Resilienced Kryptographie

Ruediger Weis, Christian Forler (34C3)

Leipzig, Dezember 2017

Wissenschaftlich starke Kryptographie ist auch für übermächtige Geheimdienste nicht brechbar.

Bruce Schneier, Guardian, 6. September 2013

"Trust the math. Encryption is your friend."

Intel ME

Gameover

Praktisch unauffindbarer Zugriff auf den gesamten Speicher.

- Black Hat 2017
- Mark Ermolov , Maxim Goryachy,
  - "HOW TO HACK A TURNED OFF COMPUTER, OR RUNNING UNSIGNED CODE IN INTEL ME"

Minix Worlddomination

Intel ME auf Minix 3

- Minix 3 läuft auf mehr Computersystemen, als Windows und Mac zusammen.

Open Source Booting

Core Boot

Libre Boot

U-Boot

. . .

NERF (Non-Extensible Reduced Firmware)

Replace your exploit-ridden firmware with a Linux kernel

- Google: Ron Minnich, Gan-shun Lim, Ryan O'Leary, Chris Koch, Xuan Chen
- Two Sigma: Trammell Hudson
- Cisco: Andrey Mirtchovski
- Splitted-Desktop: Jean-Marie Verdun, Guillaume Giamarchi

Einfache Mathematik

RSA

- Sicherheitsbeweise
- Verstandene Mathematik
- Kurze Impementierung

Primzahlverteilung

Anzahl der Primzahlen $\leq x$

$$\pi(x) \approx \frac{x}{\ln(x)}$$

Genaue Abschätzung

Für $x \geq 17$ gilt

$$\frac{x}{\ln(x)} < \pi(x) < 1,25506 \cdot \frac{x}{\ln(x)}$$

gpg Primzahlentests

- `https://www.gnupg.org/documentation/manuals/gcrypt/Prime_002dNumber_002dGenerator-Subsystem-Architecture.html`

Prime-Number-Generator Subsystem Architecture

The primality test works in three steps:

- The standard sieve algorithm using the primes up to 4999 is used as a quick first check.
- A Fermat test filters out almost all non-primes.
- A 5 round Rabin-Miller test is finally used. The first round uses a witness of 2, whereas the next rounds use a random witness.

Fermat Test (Wikipedia)

Repeat k times:

- Pick a randomly in the range [2, n - 2]
- If $a^{n-1} \neq 1 \bmod n$
  - then return composite

If composite is never returned: return probably prime

Rabin Miller Test (Wikipedia)

```
#include <stdint.h>
bool mrt (const uint32_t n, const uint32_t a) { // n ungerade, 1 < a < n-1
   const uint32_t n1 = n - 1;
   uint32_t d = n1 >> 1;
   int j = 1;
   while ((d & 1) == 0) d >>= 1, ++j;
   uint64_t t = a, p = a;
   while (d >>= 1) { //square and multiply: a^d mod n
      p = p*p % n;
      if (d & 1) t = t*p % n;
   }
   if (t == 1 || t == n1) return true; // n ist wahrscheinlich prim
   for (int k=1 ; k<j ; ++k) {
      t = t*t % n;
      if (t == n1) return true;
      if (t <= 1) break;
   }
   return false; // n ist nicht prim
}
```

Was soll da schon schief gehen?

Fast Prime Generation

- Keine gute Idee.

Angriffe aus dem letzten Jahrtausend

Preiswerter Totalschaden

- Coppersmith (1996)
- Matus Nemec, Marek Sys, Petr Svenda (2017)
  - The Return of Coppersmith's Attack: Practical Factorization of Widely Used RSA Moduli

512 bit: $0,002

1024 bit: $1,78

2048 bit: $944

3072 bit: $1,90 * 10^{26}

4096 bit: $8,48 * 10^{9}

TPM Schlüssel

Langlebige Schlüssel: Storage Root Key

- Völlig unnötiger Weise unsichere "schnelle" Schlüsselgenerierung.

TPM Schlüssellänge <= 2048

- Neuer Standard mit sicheren Schlüssellängen

TPM Desaster

- Infineon TPM
  - Problemen nicht bei BSI Zertifizierung entdeckt.
  - Google Cromebooks
- ...
- [Nemec Sys Svenda 2017]

Domain name	Analyzed datasets	# Vuln. keys/devices	% Vulnerable
Complete/larger-scale datasets			
Certification authorities	all browser-trusted roots (173), level ⩽ 3 intermediates (1,869)	0 keys	0
ePass signing certificates	ICAO Document Signing Certificates, CSCA Master Lists	0 keys	0
Estonian eID	sample of 130,152 randomly selected citizens	71,417 keys	54.87
Estonian mobile eID	sample of 30,471 randomly selected citizens	0 keys	0
Estonian e-residents	sample of 4,414 e-residents	4,414 keys	100
Message security (PGP)	complete PGP key server dump (9 M)	2,892 keys	0.03
Software signing (GitHub)	SSH keys for GitHub developers (4.7 M)	447 keys	0.01
Software signing (Maven)	signing keys for all public Maven artifacts	5 keys	0.003
TLS/HTTPS	complete IPv4 scan, Certificate Transparency	15 keys	<0.001
Trusted boot (TPM)	41 laptops with different chips by 6 TPM manufacturers	10 devices	24.39

Tuwat: Open Source TPM Hardware

Exkurs: RSA Padding

- Bleichenbacher (1998)
- Robot (2017)
  - Hanno Böck, Juraj Somorovsky, Craig Young
    - Return Of Bleichenbacher's Oracle Threat
  - `https://robotattack.org/`

Who is affected?

We have identifed vulnerable implementations from at least seven vendors including F5, Citrix, and Cisco. (Current patch status is listed below.)
Some of the most popular webpages on the Internet were affected, including Facebook and Paypal. In total, we found vulnerable subdomains on 27 of the top 100 domains as ranked by Alexa.

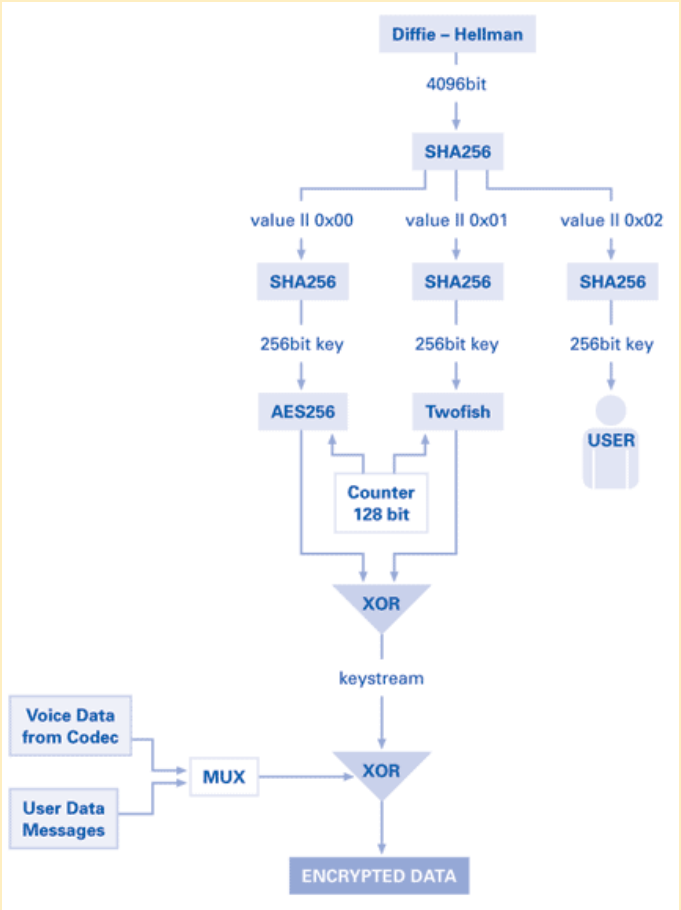
Robuste Krypto

Viel hilft viel.

- XOR ist Dein Freund.
- Doppel Hash
  - Aber richtig.
- Längere Schlüssellängen
- ...

Cryptophone (2003)

Cryptophone Kryptodesign (Weis 2003)



Safer Use of Hash

Lieber SHA-512 nutzen, auch wenn man nicht alle 512 bit braucht.

- Mehr Runden und 64-bit Operationen
- Abschneiden ist ok. (Vergleiche SHA-384.)
- Schön randomisierte bit kann man immer brauchen.

SHA-512 ist schnell

- SHA-512 ist auf alten Plattformen etwas langsamer.
- Auf 64-bit Architekturen ist SHA-512 deutlich schneller.

Tuwat: Nutzt SHA-512 oder SHA-3.

Längere Kurven

256 Bit ECC reicht nicht.

Besser größere Schlüssellängen

- Tuwat: Forschungsbedarf

Resilienced Kryptographie

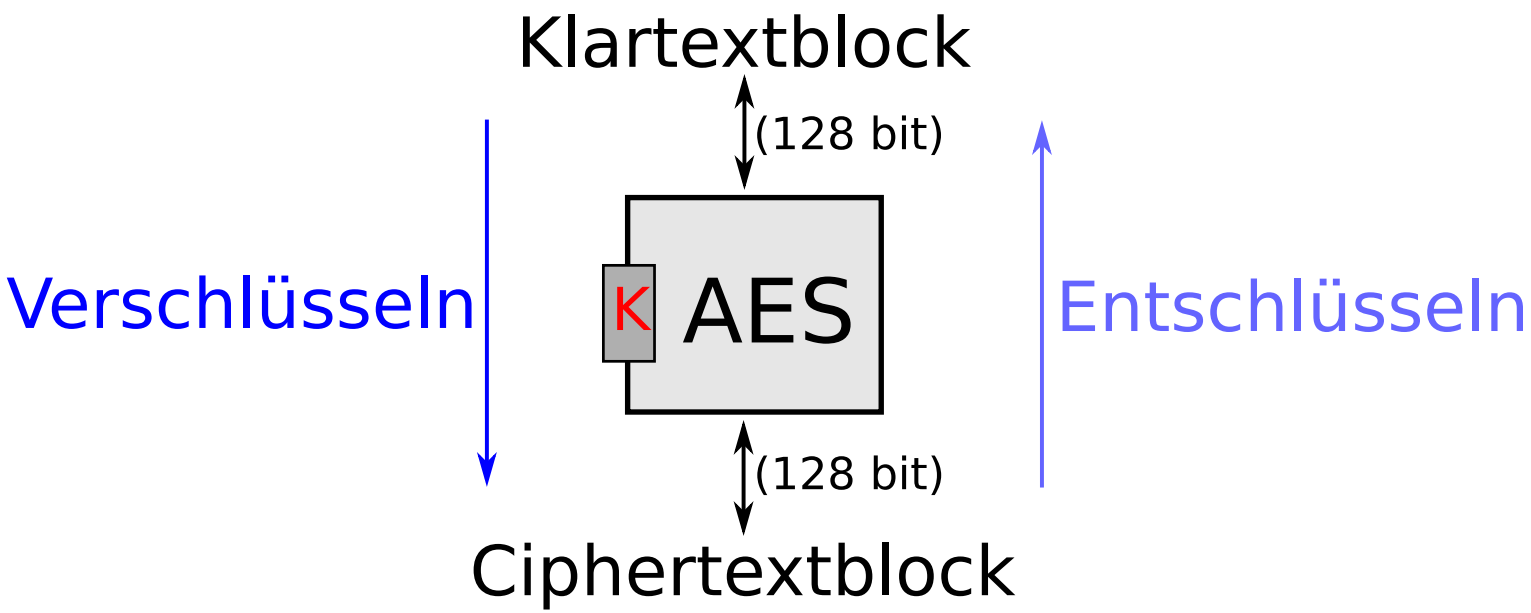
Robuste Kryptographie

Fehlbedienungsfreundliche Kryptographie

Security by Namedropping

- Hersteller bewerben die Sicherheit ihrere Softwareprodukte mittels AES.
  - Military Grade Encryption (AES 256 bit)
  - Everything is encrypted using AES-256
  - Voice and data encrypted with AES 256
  - ...
- Aussagekraft geht gegen Null.
  - Der naive Einsatz von AES macht Software nicht sicher
  - Integrität der Daten ist oftmals nicht gewährleistet
  - Es ist völlig unklar wie AES eingesetzt wird
  - Richtiger Einsatz ist nicht trivial

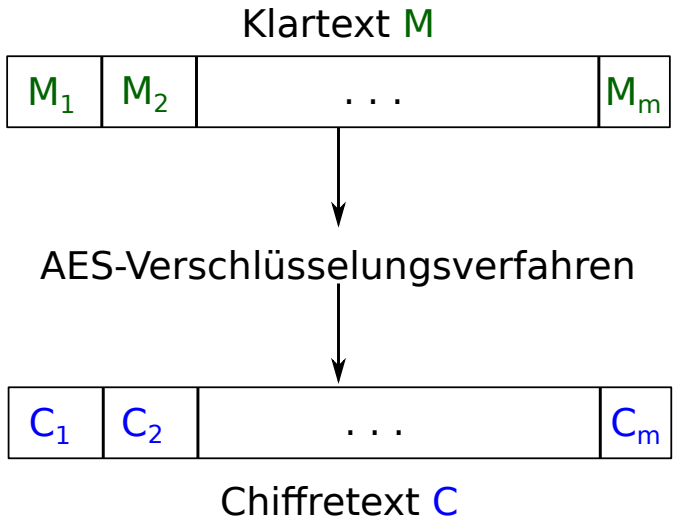
Advanced Encryption Standard (AES)



AES ist eine sichere 128 bit Blockchiffre

Schlüssellängen (in Bits): 128, 192, 256

Generelle AES-Verschlüsselungsstrategie

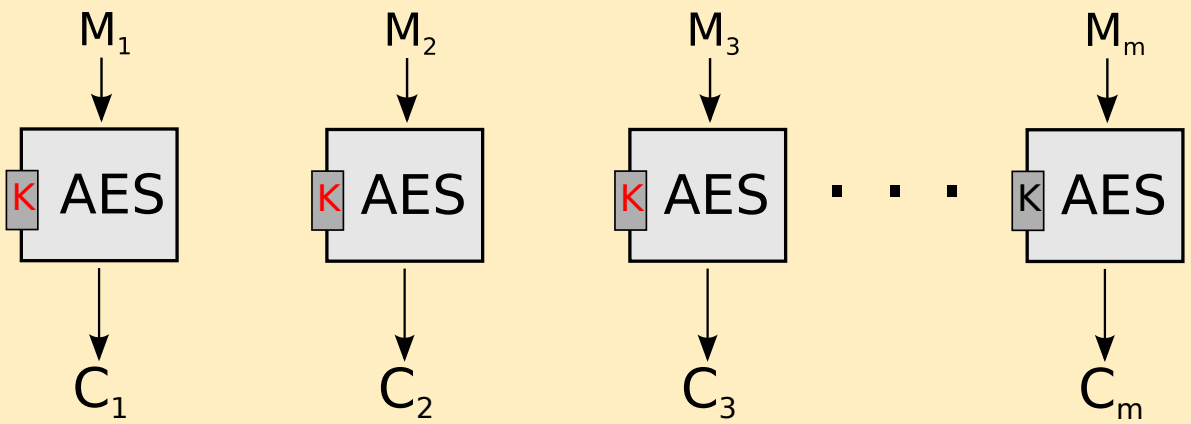


Generelle Verschlüsselungsstrategie

- Aufteilung der Nachricht M in 128-bit Blöcke.
- Verschlüsselung der einzelnen Blöcke mittels AES.
- Ergebnis: Chiffretext $C = C_1, C_2, \ldots, C_m$.

Naive AES-Verschlüsselungsstrategie

Electronic Code Book (ECB)

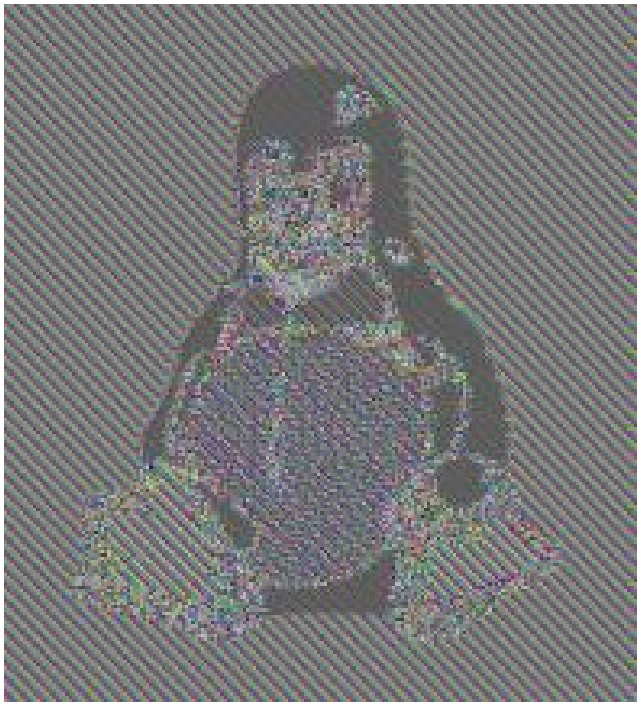


Problem

- Gleiche Klartextblöcke $\Longrightarrow$ gleiche Chiffretextblöcke.
- $\Longrightarrow$ Struktur bleibt durch Verschlüsselung erhalten.
- $\Longrightarrow$ Unsicheres Verschlüsselungsverfahren

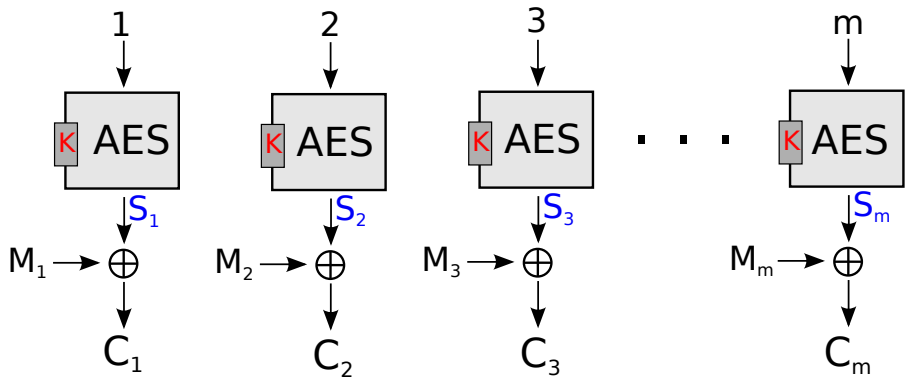
Wikipedia-Beispiel: ECB-Verschlüsselung

Klartext



Chiffretext

Einfacher Counter-Mode

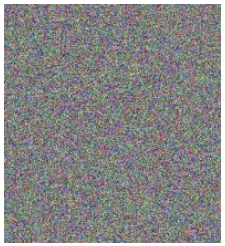


Problem

- Der einfache Counter-Mode erzeugt immer den gleichen Schlüsselstrom.
- Unter einem Schlüssel kann nur eine Nachricht verschlüsselt werden.
- Unsicher bei Mehrfachnutzung eines Schlüssels.

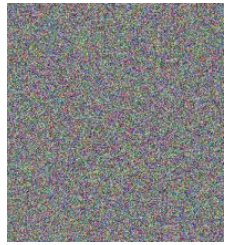
Beispiel: Wiederverwendung eines Schlüsselstroms

M



$C = M \oplus S$

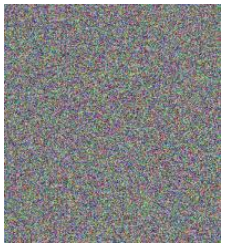
M'



$C' = M' \oplus S$

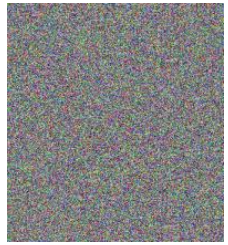
Beispiel: Wiederverwendung eines Schlüsselstroms

M



$C = M \oplus S$

M'

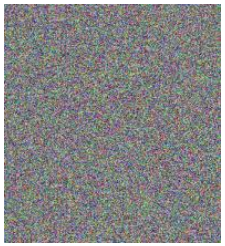


$C' = M' \oplus S$

$$C \oplus C' = (M \oplus S) \oplus (M' \oplus S) = M \oplus M'$$

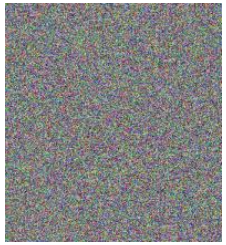
Beispiel: Wiederverwendung eines Schlüsselstroms

M

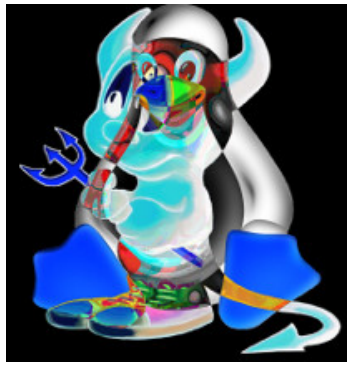


$C = M \oplus S$

M'

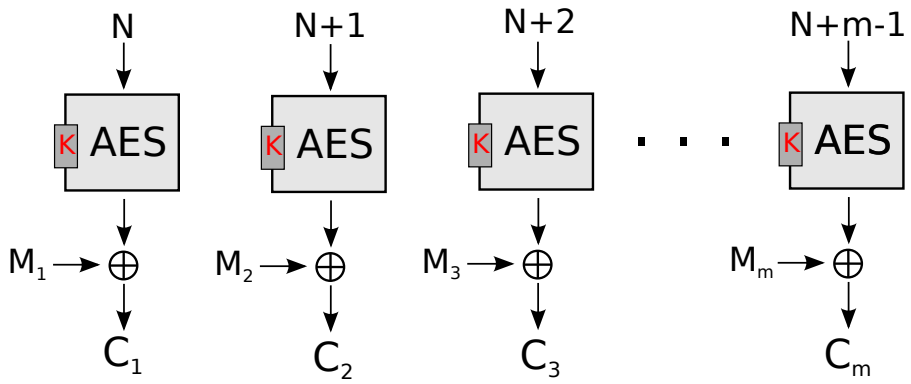


$C' = M' \oplus S$



$$C \oplus C' = (M \oplus S) \oplus (M' \oplus S) = M \oplus M'$$

Counter-Mode



- Der Zustand N (Nonce) ermöglicht es viele Nachrichten sicher unter einem Schlüssel zu verschlüsseln.
- Gute Nonce basierte Verschlüseselungsverfahren sind sicher, so fern sich kein (Nonce, Schlüssel)-Paar wiederholt.

Nachteile von Verschlüsselungsverfahren

Nonce-basierte Verschlüsselungsverfahren

- Ctr-Mode
- CBC-Mode
- CFB-Mode
- OFB-Mode
- ...

Nachteile

1. Integrität des Klartextes wird nicht geschützt
2. Keine Sicherheit bei Nonce-Wiederholung.
   (siehe Einfacher Counter-Mode)

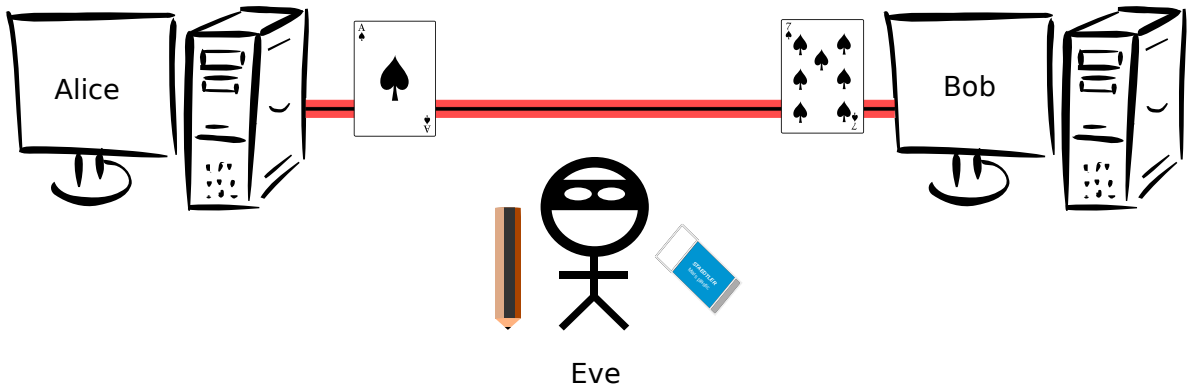
$$\Longrightarrow \text{ Eingeschränkte Praxistauglichkeit}$$

Authentisierte Verschlüsselung (AE)

Verfahren zur Authenthisierte Verschlüsselung

- Schützen die Vertraulichkeit der verschlüsselten Nachricht.
- Schützen die Integrität der verschlüsselten Nachricht.

Authentisierte Verschlüsselung (AE)

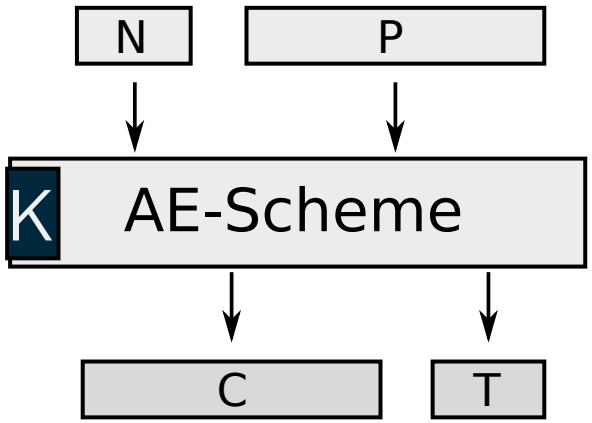


Verfahren zur Authenthisierte Verschlüsselung

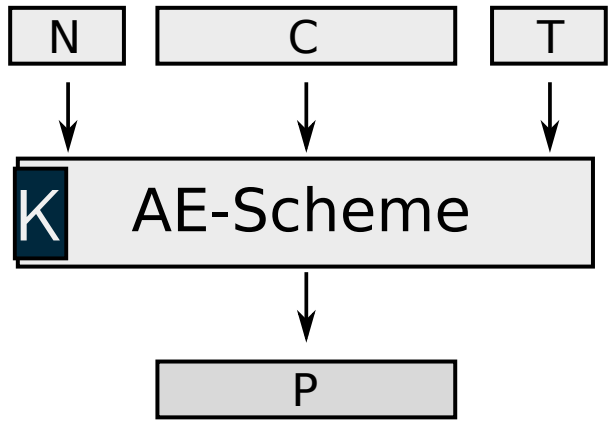
- Schützen die Vertraulichkeit der verschlüsselten Nachricht.
- Schützen die Integrität der verschlüsselten Nachricht.

Verfahren zur Authentisierten Verschlüsselung

Encryption



Decryption



N: Nonce

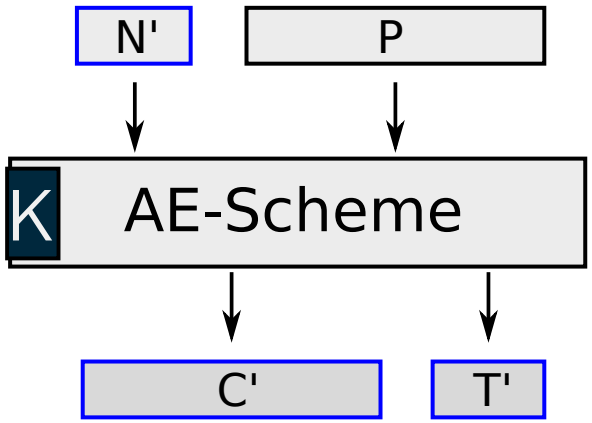
P: Nachricht/Klartext

C: Chiffretext

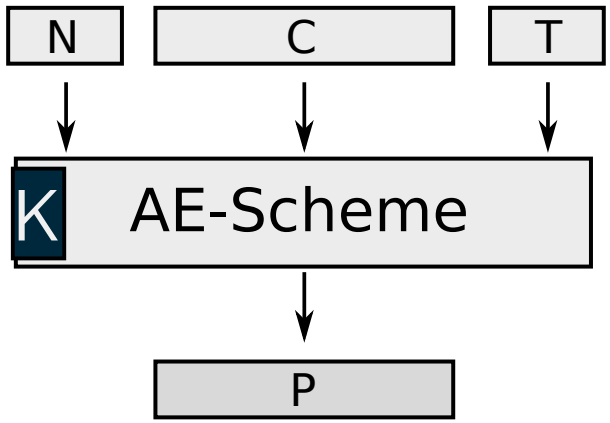
T: kryptographische Prüfsumme

Verfahren zur Authentisierten Verschlüsselung

Encryption



Decryption



N: Nonce

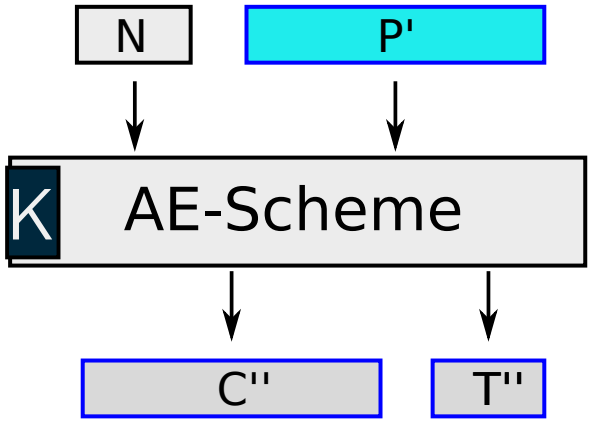
P: Nachricht/Klartext

C: Chiffretext

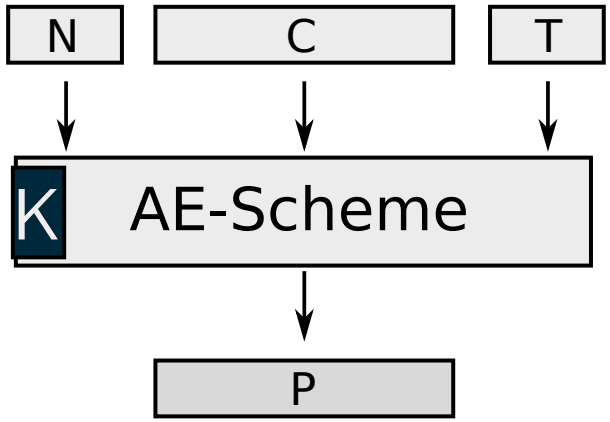
T: kryptographische Prüfsumme

Verfahren zur Authentisierten Verschlüsselung

Encryption



Decryption



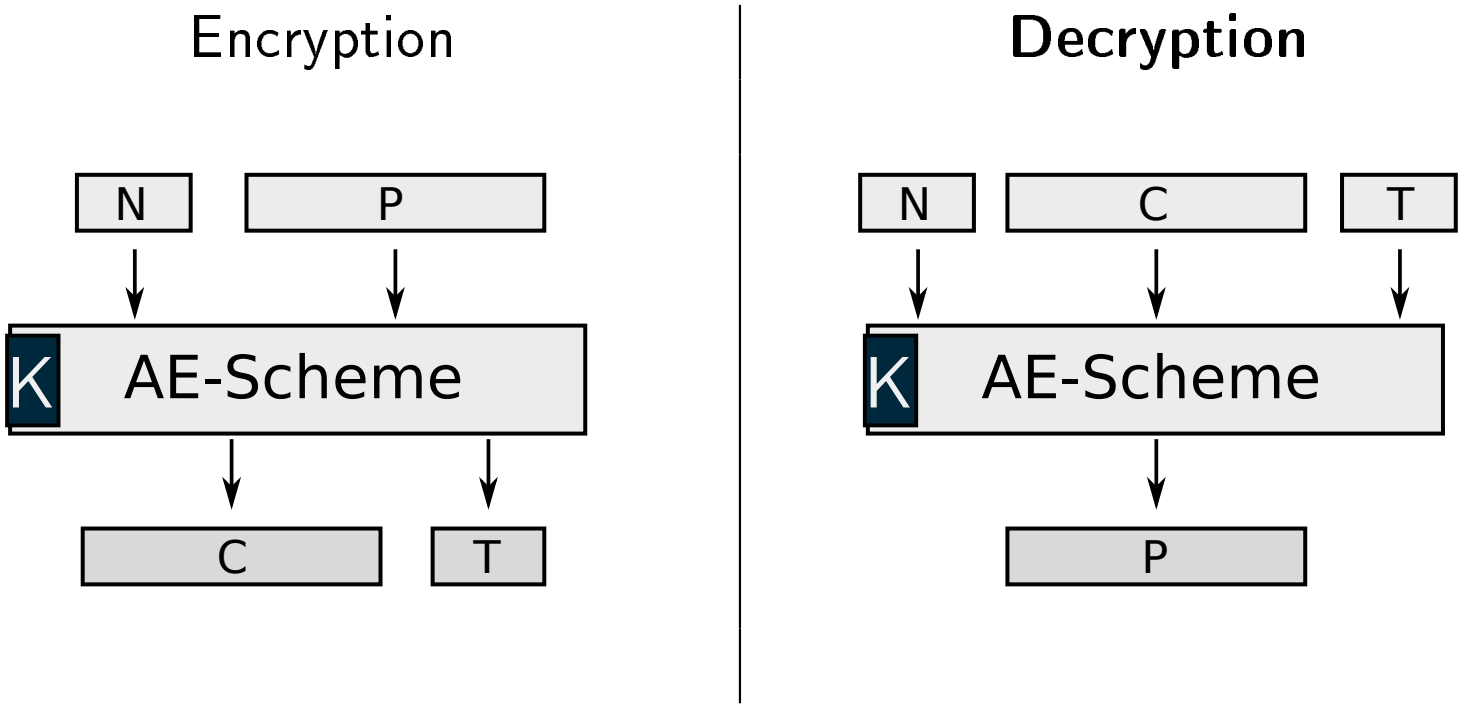
N: Nonce

P: Nachricht/Klartext

C: Chiffretext

T: kryptographische Prüfsumme

Verfahren zur Authentisierten Verschlüsselung



N: Nonce

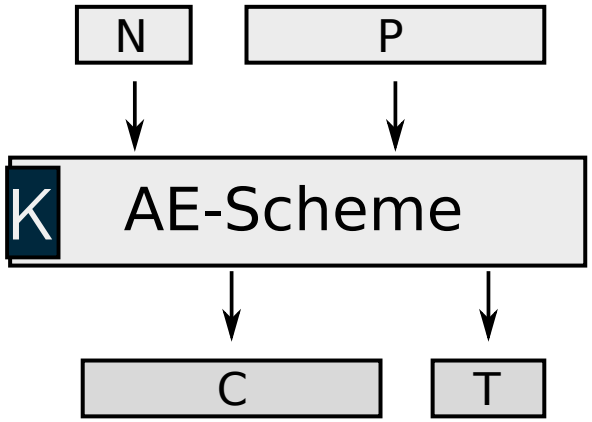
P: Nachricht/Klartext

C: Chiffretext

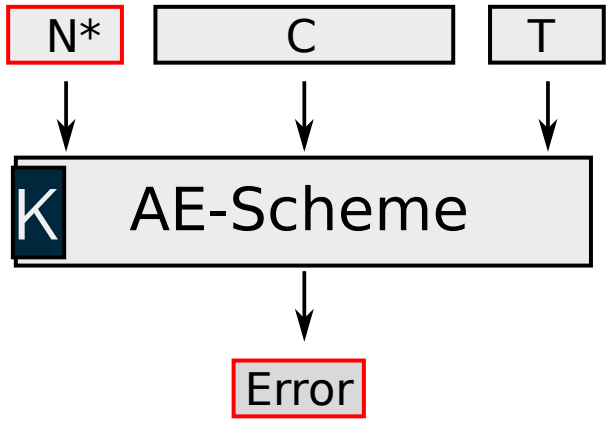
T: kryptographische Prüfsumme

Verfahren zur Authentisierten Verschlüsselung

Encryption



Decryption



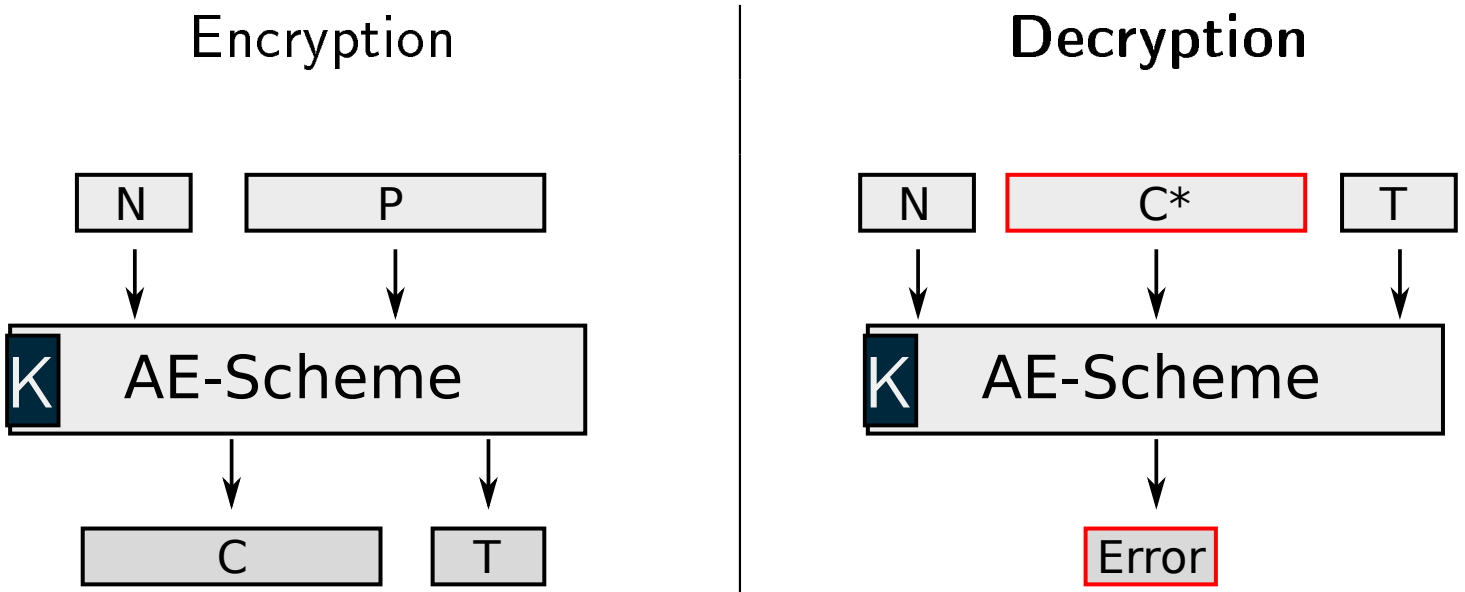
N: Nonce

P: Nachricht/Klartext

C: Chiffretext

T: kryptographische Prüfsumme

Verfahren zur Authentisierten Verschlüsselung



N: Nonce

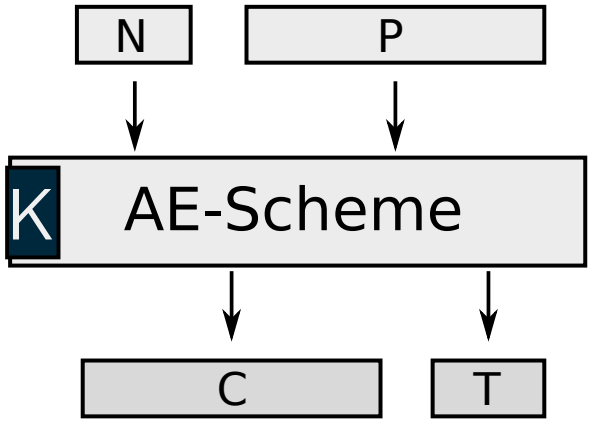
P: Nachricht/Klartext

C: Chiffretext

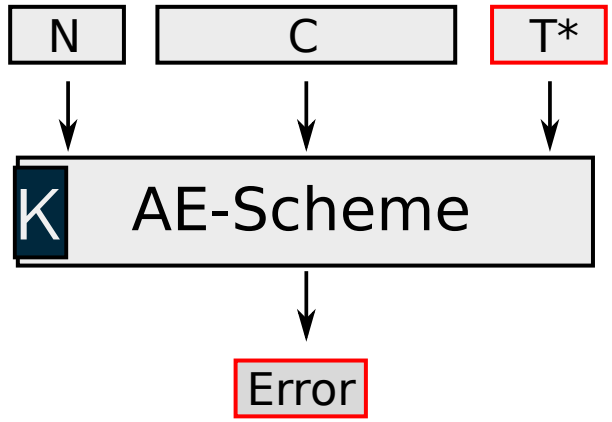
T: kryptographische Prüfsumme

Verfahren zur Authentisierten Verschlüsselung

Encryption



Decryption



N: Nonce

P: Nachricht/Klartext

C: Chiffretext

T: kryptographische Prüfsumme

Nachteil von AE-Scheme

- AE-Schemes
  - GCM (McGrew und Viega; RFC 5116)
  - OCB (Krovetz und Rogaway; RFC 7253)
  - ...

- **Problem**
  Keine Sicherheit bei Nonce-Wiederholung

Keine Sicherheit bei Nonce-Wiederholung

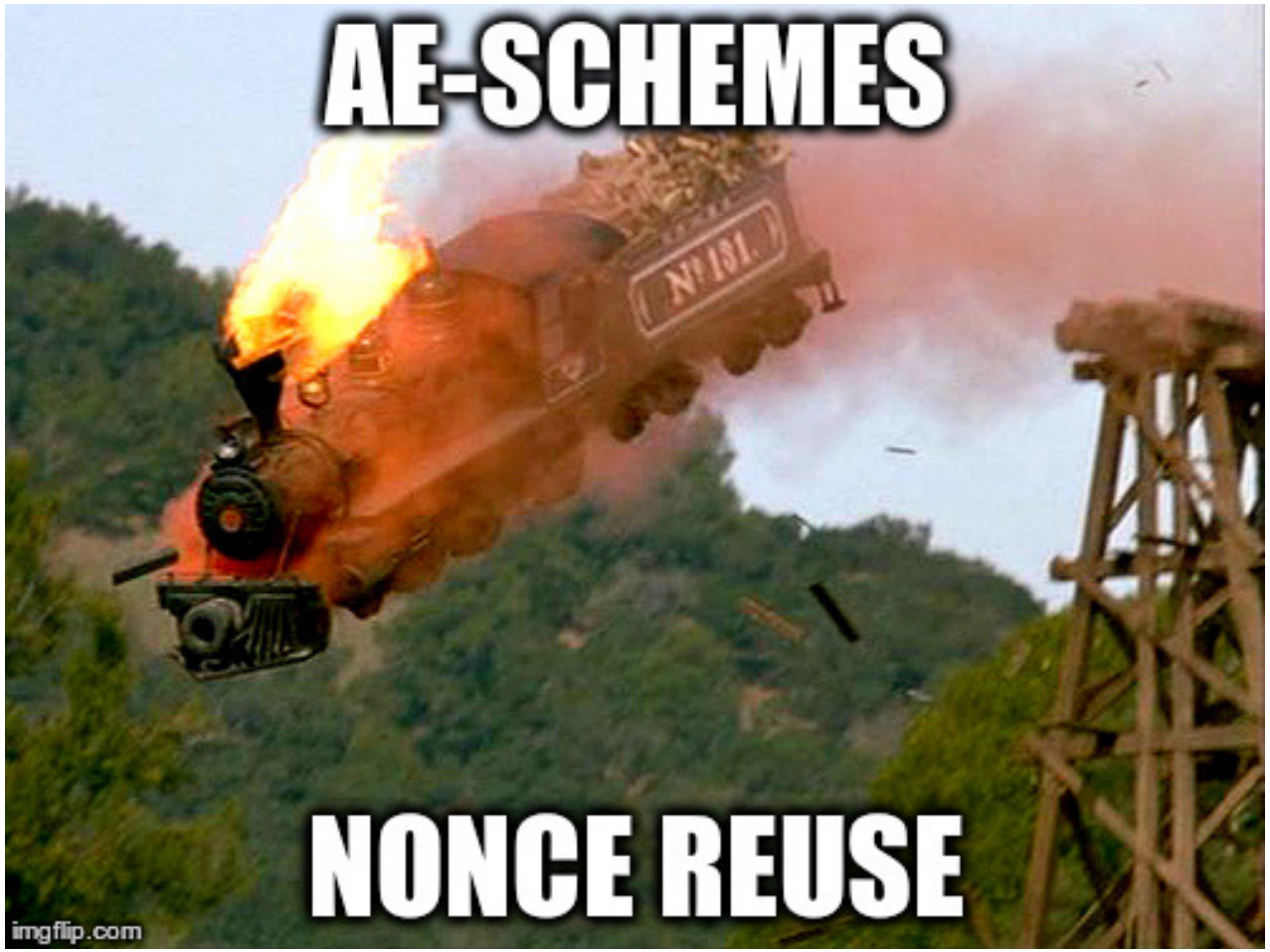
Scheme	**Nonce Misuse**	
	privacy	integrity
CCFB	**kaputt**	**kaputt**
CHM	**kaputt**	**kaputt**
CWC	**kaputt**	**kaputt**
EAX	**kaputt**	**kaputt**
GCM	**kaputt**	**kaputt**
IACBC	**kaputt**	**kaputt**
IAPM	**kaputt**	**kaputt**
OCB	**kaputt**	**kaputt**
RPC	**kaputt**	**kaputt**
TAE	**kaputt**	**kaputt**
XCBC-XOR	**beschädigt**	**kaputt**

---

kaputt: Angriff mit Komplexität: $O(1)$.

beschädigt: Angriff mit Komplexität $O(2^{n/4})$.

Zusammenfassung



GCM ist extrem fragile

- Nonce-Kollisionen: $|N| \neq 96$-bit
- Kollisionsschwäche beim Kürzen der Prüfsumme [Ferguson 2005]
- Nonce-Wiederholung ermöglicht Key Recovery [Joux 2006, Mattsson und Westerlund 2015]
- Schwache Schlüssel [Handschuh und Prennel 2008, Saarinen 2015]

GCM ist extrem fragile

"Based on these weaknesses, our recommendations are: Do not use GCM . . . If other considerations dictate the use of GCM, use it only with a 128-bit tag." – Niels Ferguson

GCM ist extrem fragile

"The ICV consists solely of the AES-GCM Authentication Tag. Implementations MUST support a full-length 16-octet ICV, and MAY support 8 or 12 octet ICVs, and MUST NOT support other ICV lengths." – RFC 4106 (IPSec)

Wiederverwendung einer Nonce (Nonce-Reuse)

Ist dieses Problem überhaupt praxisrelevant?

Wiederverwendung einer Nonce (Nonce-Reuse)

Ist dieses Problem überhaupt praxisrelevant?

- Fehlerhafte Implementation/Design
  - *The Misuse of RC4 in Microsoft Word and Excel* [Wu 2004]
  - *Attacking and Repairing the WinZip Verschlüsselung Scheme* [Kohno 2005]
  - *KRACK: Breaking WPA2 by forcing nonce reuse* [Vanhoef 2017]
  - ...

```
int getRandomNumber()
{
    return 4;  // chosen by fair dice roll.
               // guaranteed to be random.
}
```

Bedienfehler

- Wiederherstellung eines Backups
- Klonen von virtuellen Maschinen
- Umgebungen mit niedriger Entropie (Server, Embedded Systems, . . .)

Misuse Resistant AE-Scheme (MRAE-Schemes)

- Schützen Integrität und Vertraulichkeit, auch bei Nonce-Wiederholung
- Unterstützten beliebig lange Nonces (Initialization Vector)
- SIV [Rogaway und Shrimpton 2006]; RFC 5297
- RIV [Abed et al. 2016]
- Verfahrensweise

Verschlüsselung(N,M)

- $T = \mathrm{MAC}_{K_1}(N, M)$
- $C = \mathrm{Counter}_{K_2}(T, M)$
- $T' = \mathrm{MAC}_{K_1}(N, C) \oplus T$

Entschlüsselung(N,C,T/T')

- $T = \mathrm{MAC}_{K_1}(N, C) \oplus T'$
- $M = \mathrm{Counter}_{K_2}(T, C)$
- $\mathrm{MAC}_{K_1}(N, M) == T$? OK : Error

- Nachteil: Klartext muss zwei mal gelesen werden

$\Longrightarrow$ Benutzt MRAE-Schemes falls möglich

Alternativen zu MRAE

Mein Szenario erlaubt es nicht die Klartext zweimal zu lesen.

Was soll ich tun?

Alternativen zu MRAE

Mein Szenario erlaubt es nicht die Klartext zweimal zu lesen.
Was soll ich tun?

Quelle: `https://pixabay.com/`, Author: Dmitry Abramov

Alternativen zu MRAE

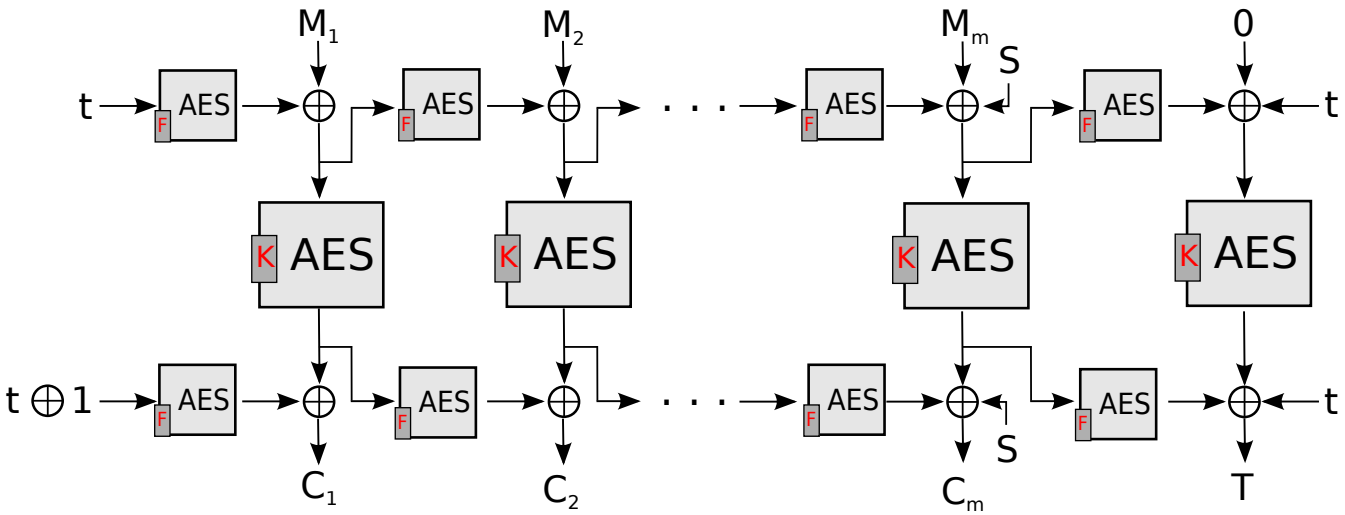
Mein Szenario erlaubt es nicht die Klartext zweimal zu lesen.
Was soll ich tun?

Robuste AE-Schemes

Robuste AE-Schemes: schützen bei einer Nonce-Wiederholung die Integrität und ggf. auch die Vertraulichkeit

- McOE-G [Fleischmann, Forler und Lucks 2012]
- POET (CAESAR 2 Runde)) [Abed et al. 2014]
- COLM (CAESAR 3 Runde) [Andreeva et al. 2016]
  - AES-COPA [Andreeva et al. 2013]
  - ELmD [Datta und Nandi 2014]

POET



$$t = \mathrm{PMAC}^*_{K,L}(N) \qquad S = \mathrm{AES}_K(|M|)$$

$$Adv(q,\ell,t) \leq \frac{5.5(\ell+2q)^2}{2^{128}} + (\ell+q)^2\epsilon + 2\max\left\{q(\ell+q)\epsilon, \frac{q(\ell+q)}{2^{128}-q}\right\}$$

$$+ \frac{(\ell+4q)^2}{2^{128}-(\ell+4q)} + \mathbf{Adv}^{SPRP}_{AES}(\ell+4q, \mathrm{O}(t)).$$

Was tun?

- WIEDERHOLT NIEMALS EINE NONCE!
- Mit welchem Verfahren soll ich meine Daten verschlüsseln
  1. Misuse Resistant AE-Schemes (SIV, RIV ...)
  2. Robuste AE-Schemes (COLM, McOE-G, POET, ...)
  3. Reguläre AE-Schemes (GCM, OCB, ...)
  4. Verschlüsselungsverfahren (CBC, Counter, OFB, ...)
- Redet mit Kryptographen

Was mit Blockchain

Virtueller Kooperationspartner: Forschungsinstitut für Blockchain und Cyberkryptofragen

Proof of Work/Space: Password Hashing

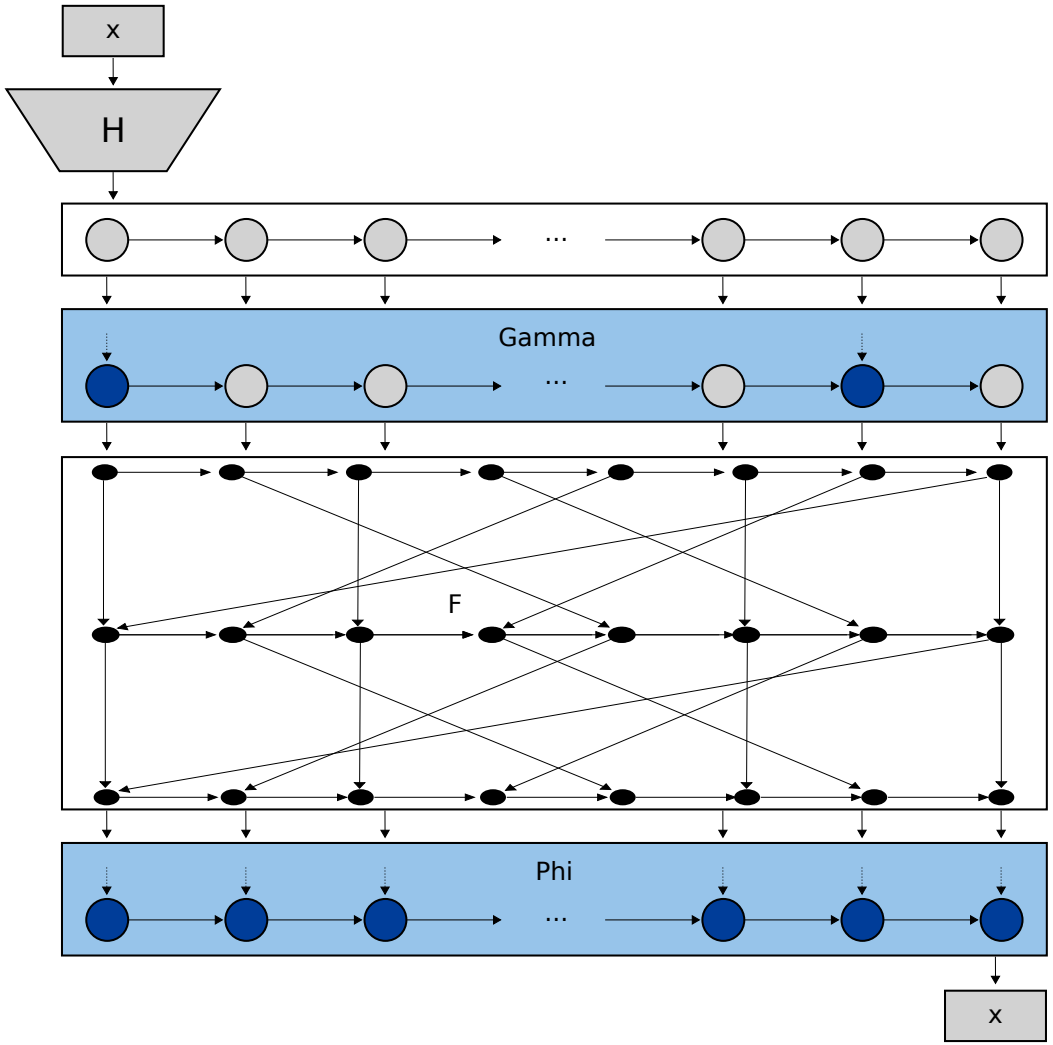
Lösung

Speicherintensive Hashfunktionen, welche schlecht auf GPU, FPGA und ASICS performen.

Beispiele

- scrypt [Colin 2009]
- Argon2d (PHC Gewinner) [Biryukov, Dinu, Khovratovich 2017]
- Catena-Framework [Forler Lucks, Wenzel 2015]
- Catena-Stonefly [Lucks, Wenzel 2015]

Catena-Stonefly Rundenfunktion



Sozial nüzliches Mining

Prove of Useful Work

- Storage
- Network Services

Tuwat: Kryptomagie = Mathematik + Freie Software

Kryptomagie = Mathematik + Freie Software

- Kryptographie ermöglicht durch Mathematik
  - auf einer kleinerfingernagelgroßen Fläche oder
  - mit einer handvoll Programmzeilen,

Daten sicher selbst gegen eine weltweite Geheimdienstzusammenarbeit zu verschlüsseln.

- Freie Software
  - verhindert Hintertüren und
  - ermöglicht das Finden von Fehlern.

Verteilte Schlüsselerzeugung und Schwellenwertkryptographie

- Desmedt (1987)

Verteilte Schlüsselerzeugung und Schwellenwertkryptographie für OpenPGP

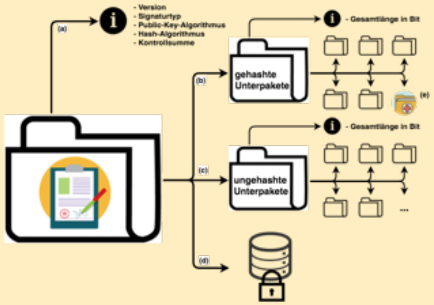
- Heiko Stamer
  - `https://savannah.nongnu.org/projects/libtmcg`
- CCB Datengarten/81 (2017)
  - `http://www.nongnu.org/libtmcg/dg81_slides.pdf`

Blind Signatures

- Chaum (1983)

Blind Signature über ECC für OpenPGP

- Rüdiger Weis, Bruno Kirschner, OpenTech Summit 2016,
  - Authenticated Anonymity by Math



0ndorio

- Some source code:
  `https://github.com/0ndorio/verify_me`

Tuwat: Defense Against the Dark Arts

Edward Snowden, Guardian, 11. März 2014

- "Crypto works.
  - It's not an arcane black art.
  - It is a basic protection,
  - the Defense Against the Dark Arts for the digital world.
- We must implement it.
- actively research it."