About “k-bit security” of MACs based on hash function Streebog

Vitaly Kiryukhin

LLC «SFB Lab», JSC «InfoTeCS»

CTCrypt 2023

June 9, 2023

vitaly.kiryukhin@sfblaboratory.ru

"k-bit security"

Informal definition

A keyed cryptoalgorithm is "k-bit secure" (to some threat)

if the attacker's probability of success is bounded by

$$p \leq \frac{t}{2^k}$$

t – computational power of the adversary

k – key length in bits

"k-bit security"

Informal definition

A keyed cryptoalgorithm is "k-bit secure" (to some threat)

if the attacker's probability of success is bounded by

$$p \leq \frac{t}{2^k}$$

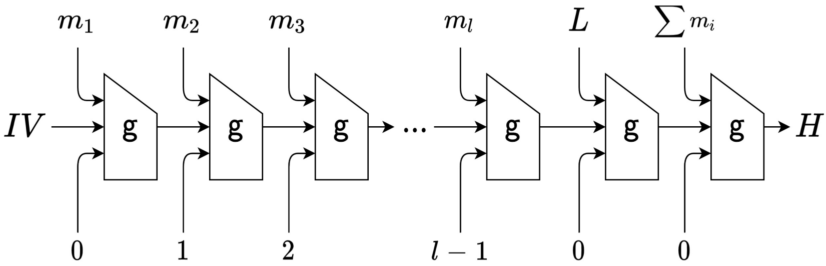
t – computational power of the adversary

k – key length in bits

Further in the presentation, we use the term "k-bit security" simultaneously for key recovery, forgery, and distinguishing

GOST 34.11-2018 – «Streebog»

Streebog is a **keyless** hash function



- Modified MD-structure (checksum and counters have been added)
- Compression function $g : V^n \times V^n \times V^n \to V^n$, $n = 512$ bit
- Finalization with message bit-length L and checksum Σ

From keyless to keyed

Two provably secure ways to transform Streebog to a keyed hash function:

- double hashing

$$\text{HMAC-Streebog}(K, M) = \text{H}\left((K \oplus opad)||\text{H}(K \oplus ipad||M)\right)$$

- key prepending

$$\text{Streebog-K}(K, M) = \text{H}(K||M)$$

Guaranteed security level

CTCrypt 2022:

– security proofs for HMAC-Streebog and Streebog-K,

but only $\frac{k}{2}$-bit security – enough for practical use, but far from ideal

Guaranteed security level

CTCrypt 2022:

– security proofs for HMAC-Streebog and Streebog-K,

but only $\frac{k}{2}$-bit security – enough for practical use, but far from ideal

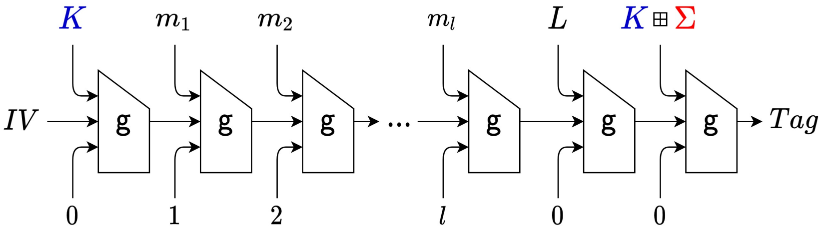
Now, CTCrypt 2023:

– precise tight bounds, k-bit security for many cases

How to improve security bounds?

Bottleneck

Even if keyed Streebog is used properly with **random and uniform keys**, a lot of **related keys** appear **inside** it because of the checksum



New message ⇒ New $\Sigma = m_1 \boxplus \ldots \boxplus m_l$ ⇒ New related key $K \boxplus \Sigma$

Even more related keys in HMAC-Streebog

How to improve security bounds?

Bottleneck

Even if keyed Streebog is used properly with **random and uniform keys**, a lot of **related keys** appear **inside** it because of the checksum

Related-key setting

In general case, the security degrades to $\frac{k}{2}$-bit

How to improve security bounds?

Bottleneck

Even if keyed Streebog is used properly with **random and uniform keys**, a lot of **related keys** appear **inside** it because of the checksum

Related-key setting

In general case, the security degrades to $\frac{k}{2}$-bit

Example

- Query $\boldsymbol{g}$ with one input x under q different keys $K \boxplus \Sigma_1, \ldots, K \boxplus \Sigma_q$

How to improve security bounds?

Bottleneck

Even if keyed Streebog is used properly with **random and uniform keys**, a lot of **related keys** appear **inside** it because of the checksum

Related-key setting

In general case, the security degrades to $\frac{k}{2}$-bit

Example

- Query $\boldsymbol{g}$ with one input x under q different keys $K \boxplus \Sigma_1, \ldots, K \boxplus \Sigma_q$
- Store outputs $y_1, \ldots, y_q$ in memory, $y_i = \boldsymbol{g}(K \boxplus \Sigma_i, x)$

How to improve security bounds?

Bottleneck

Even if keyed Streebog is used properly with **random and uniform keys**, a lot of **related keys** appear **inside** it because of the checksum

Related-key setting

In general case, the security degrades to $\frac{k}{2}$-bit

Example

- Query $\boldsymbol{g}$ with one input x under q different keys $K \boxplus \Sigma_1, \ldots, K \boxplus \Sigma_q$
- Store outputs $y_1, \ldots, y_q$ in memory, $y_i = \boldsymbol{g}(K \boxplus \Sigma_i, x)$
- Repeat t times: guess $\tilde{K}$, compute $\tilde{y} = \boldsymbol{g}(\tilde{K}, x)$, check $\tilde{y} \in \{y_1, \ldots, y_q\}$
- If $\tilde{y} = y_i$ then $\tilde{K} = K \boxplus \Sigma_i$ and all keys are revealed

How to improve security bounds?

Bottleneck

Even if keyed Streebog is used properly with **random and uniform keys**, a lot of **related keys** appear **inside** it because of the checksum

Related-key setting

In general case, the security degrades to $\frac{k}{2}$-bit

Example

- Query $\boldsymbol{g}$ with one input x under q different keys $K \boxplus \Sigma_1, \ldots, K \boxplus \Sigma_q$
- Store outputs $y_1, \ldots, y_q$ in memory, $y_i = \boldsymbol{g}(K \boxplus \Sigma_i, x)$
- Repeat t times: guess $\tilde{K}$, compute $\tilde{y} = \boldsymbol{g}(\tilde{K}, x)$, check $\tilde{y} \in \{y_1, \ldots, y_q\}$
- If $\tilde{y} = y_i$ then $\tilde{K} = K \boxplus \Sigma_i$ and all keys are revealed
- The attack is successful if $t \cdot q = 2^k$, optimum at $t = q = 2^{\frac{k}{2}}$

How to improve security bounds?

Bottleneck

Even if keyed Streebog is used properly with **random and uniform keys**, a lot of **related keys** appear **inside** it because of the checksum

Related-key setting

In general case, the security degrades to $\frac{k}{2}$-bit

$\Rightarrow$ we should develop a more subtle related-key model and prove that this is sufficient for keyed Streebog

Observation

Suppose we query **different** x_1,...,x_q (instead of one x)

under corresponding keys $K \boxplus \Sigma_1$,...,$K \boxplus \Sigma_q$

$$y_i = \boldsymbol{g}(K \boxplus \Sigma_i, x_i)$$

Before the guessing, we can choose **only one** x_i and one "target" $K \boxplus \Sigma_i$,

instead of any from $\{K \boxplus \Sigma_1,...,K \boxplus \Sigma_q\}$ in the general case

The success probability is $\approx t \cdot 2^{-k}$ and does not depend on q

The situation is similar for the provable security approach

Detailed *PRF-RKA* model

PRF-RKA

$$\mathrm{Adv}_{\mathrm{g}}^{\mathit{PRF\text{-}RKA}_{\boxplus}}(\mathcal{A}) = \Pr\left(K \xleftarrow{\mathrm{R}} \boldsymbol{K}; \mathcal{A}^{\mathrm{g}_{K\boxplus\cdot}(\cdot)} \Rightarrow 1\right) -$$

$$- \Pr\left(K \xleftarrow{\mathrm{R}} \boldsymbol{K}; \mathrm{R}_i \xleftarrow{\mathrm{R}} \mathrm{Func}(\boldsymbol{X}, \boldsymbol{Y}), \forall i \in \boldsymbol{K}; \mathcal{A}^{\mathrm{R}_{K\boxplus\cdot}(\cdot)} \Rightarrow 1\right)$$

The query (x, κ) from $\mathcal{A}$ is the pair (input, relation)

The resources of $\mathcal{A}$:

– t computations; q queries to the oracle; r related keys;

– d different relations queried with the same x ($d \leq r \leq q$).

Detailed *PRF-RKA* model

PRF-RKA

$$\mathrm{Adv}_{\mathrm{g}}^{PRF\text{-}RKA_{\boxplus}}(\mathcal{A}) = \Pr\left(K \xleftarrow{\mathrm{R}} \boldsymbol{K}; \mathcal{A}^{\mathrm{g}_{K\boxplus\cdot}(\cdot)} \Rightarrow 1\right) -$$

$$- \Pr\left(K \xleftarrow{\mathrm{R}} \boldsymbol{K}; \mathrm{R}_i \xleftarrow{\mathrm{R}} \mathrm{Func}(\boldsymbol{X}, \boldsymbol{Y}), \forall i \in \boldsymbol{K}; \mathcal{A}^{\mathrm{R}_{K\boxplus\cdot}(\cdot)} \Rightarrow 1\right)$$

The query (x, κ) from $\mathcal{A}$ is the pair (input, relation)

The resources of $\mathcal{A}$:

– t computations; q queries to the oracle; r related keys;

– d different relations queried with the same x ($d \leq r \leq q$).

Heuristic bound

$$\mathrm{Adv}_{\mathrm{g}}^{PRF\text{-}RKA_{\boxplus}}(t, q, r, d) \lessapprox \frac{t \cdot d}{2^k} \leq \frac{t \cdot r}{2^k} \leq \frac{t \cdot q}{2^k}$$

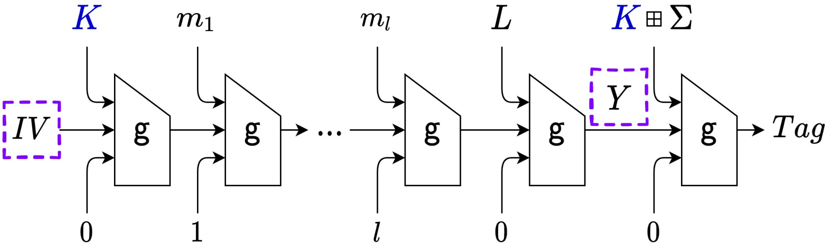
Intuition behind the proof

A new model by itself is not enough, we need a completely new proof...

Intuition behind the proof

A new model by itself is not enough, we need a completely new proof...

- The values $Y_1, ..., Y_q$ are "almost random"



- If there is no collision in $(IV, Y_1, ..., Y_q)$, then all inputs to $g(K \boxplus \Sigma, \cdot)$ are different ($d = 1$)
- Otherwise, the attacker has already achieved his goal, and we increase Adv by the collision probability

Theorem (PRF-security of Streebog-K)

$$\mathrm{Adv}^{PRF}_{\text{Streebog-K}}(t, q, l) \leq$$

$$\leq \mathrm{Adv}^{PRF\text{-}RKA_{\boxplus}}_{g^{\triangledown}}(t', q', q', d = 1) + \mathrm{Adv}^{PRF}_{\text{Csc}}(t', q, l') + \frac{q^2 + q}{2^{n+1}},$$

t – computation resources of the adversary ($t' \approx t$)

q – number of adaptively chosen messages ($q' = q + 1$)

l – maximum length of the message (in n-bit blocks)

Theorem (PRF-security of HMAC-Streebog)

$$\mathrm{Adv}^{PRF}_{\text{HMAC-Streebog}}(t, q, l) \leq$$

$$\leq \mathrm{Adv}^{PRF\text{-}RKA_{\boxplus\circ\oplus}}_{\mathrm{g}^{\triangledown}}(t', q', q', d = 2) +$$

$$+ \mathrm{Adv}^{PRF}_{\text{Csc}}(t', q, l') + \mathrm{Adv}^{PRF}_{\text{Csc}}(t', q, l'_\tau) + \frac{2q^2 + q}{2^n} + \frac{q^2}{2^{\tau+1}},$$

t – computation resources of the adversary ($t' \approx t$)

q – number of adaptively chosen messages ($q' = 2q + 2$)

l – maximum length of the message (in n-bit blocks), $l'_\tau \in \{2, 3\}$

$\tau \in \{256, 512\}$ – bit length of the output

Heuristic bounds: Streebog-K and HMAC-Streebog-512

The probability of at least one successful forgery with ν attempts

$$\Pr(\mathit{forgery}) \lessapprox \frac{t}{2^k} + \frac{t \cdot q \cdot l}{2^{n-1}} + \frac{q^2}{2^{n-1}} + \frac{\nu}{2^\tau}$$

t – computation resources of the adversary

q – number of adaptively chosen messages

l – maximum length of the message (in n-bit blocks)

τ – bit length of the tag ($\tau \leq n$)

k – bit length of the key ($k \leq n$)

Corollary: Streebog-K and HMAC-Streebog-512

Suppose that:

- 1. the amount of the processed blocks $q \cdot l < 2^{n-k}$
- 2. the tag is no shorter than the key ($\tau \geq k$)

and also recall that:

- 3. the amount of the processed blocks is less than key space $q \cdot l < 2^k$

Corollary: Streebog-K and HMAC-Streebog-512

Suppose that:

- 1. the amount of the processed blocks $q \cdot l < 2^{n-k}$
- 2. the tag is no shorter than the key $(\tau \geq k)$

and also recall that:

- 3. the amount of the processed blocks is less than key space $q \cdot l < 2^k$

The bound is simplified to

$$\Pr(\mathit{forgery}) \lessapprox \frac{t}{2^k} + \underbrace{\cancel{\frac{t \cdot q \cdot l}{2^{n-1}}}}_{(1)} + \underbrace{\cancel{\frac{q^2}{2^{n-1}}}}_{(1,\ 3)} + \underbrace{\cancel{\frac{\nu}{2^{\tau}}}}_{(2)} \approx \frac{t}{2^k}$$

Corollary: Streebog-K and HMAC-Streebog-512

Suppose that:

- 1. the amount of the processed blocks $q \cdot l < 2^{n-k}$
- 2. the tag is no shorter than the key ($\tau \geq k$)

and also recall that:

- 3. the amount of the processed blocks is less than key space $q \cdot l < 2^k$

The bound is simplified to

$$\Pr(forgery) \lessapprox \frac{t}{2^k} + \underbrace{\cancel{\frac{t \cdot q \cdot l}{2^{n-1}}}}_{(1)} + \underbrace{\cancel{\frac{q^2}{2^{n-1}}}}_{(1,\ 3)} + \underbrace{\cancel{\frac{\nu}{2^{\tau}}}}_{(2)} \approx \frac{t}{2^k}$$

$\Rightarrow$ Streebog-K and HMAC-Streebog-512 are "k-bit secure"

up to 2^{n-k} processed blocks of data

Corollary: Streebog-K and HMAC-Streebog-512

$k \leq \frac{n}{2} = 256$

The only effective way to forge/distinguish is by guessing the key

Corollary: Streebog-K and HMAC-Streebog-512

$k \leq \frac{n}{2} = 256$

The only effective way to forge/distinguish is by guessing the key

$k = \frac{3}{4}n = 384$

Up to $\approx 2^{128}$ processed blocks,

the only effective way to forge/distinguish is by guessing the key

Corollary: Streebog-K and HMAC-Streebog-512

$k \leq \frac{n}{2} = 256$

The only effective way to forge/distinguish is by guessing the key

$k = \frac{3}{4}n = 384$

Up to $\approx 2^{128}$ processed blocks,

the only effective way to forge/distinguish is by guessing the key

$k = n = 512$

Even after processing a single L-block message, the probability of forgery is extremely small, but L times greater than the "ideal" one

Corollary: Streebog-K and HMAC-Streebog-512

$k \leq \frac{n}{2} = 256$

The only effective way to forge/distinguish is by guessing the key

$k = \frac{3}{4}n = 384$

Up to $\approx 2^{128}$ processed blocks,

the only effective way to forge/distinguish is by guessing the key

$k = n = 512$

Even after processing a single L-block message, the probability of forgery is extremely small, but L times greater than the "ideal" one

For $k > \frac{n}{2} = 256$ and beyond the "2^{n-k} bound", the probability of forgery is greater than "ideal", but **negligible for most practical cases**

Corollary: HMAC-Streebog-256

If 256-bit Streebog is used, then

HMAC **narrows** the state after the first hashing from n to $\frac{n}{2}$ bits.

Due to the "internal" collision, the bound is increased by $\frac{q^2}{2^{\frac{n}{2}+1}}$.

Corollary: HMAC-Streebog-256

If 256-bit Streebog is used, then

HMAC **narrows** the state after the first hashing from n to $\frac{n}{2}$ bits.

Due to the "internal" collision, the bound is increased by $\frac{q^2}{2^{\frac{n}{2}+1}}$.

$\Rightarrow$ HMAC-Streebog-256 is "k-bit secure" if

1. the amount of blocks $q \cdot l < 2^{n-k}$
2. the amount of messages $q < 2^{\frac{n}{2}-k}$

Corollary: HMAC-Streebog-256

If 256-bit Streebog is used, then

HMAC **narrows** the state after the first hashing from n to $\frac{n}{2}$ bits.

Due to the "internal" collision, the bound is increased by $\frac{q^2}{2^{\frac{n}{2}+1}}$.

$\Rightarrow$ HMAC-Streebog-256 is "k-bit secure" if

1. the amount of blocks $q \cdot l < 2^{n-k}$
2. the amount of messages $q < 2^{\frac{n}{2}-k}$

Both conditions always hold for $k \leq \frac{n}{4} = 128$.

Short keys ($k \leq \frac{n}{2} = 256$) are not formally permitted for HMAC-Streebog.

Hence, the second condition is NOT fulfilled in practice.

Tightness of the bounds and attacks

Provable security – the upper bound:

$$\Pr(\mathit{forgery}) \lessapprox \frac{t}{2^k} + \frac{t \cdot q \cdot l}{2^{n-1}} + \frac{q^2}{2^{n-1}} + \frac{\nu}{2^\tau}$$

Tightness of the bounds and attacks

Provable security – the upper bound:

$$\Pr(\textit{forgery}) \lessapprox \frac{t}{2^k} + \frac{t \cdot q \cdot l}{2^{n-1}} + \frac{q^2}{2^{n-1}} + \frac{\nu}{2^\tau}$$

Each term in the upper bound corresponds to a term in the lower (probability of an attack):

1. Key guessing
2. "Tricky" attack through the imperfection of the cascade
3. Birthday-paradox forgery/distinguishing
4. Tag guessing

Tightness of the bounds and attacks

Provable security – the upper bound:

$$\Pr(\textit{forgery}) \lessapprox \frac{t}{2^k} + \frac{t \cdot q \cdot l}{2^{n-1}} + \frac{q^2}{2^{n-1}} + \frac{\nu}{2^\tau}$$

Each term in the upper bound corresponds to a term in the lower (probability of an attack):

1. Key guessing
2. "Tricky" attack through the imperfection of the cascade
3. Birthday-paradox forgery/distinguishing
4. Tag guessing

⇒ The obtained upper bounds are tight and cannot be further improved

“Keyed Streebog” vs “Keyed Sponge”

Streebog can be used without HMAC – the Streebog-K construction.

The same is true for the sponge-based hash functions (like SHA-3).

"Keyed Streebog" vs "Keyed Sponge"

Streebog can be used without HMAC – the Streebog-K construction.

The same is true for the sponge-based hash functions (like SHA-3).

If the state size of Streebog (n) = the capacity of sponge (c), then the security bounds for Streebog-K and "Keyed Sponge" **are the same**

Bertoni G., Daemen J., Peeters M., Van Assche G.
On the security of the keyed sponge construction – 2011

Typical warnings

- The proofs themselves use the "standard model" without heuristics, but all the statements about "k-bit security" are obtained under assumptions about the "good" properties of the compression function

Typical warnings

- The proofs themselves use the "standard model" without heuristics, but all the statements about "k-bit security" are obtained under assumptions about the "good" properties of the compression function
- Threats outside the model:
  - side-channel attacks
  - fault attacks
  - quantum computations
  - etc.

  The results do not say anything about these threats!
- All statements are only about adaptive chosen message attacks in the single-key setting and "classical" computations

Conclusion

1. New detailed *PRF-RKA* security model
   (pseudorandom function under related key attacks)

Conclusion

1. New detailed *PRF-RKA* security model
   (pseudorandom function under related key attacks)
2. New security proofs for HMAC-Streebog and Streebog-K through reduction to the *PRF-RKA* security of the compression function

Conclusion

1. New detailed *PRF-RKA* security model
   (pseudorandom function under related key attacks)
2. New security proofs for HMAC-Streebog and Streebog-K through reduction to the *PRF-RKA* security of the compression function
3. Streebog-K and HMAC-Streebog-512 are "k-bit secure" PRF up to 2^{n-k} processed blocks ($n = 512$ is the state size, $k \leq n$ is the key size)

Conclusion

1. New detailed *PRF-RKA* security model
   (pseudorandom function under related key attacks)
2. New security proofs for HMAC-Streebog and Streebog-K through reduction to the *PRF-RKA* security of the compression function
3. Streebog-K and HMAC-Streebog-512 are "k-bit secure" PRF
   up to 2^{n-k} processed blocks ($n = 512$ is the state size, $k \leq n$ is the key size)
4. Tightness: attacks match the provable security bounds

Thank you for attention!

Questions?

All reference implementations are available at

`https://gitflic.ru/project/vkir/streebog`