + \|\mathbf{b}\|_\infty$;
- $\|\alpha \mathbf{a}\|_\infty \leq |\alpha| \cdot \|\mathbf{a}\|_\infty$ for $\alpha \in \mathbb{Z}$;
- $\|\mathbf{ab}\|_\infty \leq \phi n \|\mathbf{a}\|_\infty \|\mathbf{b}\|_\infty$ for some constant ϕ depending only on f. Some polynomials f of arbitrarily large degree can ensure a small value for ϕ (say $\phi \leq 2$).

Vector length

To define small elements in $R = \mathbb{Z}_p[x]/\langle f \rangle$ and short vectors in R^m, one introduces the infinity "norm":

- for $\mathbf{z} \in R$, $\|\mathbf{z}\|_\infty$ is the supremum of the absolute values of the coefficients of $\mathbf{z}$ considered as a polynomial in $\mathbb{Z}[x]$ of degree $< n$ with coefficients in $(-p/2, p/2]$;
- for vectors in R^m, we set $\|(\mathbf{z}_1, \ldots, \mathbf{z}_m)\|_\infty = \sup_j \|\mathbf{z}_j\|_\infty$;
- $\|\mathbf{a} + \mathbf{b}\|_\infty \leq \|\mathbf{a}\|_\infty + \|\mathbf{b}\|_\infty$;
- $\|\alpha \mathbf{a}\|_\infty \leq |\alpha| \cdot \|\mathbf{a}\|_\infty$ for $\alpha \in \mathbb{Z}$;
- $\|\mathbf{ab}\|_\infty \leq \phi n \|\mathbf{a}\|_\infty \|\mathbf{b}\|_\infty$ for some constant ϕ depending only on f. Some polynomials f of arbitrarily large degree can ensure a small value for ϕ (say $\phi \leq 2$).

Collision problem

Let $\mathcal{H}_{R,m}$ be the set of hash functions $h : R^m \to R$ of the form $h_{\hat{\mathbf{a}}}(\hat{\mathbf{x}}) = a_1 x_1 + \cdots + a_m x_m$.

The collision problem Col_d takes as input a random $h \in \mathcal{H}_{R,m}$ and asks to find $\hat{\mathbf{s}} \neq \hat{\mathbf{s}}'$ such that $h(\hat{\mathbf{s}}) = h(\hat{\mathbf{s}}')$.

For $p = (\phi n)^3$, $m = \lceil \log n \rceil$ and $d = 10\phi p^{1/m} \log^2 n$, Col_d is as hard as approximating the shortest vector in every lattice corresponding to an ideal of $\mathbb{Z}[x]/\langle f \rangle$ within a factor of $\tilde{O}(\phi^5 n^2)$.

Collision problem

Let $\mathcal{H}_{R,m}$ be the set of hash functions $h : R^m \to R$ of the form $h_{\hat{\mathbf{a}}}(\hat{\mathbf{x}}) = a_1 x_1 + \cdots + a_m x_m$.

The collision problem Col_d takes as input a random $h \in \mathcal{H}_{R,m}$ and asks to find $\hat{\mathbf{s}} \neq \hat{\mathbf{s}}'$ such that $h(\hat{\mathbf{s}}) = h(\hat{\mathbf{s}}')$.

For $p = (\phi n)^3$, $m = \lceil \log n \rceil$ and $d = 10\phi p^{1/m} \log^2 n$, Col_d is as hard as approximating the shortest vector in every lattice corresponding to an ideal of $\mathbb{Z}[x]/\langle f \rangle$ within a factor of $\tilde{O}(\phi^5 n^2)$.

Collision problem

Let $\mathcal{H}_{R,m}$ be the set of hash functions $h : R^m \to R$ of the form $h_{\hat{\mathbf{a}}}(\hat{\mathbf{x}}) = a_1 x_1 + \cdots + a_m x_m$.

The collision problem Col_d takes as input a random $h \in \mathcal{H}_{R,m}$ and asks to find $\hat{\mathbf{s}} \neq \hat{\mathbf{s}}'$ such that $h(\hat{\mathbf{s}}) = h(\hat{\mathbf{s}}')$.

For $p = (\phi n)^3$, $m = \lceil \log n \rceil$ and $d = 10\phi p^{1/m} \log^2 n$, Col_d is as hard as approximating the shortest vector in every lattice corresponding to an ideal of $\mathbb{Z}[x]/\langle f \rangle$ within a factor of $\tilde{O}(\phi^5 n^2)$.

Precise form of the OTSS

- KeyGen($1^n, f$): let $p = (\phi n)^3$, $m = \lceil \log n \rceil$, $R = \mathbb{Z}_p[x]/\langle f \rangle$. Moreover, define:

$$DK_i = \{\hat{\mathbf{y}} \in R^m \mid \|\hat{\mathbf{y}}\|_\infty \leq 5ip^{1/m}\}$$
$$DL_i = \{\hat{\mathbf{y}} \in R^m \mid \|\hat{\mathbf{y}}\|_\infty \leq 5in\phi p^{1/m}\}$$

  Choose $h \in \mathcal{H}_{R,m}$ uniformly at random. Pick $\hat{\mathbf{k}}$ and $\hat{\mathbf{l}}$ uniformly at random in DK_j and DL_j, where j is the position of the first 1 in a random string $r \in \{0,1\}^{\lfloor \log^2 n \rfloor}$. Then sk $= (\hat{\mathbf{k}}, \hat{\mathbf{l}})$, pk $= (h, h(\hat{\mathbf{k}}), h(\hat{\mathbf{l}}))$.
- Sign($\mathbf{z} \in R, \|\mathbf{z}\|_\infty \leq 1$): $\hat{\mathbf{s}} = \hat{\mathbf{k}}\mathbf{z} + \hat{\mathbf{l}}$.
- Verify($\mathbf{z}, \hat{\mathbf{s}}$): accept if $\|\hat{\mathbf{s}}\|_\infty \leq 10\phi p^{1/m} n \log^2 n$ and $h(\hat{\mathbf{s}}) = h(\hat{\mathbf{k}})\mathbf{z} + h(\hat{\mathbf{l}})$.

Precise form of the OTSS

- KeyGen($1^n, f$): let $p = (\phi n)^3$, $m = \lceil \log n \rceil$, $R = \mathbb{Z}_p[x]/\langle f \rangle$. Moreover, define:

$$DK_i = \{\hat{\mathbf{y}} \in R^m \mid \|\hat{\mathbf{y}}\|_\infty \leq 5ip^{1/m}\}$$
$$DL_i = \{\hat{\mathbf{y}} \in R^m \mid \|\hat{\mathbf{y}}\|_\infty \leq 5in\phi p^{1/m}\}$$

  Choose $h \in \mathcal{H}_{R,m}$ uniformly at random. Pick $\hat{\mathbf{k}}$ and $\hat{\mathbf{l}}$ uniformly at random in DK_j and DL_j, where j is the position of the first 1 in a random string $r \in \{0,1\}^{\lfloor \log^2 n \rfloor}$. Then sk $= (\hat{\mathbf{k}}, \hat{\mathbf{l}})$, pk $= (h, h(\hat{\mathbf{k}}), h(\hat{\mathbf{l}}))$.
- Sign($\mathbf{z} \in R, \|\mathbf{z}\|_\infty \leq 1$): $\hat{\mathbf{s}} = \hat{\mathbf{k}}\mathbf{z} + \hat{\mathbf{l}}$.
- Verify($\mathbf{z}, \hat{\mathbf{s}}$): accept if $\|\hat{\mathbf{s}}\|_\infty \leq 10\phi p^{1/m} n \log^2 n$ and $h(\hat{\mathbf{s}}) = h(\hat{\mathbf{k}})\mathbf{z} + h(\hat{\mathbf{l}})$.

Precise form of the OTSS

- KeyGen($1^n, f$): let $p = (\phi n)^3$, $m = \lceil \log n \rceil$, $R = \mathbb{Z}_p[x]/\langle f \rangle$. Moreover, define:

$$DK_i = \{\hat{\mathbf{y}} \in R^m \mid \|\hat{\mathbf{y}}\|_\infty \leq 5ip^{1/m}\}$$
$$DL_i = \{\hat{\mathbf{y}} \in R^m \mid \|\hat{\mathbf{y}}\|_\infty \leq 5in\phi p^{1/m}\}$$

  Choose $h \in \mathcal{H}_{R,m}$ uniformly at random. Pick $\hat{\mathbf{k}}$ and $\hat{\mathbf{l}}$ uniformly at random in DK_j and DL_j, where j is the position of the first 1 in a random string $r \in \{0,1\}^{\lfloor \log^2 n \rfloor}$. Then sk $= (\hat{\mathbf{k}}, \hat{\mathbf{l}})$, pk $= (h, h(\hat{\mathbf{k}}), h(\hat{\mathbf{l}}))$.
- Sign($\mathbf{z} \in R, \|\mathbf{z}\|_\infty \leq 1$): $\hat{\mathbf{s}} = \hat{\mathbf{k}}\mathbf{z} + \hat{\mathbf{l}}$.
- Verify($\mathbf{z}, \hat{\mathbf{s}}$): accept if $\|\hat{\mathbf{s}}\|_\infty \leq 10\phi p^{1/m} n \log^2 n$ and $h(\hat{\mathbf{s}}) = h(\hat{\mathbf{k}})\mathbf{z} + h(\hat{\mathbf{l}})$.

Recovering the signing key from a forgery

Suppose the attacker obtains a signature $\hat{\mathbf{s}}'$ on $\mathbf{z}'$ after getting $\hat{\mathbf{s}}$ on $\mathbf{z}$. If it doesn't yield a collision, we get $\hat{\mathbf{s}}' = \hat{\mathbf{k}}\mathbf{z}' + \hat{\mathbf{l}}$, hence:

$$\hat{\mathbf{s}}' - \hat{\mathbf{s}} = \hat{\mathbf{k}}(\mathbf{z}' - \mathbf{z})$$

This actually holds in $\mathbb{Z}[x]/\langle f \rangle$, since the polynomials on the right have coefficients too small to be reduced $\bmod p$ when multiplied:

$$\|\mathbf{z}' - \mathbf{z}\|_\infty \leq 2 \text{ and } \|\hat{\mathbf{k}}\|_\infty \leq 5p^{1/m} \log^2 n$$

so the product is of norm $o(n^2)$, whereas $p = \Omega(n^3)$.

Now, R is an integral domain, since f is irreducible. Therefore:

$$\hat{\mathbf{k}} = \frac{\hat{\mathbf{s}}' - \hat{\mathbf{s}}}{\mathbf{z}' - \mathbf{z}}$$

Thus, the adversary recovers $\hat{\mathbf{k}}$, and then $\hat{\mathbf{l}}$.

Recovering the signing key from a forgery

Suppose the attacker obtains a signature $\hat{\mathbf{s}}'$ on $\mathbf{z}'$ after getting $\hat{\mathbf{s}}$ on $\mathbf{z}$. If it doesn't yield a collision, we get $\hat{\mathbf{s}}' = \hat{\mathbf{k}}\mathbf{z}' + \hat{\mathbf{l}}$, hence:

$$\hat{\mathbf{s}}' - \hat{\mathbf{s}} = \hat{\mathbf{k}}(\mathbf{z}' - \mathbf{z})$$

This actually holds in $\mathbb{Z}[x]/\langle f \rangle$, since the polynomials on the right have coefficients too small to be reduced $\bmod p$ when multiplied:

$$\|\mathbf{z}' - \mathbf{z}\|_\infty \leq 2 \text{ and } \|\hat{\mathbf{k}}\|_\infty \leq 5p^{1/m} \log^2 n$$

so the product is of norm $o(n^2)$, whereas $p = \Omega(n^3)$.

Now, R is an integral domain, since f is irreducible. Therefore:

$$\hat{\mathbf{k}} = \frac{\hat{\mathbf{s}}' - \hat{\mathbf{s}}}{\mathbf{z}' - \mathbf{z}}$$

Thus, the adversary recovers $\hat{\mathbf{k}}$, and then $\hat{\mathbf{l}}$.

Recovering the signing key from a forgery

Suppose the attacker obtains a signature $\hat{\mathbf{s}}'$ on $\mathbf{z}'$ after getting $\hat{\mathbf{s}}$ on $\mathbf{z}$. If it doesn't yield a collision, we get $\hat{\mathbf{s}}' = \hat{\mathbf{k}}\mathbf{z}' + \hat{\mathbf{l}}$, hence:

$$\hat{\mathbf{s}}' - \hat{\mathbf{s}} = \hat{\mathbf{k}}(\mathbf{z}' - \mathbf{z})$$

This actually holds in $\mathbb{Z}[x]/\langle f \rangle$, since the polynomials on the right have coefficients too small to be reduced $\bmod p$ when multiplied:

$$\|\mathbf{z}' - \mathbf{z}\|_\infty \leq 2 \text{ and } \|\hat{\mathbf{k}}\|_\infty \leq 5p^{1/m} \log^2 n$$

so the product is of norm $o(n^2)$, whereas $p = \Omega(n^3)$.

Now, R is an integral domain, since f is irreducible. Therefore:

$$\hat{\mathbf{k}} = \frac{\hat{\mathbf{s}}' - \hat{\mathbf{s}}}{\mathbf{z}' - \mathbf{z}}$$

Thus, the adversary recovers $\hat{\mathbf{k}}$, and then $\hat{\mathbf{l}}$.

Recovering the signing key is impossible

To complete the proof, it remains to show that the adversary cannot possibly recover the signing key from the information available to her, namely $(\mathbf{K}, \mathbf{L}, \hat{\mathbf{s}}) = (h(\hat{\mathbf{k}}), h(\hat{\mathbf{l}}), \hat{\mathbf{k}}\mathbf{z} + \hat{\mathbf{l}})$.

Since it happens with negligible probability that $\hat{\mathbf{k}}, \hat{\mathbf{l}}$ are picked from DK_j, DL_j with $j = \lfloor \log^2 n \rfloor$, we can assume that they belong to DK_{j-1}, DL_{j-1}.

Suppose then that we fix a verification key $(h, \mathbf{K}, \mathbf{L})$ and a signature $\hat{\mathbf{s}}$ on a message $\mathbf{z}$. The authors prove using a counting argument that, for any given signing key $(\hat{\mathbf{k}}, \hat{\mathbf{l}}) \in DK_{j-1} \times DL_{j-1}$ such that $h(\hat{\mathbf{k}}) = \mathbf{K}$, $h(\hat{\mathbf{l}}) = \mathbf{L}$ and $\hat{\mathbf{s}} = \hat{\mathbf{k}}\mathbf{z} + \hat{\mathbf{l}}$, the probability that this was the actual signing key generated by the key generation algorithm is negligibly small (tight reduction).

Recovering the signing key is impossible

To complete the proof, it remains to show that the adversary cannot possibly recover the signing key from the information available to her, namely $(\mathbf{K}, \mathbf{L}, \hat{\mathbf{s}}) = (h(\hat{\mathbf{k}}), h(\hat{\mathbf{l}}), \hat{\mathbf{k}}\mathbf{z} + \hat{\mathbf{l}})$.

Since it happens with negligible probability that $\hat{\mathbf{k}}, \hat{\mathbf{l}}$ are picked from DK_j, DL_j with $j = \lfloor \log^2 n \rfloor$, we can assume that they belong to DK_{j-1}, DL_{j-1}.

Suppose then that we fix a verification key $(h, \mathbf{K}, \mathbf{L})$ and a signature $\hat{\mathbf{s}}$ on a message $\mathbf{z}$. The authors prove using a counting argument that, for any given signing key $(\hat{\mathbf{k}}, \hat{\mathbf{l}}) \in DK_{j-1} \times DL_{j-1}$ such that $h(\hat{\mathbf{k}}) = \mathbf{K}$, $h(\hat{\mathbf{l}}) = \mathbf{L}$ and $\hat{\mathbf{s}} = \hat{\mathbf{k}}\mathbf{z} + \hat{\mathbf{l}}$, the probability that this was the actual signing key generated by the key generation algorithm is negligibly small (tight reduction).

Recovering the signing key is impossible

To complete the proof, it remains to show that the adversary cannot possibly recover the signing key from the information available to her, namely $(\mathbf{K}, \mathbf{L}, \hat{\mathbf{s}}) = (h(\hat{\mathbf{k}}), h(\hat{\mathbf{l}}), \hat{\mathbf{k}}\mathbf{z} + \hat{\mathbf{l}})$.

Since it happens with negligible probability that $\hat{\mathbf{k}}, \hat{\mathbf{l}}$ are picked from DK_j, DL_j with $j = \lfloor \log^2 n \rfloor$, we can assume that they belong to DK_{j-1}, DL_{j-1}.

Suppose then that we fix a verification key $(h, \mathbf{K}, \mathbf{L})$ and a signature $\hat{\mathbf{s}}$ on a message $\mathbf{z}$. The authors prove using a counting argument that, for any given signing key $(\hat{\mathbf{k}}, \hat{\mathbf{l}}) \in DK_{j-1} \times DL_{j-1}$ such that $h(\hat{\mathbf{k}}) = \mathbf{K}$, $h(\hat{\mathbf{l}}) = \mathbf{L}$ and $\hat{\mathbf{s}} = \hat{\mathbf{k}}\mathbf{z} + \hat{\mathbf{l}}$, the probability that this was the actual signing key generated by the key generation algorithm is negligibly small (tight reduction).

Sketch of the counting argument

Consider $Y = \{\hat{\mathbf{y}} \in R^m \mid \|y\|_\infty \leq 5p^{1/m}$ and $h(\hat{\mathbf{y}}) = 0\}$. A pigeonhole argument shows that $|Y| \geq 5^{mn}$.

Now if we let $\hat{\mathbf{k}}' = \hat{\mathbf{k}} + \hat{\mathbf{y}}$, $\hat{\mathbf{l}}' = \hat{\mathbf{l}} - \hat{\mathbf{y}}\mathbf{z}$, we have $h(\hat{\mathbf{k}}') = \mathbf{K}$, $h(\hat{\mathbf{l}}') = \mathbf{L}$ and $\hat{\mathbf{k}}'\mathbf{z} + \hat{\mathbf{l}}' = \hat{\mathbf{s}}$. Moreover:

$$\|\hat{\mathbf{k}}'\|_\infty \leq \|\hat{\mathbf{k}}\|_\infty + 5p^{1/m} \leq 5p^{1/m}\lfloor \log^2 n \rfloor$$
$$\|\hat{\mathbf{l}}'\|_\infty \leq \|l\|_\infty + 5p^{1/m} \cdot \phi n \leq 5\phi n p^{1/m}\lfloor \log^2 n \rfloor$$

Thus, $(\hat{\mathbf{k}}', \hat{\mathbf{l}}')$ is always a possible signing key corresponding to $(h, \mathbf{K}, \mathbf{L})$ and $\hat{\mathbf{s}}$.

The probability of the actual signing key being $(\hat{\mathbf{k}}, \hat{\mathbf{l}})$ conditional to $(h, \mathbf{K}, \mathbf{L})$ and $\hat{\mathbf{s}}$ is thus bounded by $(\hat{\mathbf{k}}, \hat{\mathbf{l}})$ being picked among keys of the form $(\hat{\mathbf{k}}', \hat{\mathbf{l}}')$, which is shown to be exponentially small.

Sketch of the counting argument

Consider $Y = \{\hat{\mathbf{y}} \in R^m \mid \|y\|_\infty \leq 5p^{1/m} \text{ and } h(\hat{\mathbf{y}}) = 0\}$. A pigeonhole argument shows that $|Y| \geq 5^{mn}$.

Now if we let $\hat{\mathbf{k}}' = \hat{\mathbf{k}} + \hat{\mathbf{y}}$, $\hat{\mathbf{l}}' = \hat{\mathbf{l}} - \hat{\mathbf{y}}\mathbf{z}$, we have $h(\hat{\mathbf{k}}') = \mathbf{K}$, $h(\hat{\mathbf{l}}') = \mathbf{L}$ and $\hat{\mathbf{k}}'\mathbf{z} + \hat{\mathbf{l}}' = \hat{\mathbf{s}}$. Moreover:

$$\|\hat{\mathbf{k}}'\|_\infty \leq \|\hat{\mathbf{k}}\|_\infty + 5p^{1/m} \leq 5p^{1/m}\lfloor \log^2 n \rfloor$$
$$\|\hat{\mathbf{l}}'\|_\infty \leq \|l\|_\infty + 5p^{1/m} \cdot \phi n \leq 5\phi n p^{1/m}\lfloor \log^2 n \rfloor$$

Thus, $(\hat{\mathbf{k}}', \hat{\mathbf{l}}')$ is always a possible signing key corresponding to $(h, \mathbf{K}, \mathbf{L})$ and $\hat{\mathbf{s}}$.

The probability of the actual signing key being $(\hat{\mathbf{k}}, \hat{\mathbf{l}})$ conditional to $(h, \mathbf{K}, \mathbf{L})$ and $\hat{\mathbf{s}}$ is thus bounded by $(\hat{\mathbf{k}}, \hat{\mathbf{l}})$ being picked among keys of the form $(\hat{\mathbf{k}}', \hat{\mathbf{l}}')$, which is shown to be exponentially small.

Sketch of the counting argument

Consider $Y = \{\hat{\mathbf{y}} \in R^m \mid \|y\|_\infty \leq 5p^{1/m} \text{ and } h(\hat{\mathbf{y}}) = 0\}$. A pigeonhole argument shows that $|Y| \geq 5^{mn}$.

Now if we let $\hat{\mathbf{k}}' = \hat{\mathbf{k}} + \hat{\mathbf{y}}$, $\hat{\mathbf{l}}' = \hat{\mathbf{l}} - \hat{\mathbf{y}}\mathbf{z}$, we have $h(\hat{\mathbf{k}}') = \mathbf{K}$, $h(\hat{\mathbf{l}}') = \mathbf{L}$ and $\hat{\mathbf{k}}'\mathbf{z} + \hat{\mathbf{l}}' = \hat{\mathbf{s}}$. Moreover:

$$\|\hat{\mathbf{k}}'\|_\infty \leq \|\hat{\mathbf{k}}\|_\infty + 5p^{1/m} \leq 5p^{1/m}\lfloor \log^2 n \rfloor$$
$$\|\hat{\mathbf{l}}'\|_\infty \leq \|l\|_\infty + 5p^{1/m} \cdot \phi n \leq 5\phi n p^{1/m}\lfloor \log^2 n \rfloor$$

Thus, $(\hat{\mathbf{k}}', \hat{\mathbf{l}}')$ is always a possible signing key corresponding to $(h, \mathbf{K}, \mathbf{L})$ and $\hat{\mathbf{s}}$.

The probability of the actual signing key being $(\hat{\mathbf{k}}, \hat{\mathbf{l}})$ conditional to $(h, \mathbf{K}, \mathbf{L})$ and $\hat{\mathbf{s}}$ is thus bounded by $(\hat{\mathbf{k}}, \hat{\mathbf{l}})$ being picked among keys of the form $(\hat{\mathbf{k}}', \hat{\mathbf{l}}')$, which is shown to be exponentially small.

Sketch of the counting argument

Consider $Y = \{\hat{\mathbf{y}} \in R^m \mid \|y\|_\infty \leq 5p^{1/m} \text{ and } h(\hat{\mathbf{y}}) = 0\}$. A pigeonhole argument shows that $|Y| \geq 5^{mn}$.

Now if we let $\hat{\mathbf{k}}' = \hat{\mathbf{k}} + \hat{\mathbf{y}}$, $\hat{\mathbf{l}}' = \hat{\mathbf{l}} - \hat{\mathbf{y}}\mathbf{z}$, we have $h(\hat{\mathbf{k}}') = \mathbf{K}$, $h(\hat{\mathbf{l}}') = \mathbf{L}$ and $\hat{\mathbf{k}}'\mathbf{z} + \hat{\mathbf{l}}' = \hat{\mathbf{s}}$. Moreover:

$$\|\hat{\mathbf{k}}'\|_\infty \leq \|\hat{\mathbf{k}}\|_\infty + 5p^{1/m} \leq 5p^{1/m}\lfloor \log^2 n \rfloor$$
$$\|\hat{\mathbf{l}}'\|_\infty \leq \|l\|_\infty + 5p^{1/m} \cdot \phi n \leq 5\phi n p^{1/m}\lfloor \log^2 n \rfloor$$

Thus, $(\hat{\mathbf{k}}', \hat{\mathbf{l}}')$ is always a possible signing key corresponding to $(h, \mathbf{K}, \mathbf{L})$ and $\hat{\mathbf{s}}$.

The probability of the actual signing key being $(\hat{\mathbf{k}}, \hat{\mathbf{l}})$ conditional to $(h, \mathbf{K}, \mathbf{L})$ and $\hat{\mathbf{s}}$ is thus bounded by $(\hat{\mathbf{k}}, \hat{\mathbf{l}})$ being picked among keys of the form $(\hat{\mathbf{k}}', \hat{\mathbf{l}}')$, which is shown to be exponentially small.

Summary

- One-time secure signature scheme for n-bit messages, with key generation, signature and verification almost linear in the security parameter $k = n$.
- Hence, a stateful signature scheme with efficient signature and verification (but costly key generation).
- Strong unforgeability under chosen-message attack if some approximate SVP in ideal lattices is hard in the worst case.

Summary

- One-time secure signature scheme for n-bit messages, with key generation, signature and verification almost linear in the security parameter $k = n$.
- Hence, a stateful signature scheme with efficient signature and verification (but costly key generation).
- Strong unforgeability under chosen-message attack if some approximate SVP in ideal lattices is hard in the worst case.

Summary

- One-time secure signature scheme for n-bit messages, with key generation, signature and verification almost linear in the security parameter $k = n$.
- Hence, a stateful signature scheme with efficient signature and verification (but costly key generation).
- Strong unforgeability under chosen-message attack if some approximate SVP in ideal lattices is hard in the worst case.

Thank you!