Kryptographische Routinen als elektronischer Fingerabdruck

Andreas Schuster

Erleben, was verbindet.

Zielsetzung

- Mengenproblem: Klassifikation, um Gruppen identischer/ähnlicher Objekte zu bilden
- Zusammenhänge erkennen:
  Gibt es eine Verbindung zwischen unterschiedlichen Objekten?
- Täterprofil:
  - Wer ist der Urheber?
  - Wie ist sein Kenntnisstand?
  - Welche Quellen nutzt er?
- Voraussetzung: Hinreichend große Anzahl von Ausprägungen des untersuchten Merkmals

Unterscheidungskriterien aus dem Bereich der Linguistik

- Orthographische Auffälligkeiten
  - Buchstabendreher: bind cmd frist! (statt: first)
  - Fehlender Buchstabe: WritePip Error! (statt: WritePipe)
  - Dto., eventuell auch Lautmalerei: Dcryption Error!

- Grammatikalische Auffälligkeiten
  - Writed by UglyGorilla, 06/29/2007

- Szenesprache:
  - `d0wnl0ad`

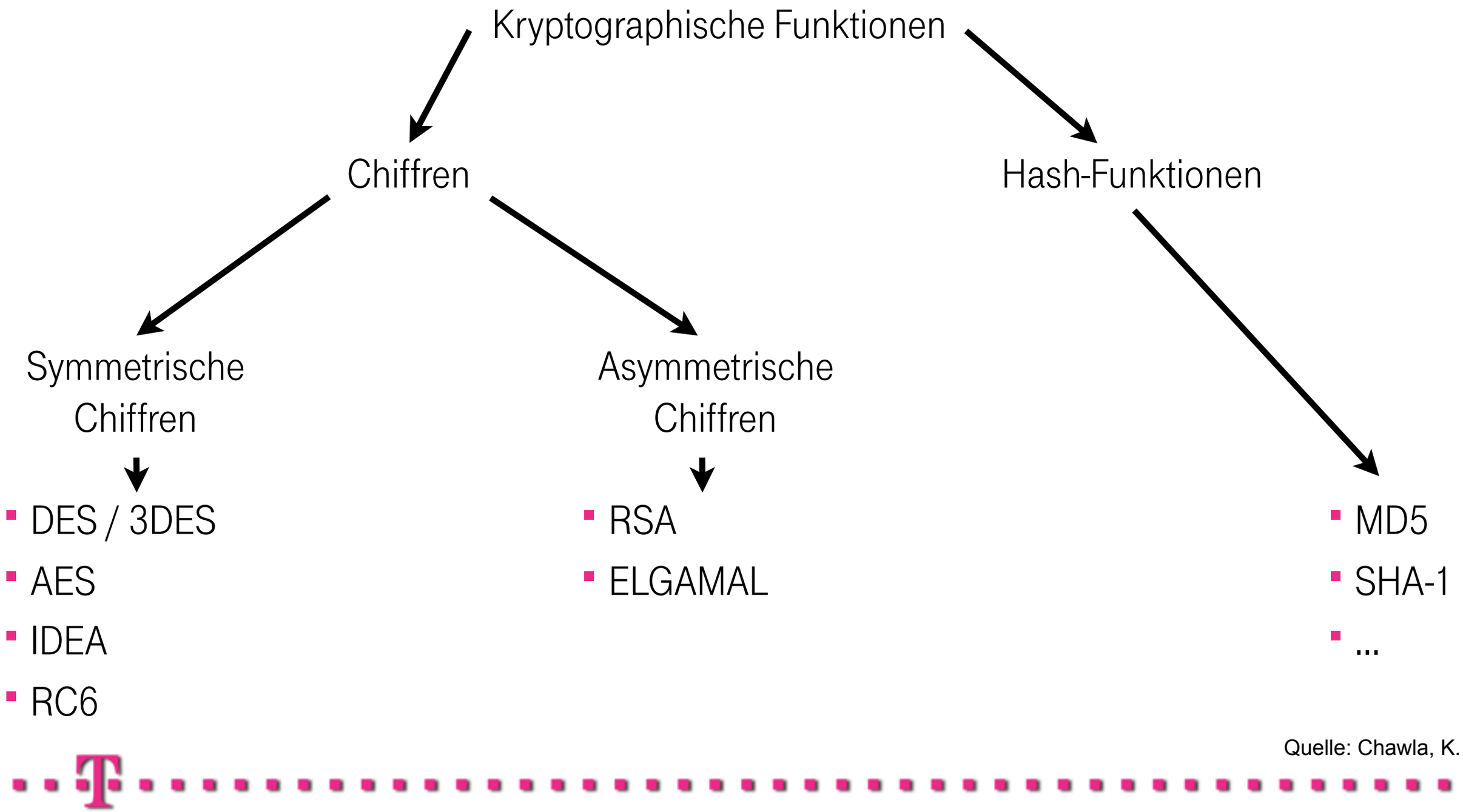
- Beispiele entnommen aus:
  Mandiant. APT1: Exposing One of China‘s Cyber Espionage Units. (2013)

Unterscheidungskriterien aus dem Bereich der Informatik

- Algorithmus
  - Ist das Verfahren angemessen?
  - Wie bekannt ist das Verfahren?

- Implementierung
  - Ist die Implementierung korrekt?
  - Standardbibliothek oder eigene Implementierung?
  - Wahl der Programmiersprache/Entwicklungsumgebung

Systematik kryptographischer Funktionen



Quelle: Chawla, K.

Asymmetrische Verschlüsselung

- Öffentlicher und privater Schlüssel
- Sicherheit basiert auf Komplexität mathematischer Probleme (z.B. Faktorisierung großer Primzahlen)
- Etwa 3 Größenordnungen langsamer als symmetrische Verfahren
- Typische Anwendung bei Schadsoftware:
  - Schadsoftware erzeugt Sitzungsschlüssel für symmetrisches Verfahren
  - Übermittlung des Schlüssels an Command&Control Server mittels asymmetrischem Verfahren
    - Schadsoftware enthält öffentlichen Schlüssel
    - Command&Control Server enthält privaten Schlüssel

Symmetrische Verschlüsselung

- Beide Kommunikationspartner verwenden den gleichen Schlüssel
- Sicherheit basiert wesentlich auf der Vertraulichkeit des Schlüssels
- Relativ schnelle Verfahren

- In Schadsoftware:
  - Schlüssel sind häufig fest codiert
  - Algorithmen sind häufig sehr einfach (XOR), mitunter Eigenentwicklungen

Beispiel zur Einschätzung der Kompetenz eines Angreifers

- Verschlüsselung mit XOR:
  - populär
  - symmetrische Chiffre
  - $c_i = p_i \text{ ^ } k$; $p_i = c_i \text{ ^ } k$
  - $p_i = p_i \text{ ^ } k \text{ ^ } k$

a	b	a ^ b
0	0	0
0	1	1
1	0	1
1	1	0

- Gefahr der Kompromittierung des Schlüssels, da 0 ^ k = k

- „Verbessertes Verfahren“: XOR mit mehreren Runden
  - $c_i = p_i \text{ ^ } a \text{ ^ } b \text{ ^ } b \text{ ^ } c$
  - $c_i = p_i \text{ ^ } (a \text{ ^ } c)$
  - $c_i = p_i \text{ ^ } k$, im untersuchten Fall wurden Groß- durch Kleinbuchstaben ersetzt, u.U.

Beispiel für Auffälligkeiten in Implementierungen

- Tiny Encryption Algorithm (TEA)
  - Wheeler, D. and Needham, R.: TEA, a Tiny Encryption Algorithm. (1995)
  - Statische Analyse: Suche nach Konstante 0x9e3779b9

Encode Routine

Routine, written in the C language, for encoding with key k[0] - k[3]. Data in v[0] and v[1].

```
void code(long* v, long* k)  {
unsigned long y=v[0],z=v[1], sum=0,   /* set up */
 delta=0x9e3779b9, n=32 ;             /* a key schedule constant */
while (n-->0) {                       /* basic cycle start */
  sum += delta ;
    y += (z<<4)+k[0] ^ z+sum ^ (z>>5)+k[1] ;
    z += (y<<4)+k[2] ^ y+sum ^ (y>>5)+k[3] ;   /* end cycle */
               }
v[0]=y ; v[1]=z ; }
```

Beispiel für Auffälligkeiten in Implementierungen

- Joan Calvet: Cryptographic Function Identification in Obfuscated Binary Programs. (2012)

- Calvet findet Konstante in Code des „Storm Worms“, jedoch abweichendes Verhalten bei dynamischer Analyse
  - TEA: z+= ((y<<4) + k[2]) ^ (**y**+**sum**) ^ ((y>>5) + k[3])
  - Storm Worm: z+= (y<<4) + (**y** ^ k[2]) + (**sum** ^ (y>>5)) + k[3]
  - Identische Form in Silent Banker!

- Russischer Wikipedia-Artikel verlinkt auf Website mit falscher Implementierung!

Beispiel aus einem realen Fall

Kryptographische Routine in einem Trojanischen Pferd

- Schadcode
  - Verwendung im Rahmen gezielter Angriffe zur Industriespionage
  - erstes gefundenes Sample wurde April 2012 erstellt

- Viginère Chiffre
  - symmetrisch
  - polyalphabetische Substitution

- Implementierung:
  - frei von Fehlern
  - Schlüssel „thequickbrownfxjmpsvalzydg“

Quelle: Cryptomuseum.org

Signaturerstellung und Suche nach entsprechender Schadsoftware

virustotal intelligence

Rulesets | Notifications

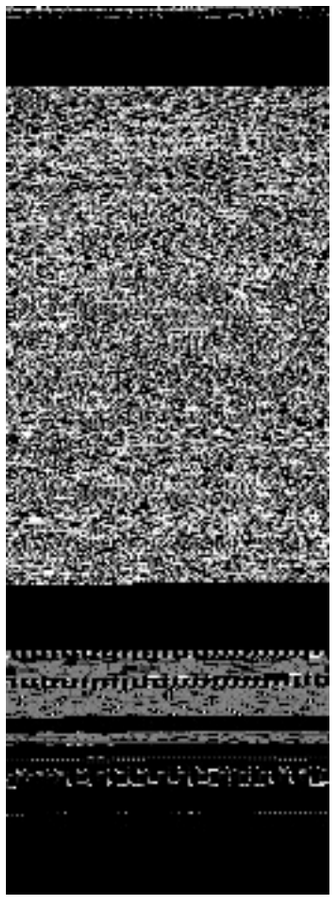
Foxy

Enabled | Disabled | Delete

```
rule malware_brownfox_encryption: malware encryption brownfox mswindows
{
    meta:
        domain = "malware"
        os = "mswindows"
        description = "Foxy RAT implementation of Viginere cipher"
        author = "Andreas Schuster"

    strings:
        // there are different alphabets in use
        $alpha1 = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789@#=" ascii fullword
        $alpha2 = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789-_=" ascii fullword
        $alpha3 = { 60 31 32 33 34 35 36 37 38 39 30 2d 3d 7e 21 40 23 24 5e
            26 2a 28 29 5f 2b 71 77 65 72 74 79 75 69 6f 70 5b 5d 51 57 45 52
            54 59 55 49 4f 50 7c 61 73 64 66 67 68 6a 6b 6c 3b 27 41 53 44 46
            47 48 4a 4b 4c 3a 7a 78 63 76 62 6e 6d 2c 2e 2f 5a 58 43 56 42 4e
            4d 3c 3e 3f 00 }
        $alpha4 = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789=" ascii fullword
        $alpha5 = "1234567890qwertyuiopQWERTYUIOPasdfghjklASDFGHJKLzxcvbnmZXCVBNM=@#&*+:/." ascii fullword

        // but all use the same Viginere key
        $key = "thequickbrownfxjmpsvalzydg"

        // encrypted "http"
        $encrypted_http = { 47 00 6f 00 76 00 64 00 4a 00 }

    condition:
        (any of ($alpha*) and $key)
        or $encrypted_http
}
```

Get Help!

Die Suche führt zu sehr unterschiedlichen Arten von Schadsoftware

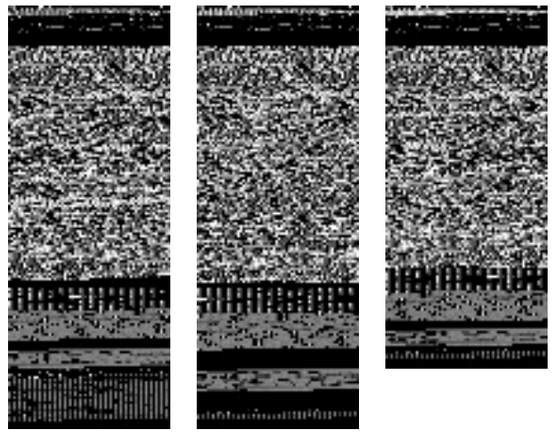


Downbot,
April 2008

Tapslix.A,
ShadyRAT

Danginex

Comsirig



Govdi.A

RPC-Server für
Seitwärtsbewegung

Visualisierung nach:
L. Nataraj, S. Karthikeyan, G. Jacob, and B. S. Manjunath. 2011. Malware images: visualization and automatic classification.
In Proceedings of the 8th International Symposium on Visualization for Cyber Security (VizSec '11).

Die Programme lassen sich in vier Kategorien einteilen

- Backdoor 1
  - Steuerung über steganographischen und verschlüsselten Kanal
  - durch Dropper installiert (Spearphishing) oder
    vom Angreifer eingebracht (Softwareaktualisierung, Seitwärtsbewegung)

- Backdoor 2
  - ausschließlich durch den Angreifer eingebracht
  - C2 Server ist zum Zeitpunkt der Installation nicht (erkennbar) aktiv

- Seitwärtsbewegung: keine Schadsoftware im engeren Sinne, bestenfalls „unerwünscht“

- Steuerung: Encoder für Kommandos

Zusammenfassung

- Bildung einer Signatur aus der initial vorgefundenen Viginére Chiffre und Schlüssel

- Suche in großen Korpora (VirusTotal,VirusShare)
  - Ergebnis derzeit ca. 100 Dateien
  - mehrere Arten von (Schad-) Software
  - „gutartige“ Software, keine Erkennung durch AV-Produkte, aber klarer Fallbezug

- In dem vorliegenden Fall erwiesen sich kryptographische Routinen als trennscharfes Merkmal.

- Einschränkungen:
  - bei Verwendung von Standardverfahren und -bibliotheken
  - bei sorgfältiger Planung und Implementierung von Schlüsselerzeugung und -austausch

Vielen Dank für Ihre Aufmerksamkeit!

Andreas Schuster
andreas.schuster@telekom.de

Erleben, was verbindet.