

SHA-3 Zoo::Shabal

Vorstellung des Shabal Algorithmus aus dem SHA-3 Zoo

Stephan Müller

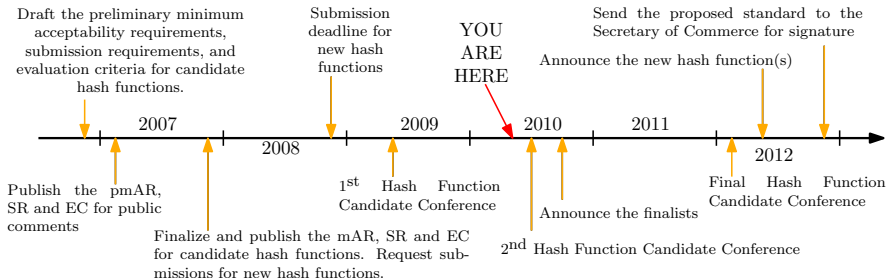
Institut für Informatik
Humboldt-Universität zu Berlin

01.06.2010

NIST's Cryptographic Hash Algorithm Competition

- NIST (National Institute of Standards and Technology) sucht nach einem neuen, besseren Hash-Algorithmus:

website: *NIST has opened a public competition to develop a new cryptographic hash algorithm, which converts a variable length message into a short "message digest" that can be used for digital signatures, message authentication and other applications. The competition is NIST's response to recent advances in the cryptanalysis of hash functions. The new hash algorithm will be called "SHA-3" and will augment the hash algorithms currently specified in FIPS 180-2, Secure Hash Standard.*



NIST's cryptographic hash Algorithm Competition

Was ist der SHA-3 Zoo?

- Zur Zeit sind noch 14 Kandidaten im Wettbewerb:
 - BLAKE
 - Grøstl
 - Shabal
 - BLUE MIDNIGHT WISH
 - Hamsi
 - SHAvite-3
 - CubeHash
 - JH
 - SIMD
 - ECHO
 - Keccak
 - Skein
 - Fugue
 - Luffa

NIST's cryptographic hash Algorithm Competition

Shabal - the caveman?

- Zur Zeit sind noch 14 Kandidaten im Wettbewerb:

- BLAKE
- Grøstl
- **Shabal**
- BLUE MIDNIGHT WISH
- Hamsi
- SHAvite-3
- CubeHash
- JH
- SIMD
- ECHO
- Keccak
- Skein
- Fugue
- Luffa



Shabal wurde vom französischen *Saphir-Projekt* (Security and Analysis of Hash Primitives) eingereicht und ist nach dem französischen Rugbyspieler *Sébatien Chabal* benannt.

Fahrplan

- Einführung
 - NIST's cryptographic hash Algorithm Competition
 - SHA-3 Zoo
- Mode of Operation
 - Nachrichtenrunde
 - Verkettung
 - Finalrunden
 - Nachrichtenpadding
- Keyed Permutation
 - Idee
 - Exkurs: NLFSR
 - Berechnung
- Sicherheitsziele
- Mode of Operation und innere Kollisionen
- Sicherheitsbehauptungen
- beweisbare Sicherheitseigenschaften
 - Vorbereitene Definitionen
- exemplarisch eine beweisbare Sicherheitsaussage
- ENDE
 - Letzte Gelegenheit für Fragen

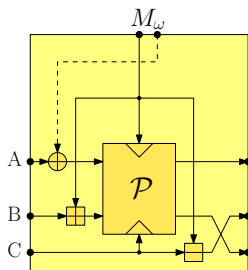
Shabal

Charakteristika

- **Eingabe:** Nachricht $M = (M_1, M_2, \dots, M_k)$ beliebiger Länge ($< 2^{73}$ bit), aufgeteilt in k 512-bit Blöcke
- **Ausgabe:** l_h -bit großer Hashwert $\mathcal{H}(M)$. Dabei ist $l_h \in \{192, 224, 256, 384, 512\}$.
- orientiert sich am klassischem MD-Design
- sehr einfacher *Mode of Operation*
- Herzstück ist eine *keyed Permutation* \mathcal{P}
- Sehr ausführliche Dokumentation (300 Seiten) vgl. [?]
- *mixed Endianness*: *big endian* auf Bitlevel, *little endian* auf Bytelevel - vgl. MD5

Mode of Operation

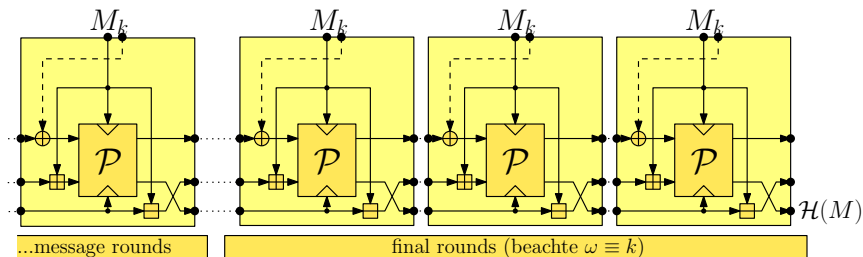
Grundbaustein: Messageround



- ω Zähler, $|\omega| = 64$ bit
 - M_ω - Nachrichtenblock ω
 - A, B, C - interne Variablen
 - $|A| = 384$ bit = 12 Worte à 32 bit
 $|M_\omega| = |B| = |C| = 512$ bit = 16 Worte à 32 bit
 - $\mathcal{P} = \mathcal{P}_{(M_\omega, C)} \in \text{Sym}(\{0, 1\}^{896})$ permutiert (A', B') in Abhängigkeit vom Schlüssel (M_ω, C) .
 - \boxplus und \boxminus bezeichnen *wortweise* Addition bzw. Subtraktion modulo 2^{32} (*little endian* auf Bytelevel)
 - \oplus ist XOR. Benutze die ersten zwei Worte von A
-
- Sowohl M_ω als auch ω gehen in die Berechnung ein
 - Der Nachrichtenblock fließt an mehreren Stellen ein (*multiple message insertion*)

Mode of Operation

Finalrunden



Es gibt 3 Finalrunden. Diese unterscheiden sich nur durch den Zähler ω von normalen Nachrichtenrunden. Man beachte, dass hier optimiert werden kann. Die Nachricht wird erst addiert und dann wieder subtrahiert. Der C Speicher enthält am Ende den Hashwert. A und B werden verworfen.

Nachrichtenpadding und Initialisierung

- Klassisches *Padding*: Die Nachricht wird am Ende mit $10^n, 0 \leq n \leq 511$ soweit verlängert, dass die Gesamtlänge anschließend ein Vielfaches von 512 ist.
- Padding findet in jedem Fall statt, auch wenn die Nachricht bereits passende Länge hat.
- Initialisierung der Speicher A, B und C wird durch *Prepadding* der Nachricht durch zwei zusätzliche Blöcke erreicht
 - $\omega = -1$
 - $A = B = C = 0$
 - $M_{-1} = (l_h, l_h + 1, \dots, l_h + 15)$ und $M_0 = (l_h + 16, \dots, l_h + 31)$
- Alternativ kann dieser Status im Voraus berechnet und als *Initialisierung* benutzt werden.
- Unterschiedliche Werte von l_h führen zu unterschiedlichen Initialisierungen.

Keyed Permutation $\mathcal{P}_{(M_\omega, C)} : \mathcal{A} \times \mathcal{B} \rightarrow \mathcal{A} \times \mathcal{B}$

Idee: entangled NLFSRs

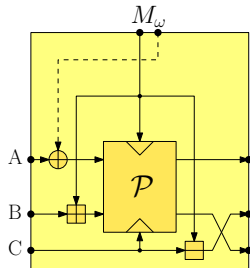
Sei $\mathcal{M} = \mathcal{B} = \mathcal{C} = \{0, 1\}^{512}$ und $\mathcal{A} = \{0, 1\}^{384}$.

\mathcal{P} ist eine Abbildung

$$\mathcal{P} : \mathcal{M} \times \mathcal{C} \times \mathcal{A} \times \mathcal{B} \rightarrow \mathcal{A} \times \mathcal{B}$$

Für feste Werte $(M_\omega, C) \in \mathcal{M} \times \mathcal{C}$ ist $\mathcal{P}_{(M_\omega, C)}$ eine **Permutation**:

$$\begin{aligned} \mathcal{P}_{(M_\omega, C)} : \mathcal{A} \times \mathcal{B} &\longrightarrow \mathcal{A} \times \mathcal{B} \\ (A, B) &\longmapsto \mathcal{P}(M_\omega, C, A, B) \end{aligned}$$



- Zur Berechnung werden die Eingaben als nichtlinear rückgekoppelte Shiftregister aufgefasst. Bestehend aus 12 bzw. 16 Worten à 32 bit.
- (M_ω, C) ist als Schlüssel zu verstehen und wird nicht verändert.
- **Idee:** Die Feedbackfunktionen hängen von den anderen Registern ab. So entstehen verschränkte NLFSRs.

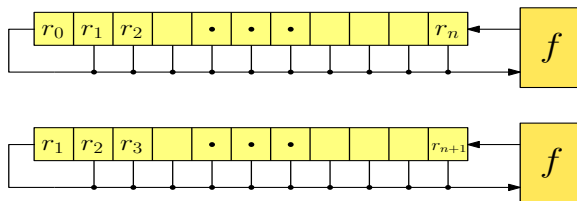
Exkurs: [nicht] linear rückgekoppelte Schieberegister

linear feedback shiftregister (LFSR) and non-linear feedback shiftregister (NLFSR)

- Sei G eine Gruppe. Ein *linear rückgekoppeltes Schieberegister* ist ein G -Register $\mathcal{R} \in G^{n+1}$ bestehend aus n Taps (r_0, \dots, r_n) und einer *linearen Feedbackfunktion*

$$f : G^{n+1} \rightarrow G \quad (r_0, \dots, r_n) \mapsto r_{n+1}.$$

Nach einer *Runde (Takt)* beinhaltet \mathcal{R} die Werte (r_1, \dots, r_{n+1}) .



- Im nichtlinearen Fall ist die Updatefunktion f nicht linear
- Im Folgenden ist $G = (\mathbb{Z}_{32}, +) = (\{0, 1\}^{32}, \oplus)$

Keyed Permutation $\mathcal{P}_{(M_\omega, C)} : \mathcal{A} \times \mathcal{B} \rightarrow \mathcal{A} \times \mathcal{B}$

Berechnung: Initiale Runden - Hauptrunden - Finale Transformation

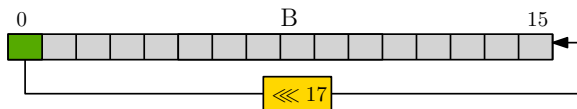
- Die Berechnung von $\mathcal{P}_{(M_\omega, C)}$ erfolgt in drei Schritten:
 - Initiale Runden
 - Hauptrunden
 - Finale Transformation
- Im Kontext der Berechnung von \mathcal{P} wird der eingehende Nachrichtenblock M_ω mit M bezeichnet.
Es ist $M_\omega = M = (M_0, \dots, M_{15})$.
- Jede einzelne Runde bzw. Subrunde ist invertierbar, so dass $\mathcal{P}_{(M, C)}$ tatsächlich eine Permutation ist.

Keyed Permutation $\mathcal{P}_{(M,C)} : \mathcal{A} \times \mathcal{B} \rightarrow \mathcal{A} \times \mathcal{B}$

Berechnung: **Initiale Runden** - Hauptrunden - Finale Transformation

- 16 Runden:

$$B_{16} = B_0 \lll 17$$

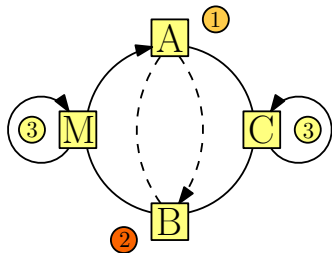


- Äquivalent: **forall** $0 \leq i \leq 15$ **do**: $B_i = B_i \lll 17$
- Dieser Schritt ist umkehrbar, etwa mit $\ggg 17$ oder $\lll 15$.

Keyed Permutation $\mathcal{P}_{(M,C)} : \mathcal{A} \times \mathcal{B} \rightarrow \mathcal{A} \times \mathcal{B}$

Berechnung: Initiale Runden - **Hauptunden (1)** - Finale Transformation

- **Idee:** Die Feedbackfunktionen von A und B hängen von den jeweils anderen Registern ab.
- Jede einzelne Runde ist umkehrbar (Beweis!)
- Die Multiplikationen werden modulo 2^{32} berechnet
- 32 Runden:



$$A_{12} = 3 \cdot (A_0 \oplus 5 \cdot (A_{11} \lll 15) \oplus C_8) \oplus B_{13} \oplus (B_9 \wedge \overline{B_6}) \oplus M_0$$

$$B_{16} = (B_0 \lll 1) \oplus \overline{A_{11}}$$

$$M_{16} = M_0$$

$$C_{-1} = C_{15} \quad (\text{C rotiert entgegengesetzt})$$

Keyed Permutation $\mathcal{P}_{(M,C)} : \mathcal{A} \times \mathcal{B} \rightarrow \mathcal{A} \times \mathcal{B}$

Berechnung: Initiale Runden - **Haupttrunden (2)** - Finale Transformation

Die Haupttrunden als Bild:

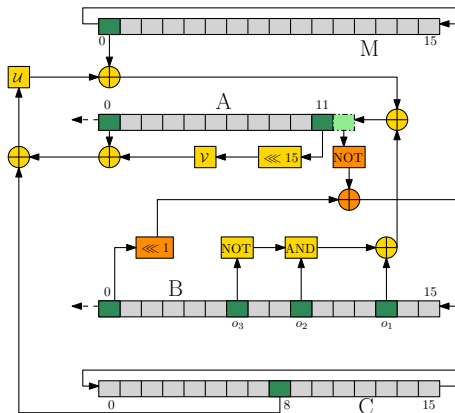
$$A_r = \mathcal{U}(A_0 \oplus \mathcal{V}(A_{r-1} \lll 15) \oplus C_8) \oplus B_{o_1} \oplus (B_{o_2} \wedge \overline{B_{o_3}}) \oplus M_0$$

$$B_{16} = (B_0 \lll 1) \oplus \overline{A_r}$$

$$M_{16} = M_0$$

$$C_{-1} = C_{15}$$

- 32 Runden
- $(o_1, o_2, o_3) = (13, 9, 6)$
- $\mathcal{U}(x) = 3x \pmod{2^{32}}$
- $\mathcal{V}(x) = 5x \pmod{2^{32}}$



- Aus Performancegründen werden in einer Runde nur sehr wenig Taps benutzt. Die Auswahl ist wohlüberlegt.

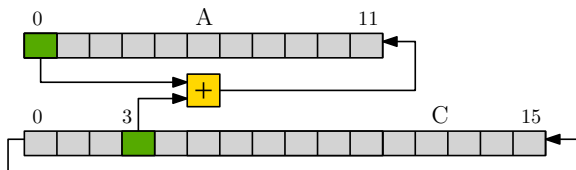
Keyed Permutation $\mathcal{P}_{(M,C)} : \mathcal{A} \times \mathcal{B} \rightarrow \mathcal{A} \times \mathcal{B}$

Berechnung: **Initiale Runden** - **Haupt**runden - **Finale Transformation**

- 36 Runden: $j = 0, \dots, 35$

$$A_j = A_j \boxplus C_{3+j} \quad (\text{wortweise Addition})$$

- Indizes sind modulo 12 bzw modulo 16 zu verstehen
- Die Finalrunden sind invertierbar
- Dies ist auch wieder durch Schieberegister realisierbar, etwa durch



und anschließende einfache Rotationen.

- Die Finalrunden dienen zur Stärkung von $\mathcal{P}_{(M,C)}^{-1}$

(Fragen!)

Teil II: Sicherheitsaspekte

Security Claims and Security Proofs

Sicherheitsziele

Für eine (kryptographische) Hashfunktion \mathcal{H} hat man drei klassische Sicherheitsziele:

- **starke Kollisionsresistenz** (*collision resistance*)
 - Es soll schwer sein, zwei Nachrichten M und M' zu finden, sodass gilt $\mathcal{H}(M) = \mathcal{H}(M')$.
- \mathcal{H} ist eine **Einwegfunktion** (*preimage resistance*)
 - Es soll schwer sein, zu einem gegebenen Hashwert h eine Nachricht M zu finden, sodass gilt $\mathcal{H}(M) = h$.
- **schwache Kollisionsresistenz** (2^{nd} *preimage resistance*)
 - Es soll schwer sein, zu einer Nachricht M eine weitere Nachricht M' zu finden, sodass gilt $\mathcal{H}(M) = \mathcal{H}(M')$.

Es ist hierbei essentiell zu klären, was sich hinter dem Begriff schwer verbirgt

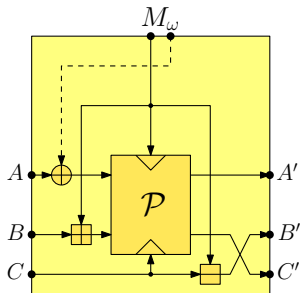
Mode of Operation

innere Kollisionen

- Um Kollisionen zu finden, muss man zunächst *innere Kollisionen* finden.
- innere Kollisionen sind auch für *length extensions attacks* interessant
 - $\mathcal{H}(M_1) = \mathcal{H}(M_2)$ impliziert $\mathcal{H}(M_1||M) = \mathcal{H}(M_2||M)$
 - Viele Kollisionen zum Preis von einer!
 - Genauer: gegeben $\mathcal{H}(M), |M|$ kann man $\mathcal{H}(M||M')$ berechnen?
- Es gibt keine inneren Kollisionen, die von nur einer Nachrichtenrunde erzeugt werden, also

$$(A', B', C', \omega + 1, M) = (A', B', C', \omega + 1, M') \Rightarrow M = M'$$

- Kollisionen über mehrere Runden zu bauen, erscheint schwierig!



Security Claims

Statement of the expected strenght

Im Wettbewerbsbeitrag finden sich Beweise, in welchem Rahmen Shabal die Sicherheitsanforderungen erfüllt. Die Beweise münden in die folgenden **Security Claims**:

- 1 Das Finden einer *Kollision* hat Aufwand mind. $2^{l_h/2}$
 - 2 Das Finden eines *Urbilds* hat Aufwand mind. 2^{l_h}
 - 3 Das Finden eines *zweiten Urbilds* für Nachrichten der Länge $\leq 2^k$ hat Aufwand mind. 2^{l_h-k}
 - 4 Jede *length extension attack* hat Aufwand mind. 2^{256}
- Aufwand bezeichnet dabei die Zahl der Auswertungen von Nachrichtenrunden
 - $l_h \in \{192, 224, 256, 384, 512\}$
 - Nachrichtenlänge ist auf 2^{64} bit beschränkt

Beweisbare Sicherheitsbehauptungen

Die genaue Formulierung der beweisbaren Sicherheitsbehauptungen benötigen ein Reihe an Definitionen und Konstruktionen. Die folgenden Begriffe sollten bekannt sein:

- ideale Blockchiffre (*Ideal Cipher*)
- Zufallsorakel (*Random Oracle*)
- Kryptosystem (*Cryptographic System*)
- Vergleiche der Sicherheit von Kryptosystemen
- Ununterscheidbarkeit (*Indistinguishability*) von Kryptosystemen
- Undifferenzierbarkeit (*Indifferentiability*) von Kryptosystemen

ROM vs ICM

Random Oracle Model and Ideal Cipher Model

- Man stelle sich *ideale* Hashfunktionen oder Blockchiffren in Form einer Blackbox vor.

ROM: Ein Zufallsorakel (*Random Oracle*) ist eine Hashfunktion. Aus einer Anfrage in Form einer Nachricht ergibt sich ein rein zufällig gewählter Hashwert der Nachricht. Die Antworten sind gleichverteilt und unabhängig (und natürlich auch konsistent).

ICM: Eine ideale Chiffre (*Ideal Cipher*) ist eine Blockchiffre der Form $\mathcal{E} \in \text{Sym}(\mathcal{M})^{\mathcal{K}}$. Die Anfragen bestehen aus einem Schlüssel und einer Nachricht, die je nach Wunsch ver- oder entschlüsselt wird. Die Permutation \mathcal{E}_k wird in Abhängigkeit von $k \in \mathcal{K}$ zufällig gewählt.

\mathcal{M} - Nachrichten- und Kryptotextraum

\mathcal{K} - Schlüsselraum

Cryptographic Systems

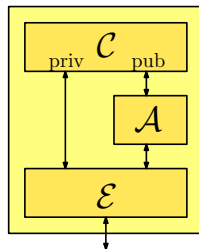
public and private Interfaces

- Ein Kryptosystem \mathcal{C} (*cryptographic system*) lässt sich als Folge von bedingten Wahrscheinlichkeitsverteilungen $(P_n)_{n \in \mathbb{N}}$ auffassen.
 $P_i = P_{Y_i | X^i Y^{i-1}}$ ist dann die Verteilung Y_i der i -ten Ausgabe unter der Bedingung, dass die Verteilungen $X_1, \dots, X_i, Y_1, \dots, Y_{i-1}$ der ersten i Eingaben und $i - 1$ Ausgaben bekannt sind. (vgl. [?])
- In der Regel hängen Kryptosysteme von einem Sicherheitsparameter $k \in \mathbb{N}$ ab. In diesem Fall betrachten wir eine Familie $(\mathcal{C}_k)_{k \in \mathbb{N}}$ von Systemen im obigen Sinne.
- Im Folgenden werden Kryptosysteme mit zwei verschiedenen Schnittstellen betrachtet
 - die *öffentliche Schnittstelle* ist jeder Partei frei zugänglich, über sie interagiert der Angreifer mit dem System.
 - die *private Schnittstelle* steht nur ehrlichen Parteien, also den eigentlichen Benutzern des Systems, zur Verfügung.
- Kryptosysteme können zu neuen Systemen zusammengesetzt werden, die Schnittstellen werden dabei entsprechend verbunden

Sicherheit von Kryptosystemen vergleichen

Seien $\mathcal{C} = (\mathcal{C}_k)_{k \in \mathbb{N}}$ und $\mathcal{C}' = (\mathcal{C}'_k)_{k \in \mathbb{N}}$ zwei Kryptosysteme.

- Ein (Krypto-)System mit Ausgabewerten aus $\{0, 1\}$ heißt Umgebung (*environment*).
- Angreifer und sonstige Parteien sind Algorithmen, aufgefasst als Systeme, die über die jeweiligen Schnittstellen mit Kryptosystemen interagieren.
- \mathcal{C} ist (**komplexitätstheoretisch**) **mindestens so sicher** wie \mathcal{C}' , falls es für jede Umgebung \mathcal{E} und für jeden Angreifer \mathcal{A} für \mathcal{C} einen Angreifer \mathcal{A}' für \mathcal{C}' gibt, sodass der Unterschied in den Verteilungen,
$$|\text{Prob}[\mathcal{E}(\mathcal{C}_k^{\text{priv}}, \mathcal{A}(\mathcal{C}_k^{\text{pub}})) = 1] - \text{Prob}[\mathcal{E}(\mathcal{C}'_k^{\text{priv}}, \mathcal{A}'(\mathcal{C}'_k^{\text{pub}})) = 1]|$$
vernachlässigbar in k ist (und $\mathcal{E}, \mathcal{A}, \mathcal{A}'$ effiziente Algorithmen sind).
- Bezeichnung $\mathcal{C} \succ \mathcal{C}'$ (bzw. $\mathcal{C} \succ_c \mathcal{C}'$)



Indistinguishability

Ununterscheidbarkeit von Kryptosystemen

- Seien $\mathcal{S} = (\mathcal{S}_k)_{k \in \mathbb{N}}$ und $\mathcal{T} = (\mathcal{T}_k)_{k \in \mathbb{N}}$ zwei Kryptosysteme. \mathcal{T} und \mathcal{S} heißen ununterscheidbar (*undistinguishable*), falls für jeden Angriffsalgorithmus \mathcal{D} (*=distinguisher*) der Vorteil,

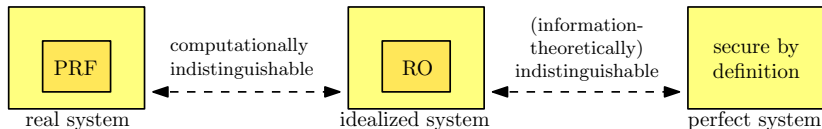
$$|\Pr[\mathcal{D}(\mathcal{S}_k) = 1] - \Pr[\mathcal{D}(\mathcal{T}_k) = 1]|$$

vernachlässigbar in k ist.

- Seien \mathcal{S} und \mathcal{T} ununterscheidbar. Für jedes Kryptosystem \mathcal{C} gilt:

$\mathcal{C}(\mathcal{S})$ ist genauso sicher wie $\mathcal{C}(\mathcal{T})$

- Beweisschema: [?, Maurer - Indistinguishability of random systems]



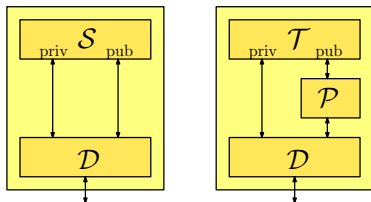
Indifferentiability

Verallgemeinerung von Indistinguishability

Problem:

$$\begin{aligned} \mathcal{T}, \mathcal{S} & \text{ ununterscheidbar} \\ \implies & \forall \mathcal{C}(\cdot) : \mathcal{C}(\mathcal{S}) \prec \mathcal{C}(\mathcal{T}) \end{aligned}$$

gilt nur falls Angreifer keine zusätzliche Information über \mathcal{S} und \mathcal{T} haben, also falls \mathcal{S} und \mathcal{T} keine öffentlichen Schnittstellen haben.



- Seien $\mathcal{S} = (\mathcal{S}_k)_{k \in \mathbb{N}}$ und $\mathcal{T} = (\mathcal{T}_k)_{k \in \mathbb{N}}$ zwei Kryptosysteme. \mathcal{S} heißt undifferenzierbar von \mathcal{T} (\mathcal{S} is indifferentiable from \mathcal{T}), falls für jeden Angriffsalgorithmus \mathcal{D} existiert ein System \mathcal{P} , sodass der Vorteil für \mathcal{D} ,

$$|\Pr[\mathcal{D}(\mathcal{S}_k^{priv}, \mathcal{S}_k^{pub}) = 1] - \Pr[\mathcal{D}(\mathcal{T}_k^{priv}, \mathcal{P}(\mathcal{T}_k^{pub})) = 1]|$$

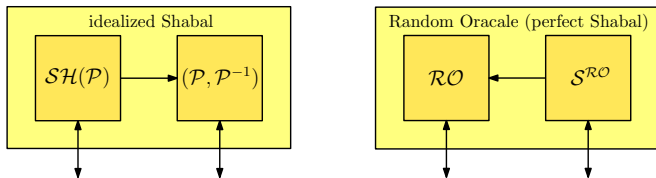
vernachlässigbar in k ist. Bezeichnung: $\mathcal{S} \sqsubset \mathcal{T}$.

- **Theorem:** Ohne Einschränkung an öffentliche Schnittstellen gilt:

$$\mathcal{S} \sqsubset \mathcal{T} \iff \forall \mathcal{C}(\cdot) : \mathcal{C}(\mathcal{S}) \prec \mathcal{C}(\mathcal{T})$$

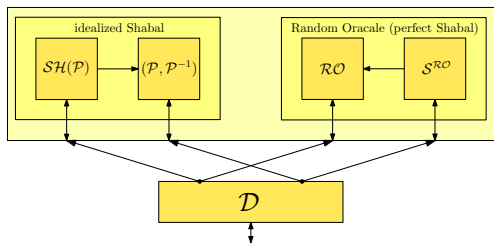
Anwendungen auf Shabal

- Shabal wird durch folgende Kryptosysteme dargestellt:



- $SH(\cdot)$ entspricht dem *Mode of Operation* von Shabal
- \mathcal{P} entspricht der idealisierten keyed Permutation, also einer idealen Blockchiffre
- \mathcal{RO} ist ein Zufallsorakel, eine perfekte Hashfunktion
- $\mathcal{S}^{(\cdot)}$ ist eine Simulation von \mathcal{P} unter Verwendung des Zufallsorakles
- Jeder, also auch der Angreifer, kann
 - Hashwerte berechnen (lassen)
 - die keyed Permutation \mathcal{P} berechnen
 - die Inverse \mathcal{P}^{-1} der keyed Permutation \mathcal{P} berechnen

Shabal is indifferentiable from a random Oracle






- **Theorem:** Es existiert eine Simulation \mathcal{S} von $(\mathcal{P}, \mathcal{P}^{-1})$, sodass jeder Distinguisher \mathcal{D} nach N Fragen an das rechte Interface einen Vorteil,

$$|\Pr[\mathcal{D}((\mathcal{P}, \mathcal{P}^{-1})) = 1] - \Pr[\mathcal{D}(\mathcal{S}^{\mathcal{RO}}) = 1]| \leq N(4N - 3)2^{-896}$$

hat. \mathcal{S} stellt dabei maximal N Fragen an \mathcal{RO} und läuft in Zeit $O(N^2)$.

- Daraus folgt: (idealisierter) Shabal ist bis für 2^{488} Auswertungen von \mathcal{P} oder \mathcal{P}^{-1} nicht von einem Zufallsorakel differenzierbar.

ENDE

-  Ueli Maurer.
Indistinguishability of random systems.
In In Advances in Cryptology — EUROCRYPT '02, volume 2332 of LNCS, pages 110–132. Springer-Verlag, 2002.
-  Ueli Maurer, Renato Renner, and Clemens Holenstein.
Indifferentiability, impossibility results on reductions, and applications to the random oracle methodology.
In Theory of Cryptography - TCC 2004, Lecture Notes in Computer Science, pages 21–39. Springer-Verlag, 2003.
-  Saphir project, www.shabal.com.
Shabal, a Submission to NISTs Cryptographic Hash Algorithm Competition, 1.0 edition, 01 2009.