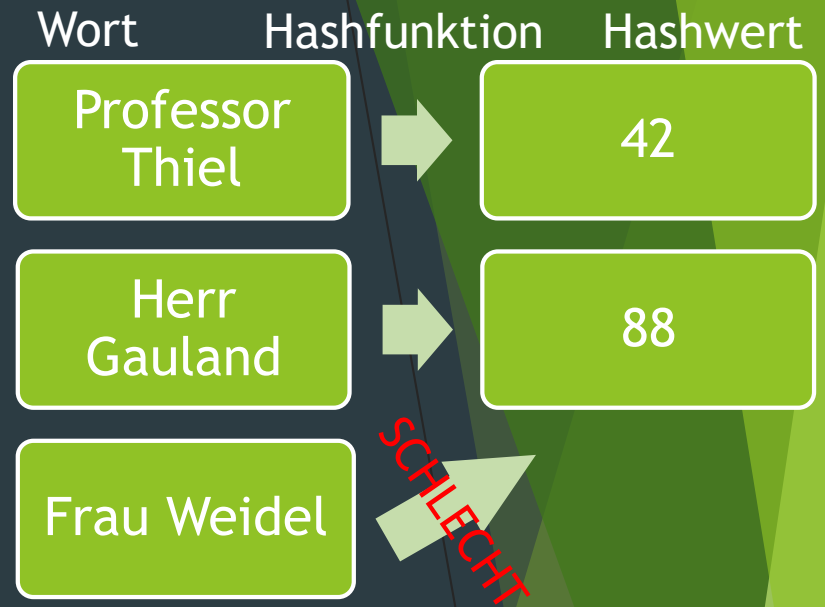
Grundlegende Funktionsweisen dieses und ähnlicher Tools

Themenübersicht

- Hashfunktionen und kryptographische Hashfunktionen
- Wie Unix Passwörter speichert
  - und welche Hashverfahren bei Unix üblich sind
- Wer war John the Ripper
  - Und wie funktioniert er
- Die Modi und was man gegen Sie tuen kann
  - Single Mode
  - Incremental Mode
  - Wordlist Mode
- Distributed John und MPI
- Ausblick auf ähnliche Programme

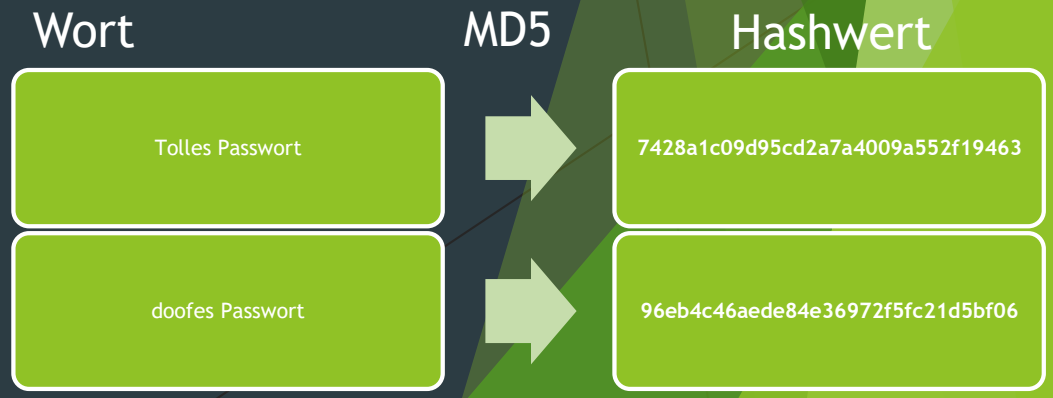
Hashfunktionen

- Weisen einem Wort mittels einer Funktion einen Hashwert zu
  - Selbes Wort = selber Hashwert
- Einwegfunktion
  - Aus dem Hashwert lässt sich der Ursprungstext nicht erzeugen
- Hashwerte haben eine feste Länge (hier 2 Zeichen)



Kryptographische Hashfunktionen

- Werden bewertet nach
  - Kollisionssicherheit
    - Unterschiedliche Wörter sollen nicht den selben Hashwert ergeben
  - Geschwindigkeit
    - Wie schnell ein Hashwert aus einem Wort errechnet wird



Wie Unix Passwörter speichert

- Früher /etc/passwd
  - Für alle Programme lesbares speichern aller Nutzerdaten auch der Passwörter als Hashwerte
    - Aber halt für jeden lesbar
- Heute /etc/shadow
  - Nutzerdaten immernoch in /etc/passwd
  - Passwörter gezielt in einer Datei als Hashwerte gespeichert
    - Aber nicht jeder hat Leserechte
  - Ebenfalls gespeichert welche Hashfunktion verwendet wurde
  - Beispiel
    - sschuck:6Finhqm.v$kgiVkr38bZXIUNLFJZQYa..mXmCmVYn7ohD/OkYgB568jVyhNm2ZQUrknk14kXZdMnAJsDynHQbgz7elqOQlV0:17670:0:99999:7:::
      - keine Sorge ist ein unsicheres und kein sicheres Passwort
      - keine Sorge „knacken“ wir nachher auch noch

Hashfunktionen in der /etc/shadow

- 1 = MD5 (Message-Digest Algorithm 5)
  - 1991 entwickelt
    - Schon 1994 bereits Beweis von Kollisionen
    - 2012 letzter großer Angriff auf MD5 mittels einer Chosen-Prefix-Kollision
  - 128 Bit lange Hashwerte, als Hexadezimalzahl gespeichert
- $2a$ oder $2y$ oder $2b$ = bcrypt bzw. Blowfish
  - „a“ steht für 10 Iterationen des ursprünglichen Algortihmus
    - 1999, erste Version des Algortihmus
  - „y“ steht für den mit einem „verbesserten“ Algorithmus erstellte Hashwerte
    - 2011, nachdem ein Bug entdeckt und behoben wurde
  - „b“ steht für den mit der neuere Version des Algorithmus erstellten Hashwert
    - 2014, nachdem ein anderer Bug entdeckt wurde
- 5 = SHA-256 (Secure Hash Algorithm)
  - Gehört zur zweiten Generation des SHA Hashfunktionsfamilie
    - Wurde 2004 entwickelt, nachdem bekannt wurde das SHA 1 diverse Schwächen aufweist
  - 256 Bit lange Hashwerte
- 6 = SHA-512
  - 512 Bit lange Hashwerte, ebenfalls Teil der 2. Generation

Wie Windows Passwörter speichert

- Was ist ein „SAM“ File?
- „SAM“ File encrypted mit NTLM hash (LM-Hash)
  - Passwortlänge <= 15 Zeichen
  - %SystemRoot%\system32\config\SAM
- „SAM“ File decrypted mit SAM Lock Tool (Syskey)
  - %SystemRoot%\system32\config\system
- Boot „SAM“ und „system“ gesperrt

Ergänzung:

Passwörter werden in SAM Files gespeichert
NTLM Hash (LM-Hash) auch Lan-Manager Hash genannt
Passwörter werden in Windows mit einer Maximallänge von 15 Zeichen gespeichert
Das Passwort wird gespeichert in %SystemRoot%\system32\config\SAM
Das SAM File wird beim Bootvorgang decrypted mit SAM Lock Tool besser bekannt als Syskey
Der geladene Hash wird in die Registry gespeichert und wird für Authentifikationen genutzt
Das Tool befindet sich in %SystemRoot%\system32\config\system
Nach dem Bootvorgang ist das „SAM“ File und „system“ File gesperrt und kann nicht während der Windows runtime via regedit geöffnet werden

Wer ist John the Ripper?

- Entwickelt von Alexander Peslyak
- Version 1.0 released in 1996
- Gutes „Passwortrecoverytool“
  - Gehört zu den schnellsten Tools
    - Zumindest im „Incremented“ Mode
  - Gehört zu den meistgenutzten Tools
- Wird immer noch weiterentwickelt und supported
  - Neueste Version 1.8. von 2013

Grundlegende Funktionsweise

- John the Ripper und andere Programme vergleichen den Hashwert eines gesuchten Wortes mit einem, mit der selben Funktion, erstellten Hashwert eines ausgesuchten „Kandidaten“
- Wenn „Kandidat“ = gesuchtes Wort, dann sind die Hashwerte ebenfalls gleich
  - Probleme mit Hashkollisionen
- Dass Verfahren knackt also nicht den Passwort Algorithmus sondern „errät“ das richtige Wort durch den Vergleich der Resultate
- Für die „Kandidatenauswahl“ gibt es verschiedene Modi

Brute Force

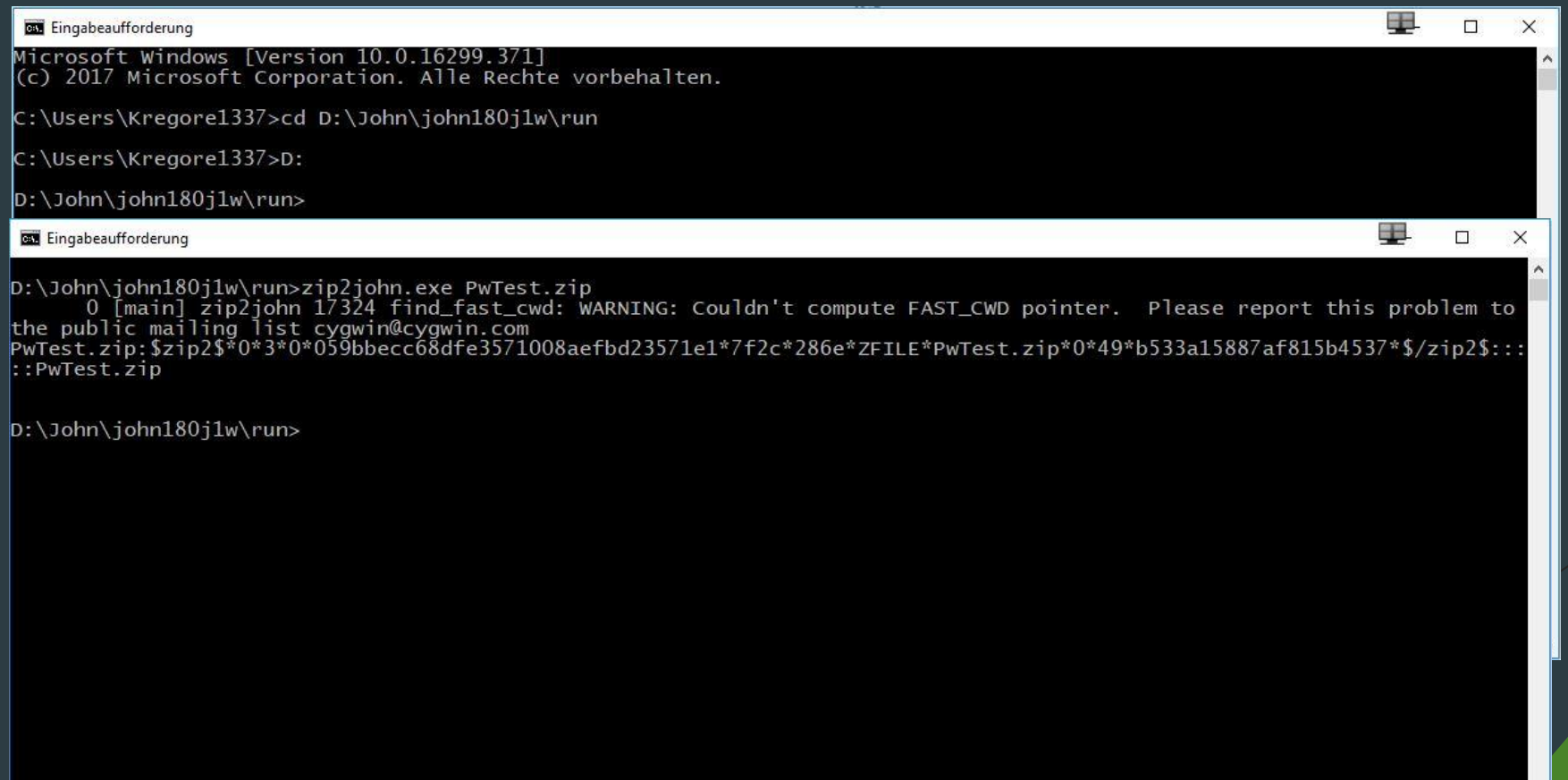
- Was ist Brute Force ?
  - Ein Angriff
  - Mit roher Gewalt
  - Durch wahlloses Ausprobieren
- Wo wird Brute Force benutzt?
  - „meistens“ auf Hochleistungsrechner
  - Viele Berechnungen pro Sekunde

Ergänzung:

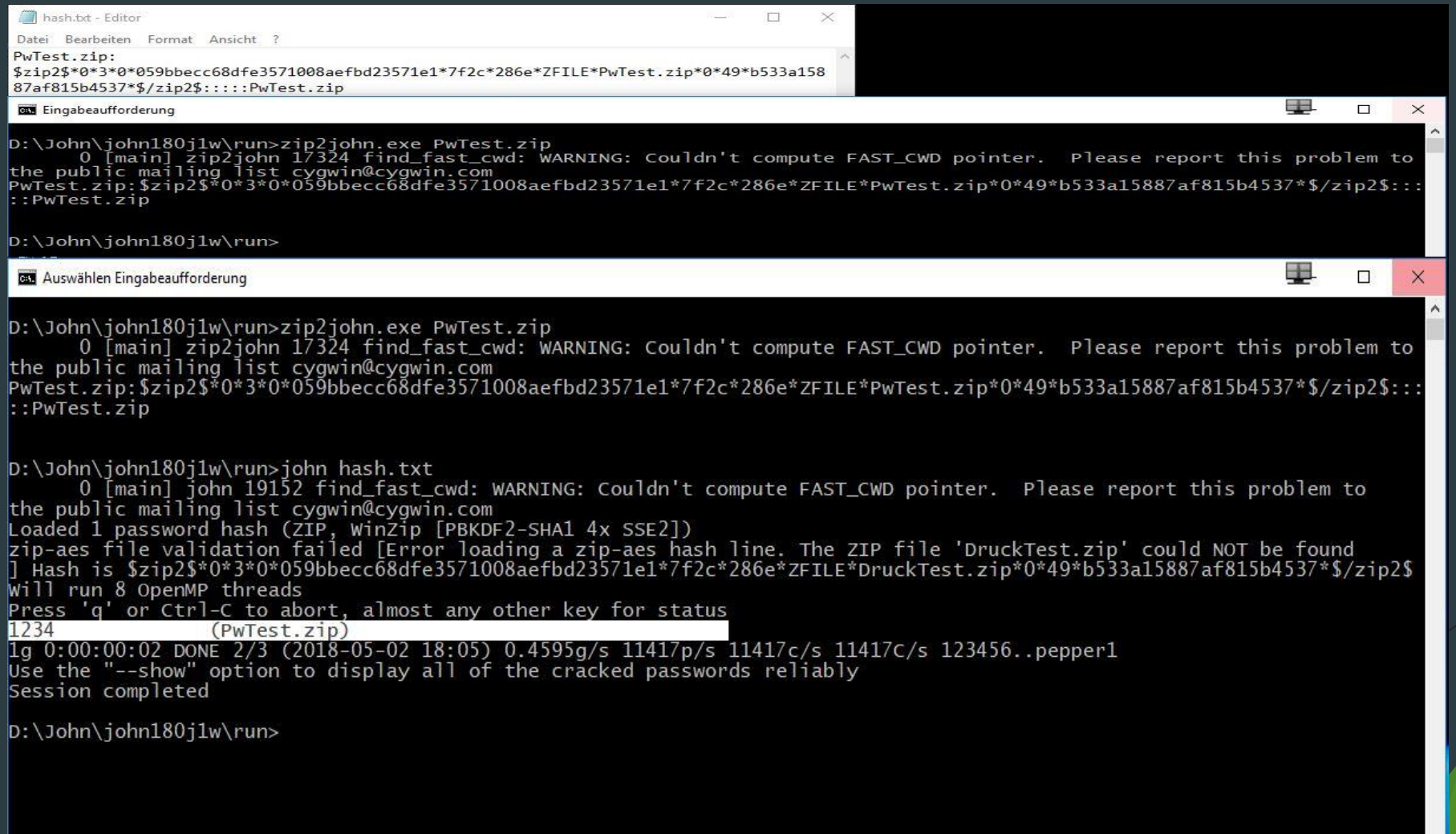
Brute-Force ist eine Methode, die versucht Passwörter oder Schlüssel durch automatisiertes, wahlloses Ausprobieren herauszufinden.
Es wird auch von der „Erschöpfende Suche“ gesprochen.
Hohe Anzahl von Kombinationen in kürzester Zeit

Single Mode Beispiel :

- Hier zeigen wir wie man Brute Force (Single Mode) auf eine einfache *.zip anwendet.



Single Mode Beispiel :



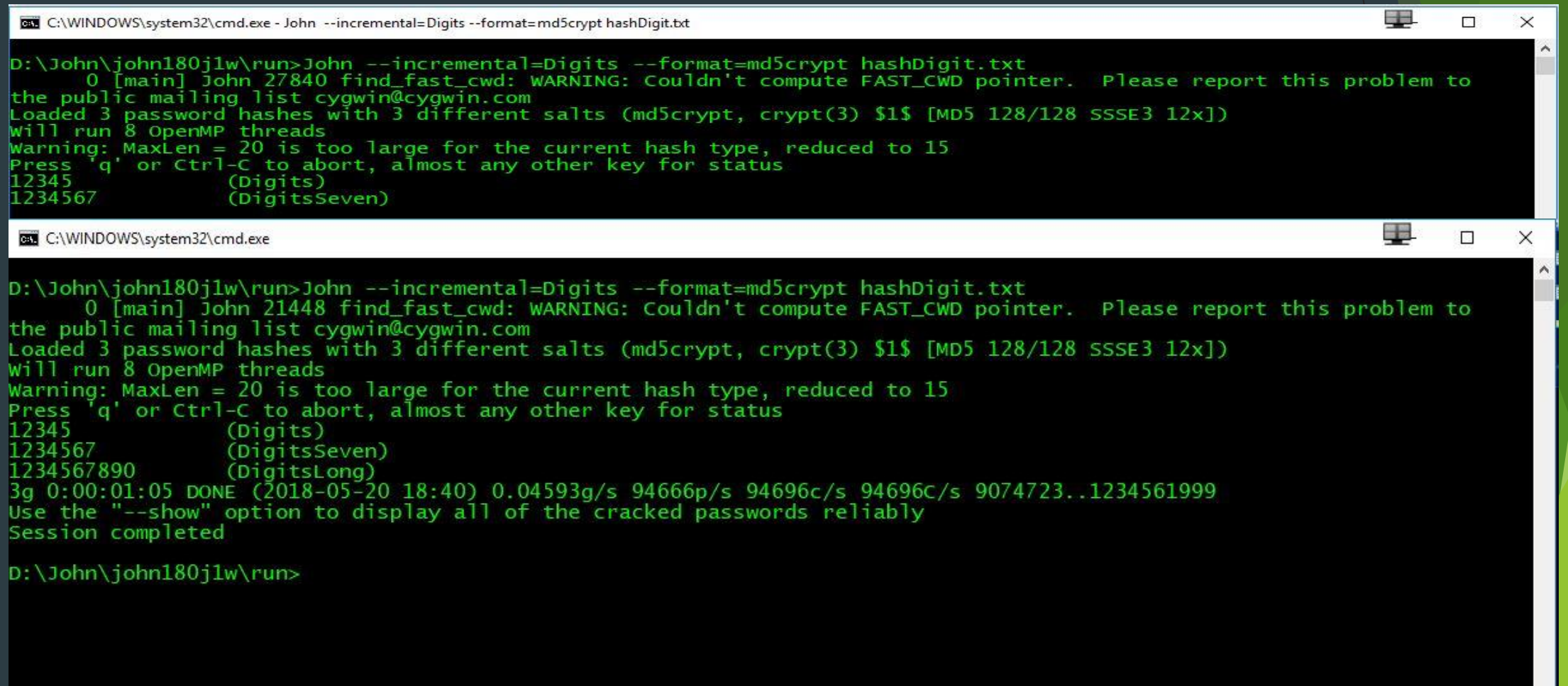
Schutz gegen Single Mode

- Schutz:
  - Lange Passwörter
    - Anzahl der möglichen Zeiten hoch Anzahl der Zeichen = mögliche Passwörter
    - Pro Zeichen mehr „bisher mögliche Passwörter“ * Anzahl an möglichen Zeichen
      - 6 Zeichen langes Passwort nur aus kleinen Buchstaben = 26^6= 308 915 776 mögliche Kombinationen
      - 7 Zeichen = 8.031.810.176 Möglichkeiten
  - Lange Schlüssel und komplexe Passwörter

Incremental Mode Beispiel :

- Der Incremental Mode ist eine Art des Brute Force Modus, im Incremental Mode werden nur definierte Regex auf den Hash angewendet.
- Das sieht folgendermaßen aus:

Incremental Mode Beispiel :



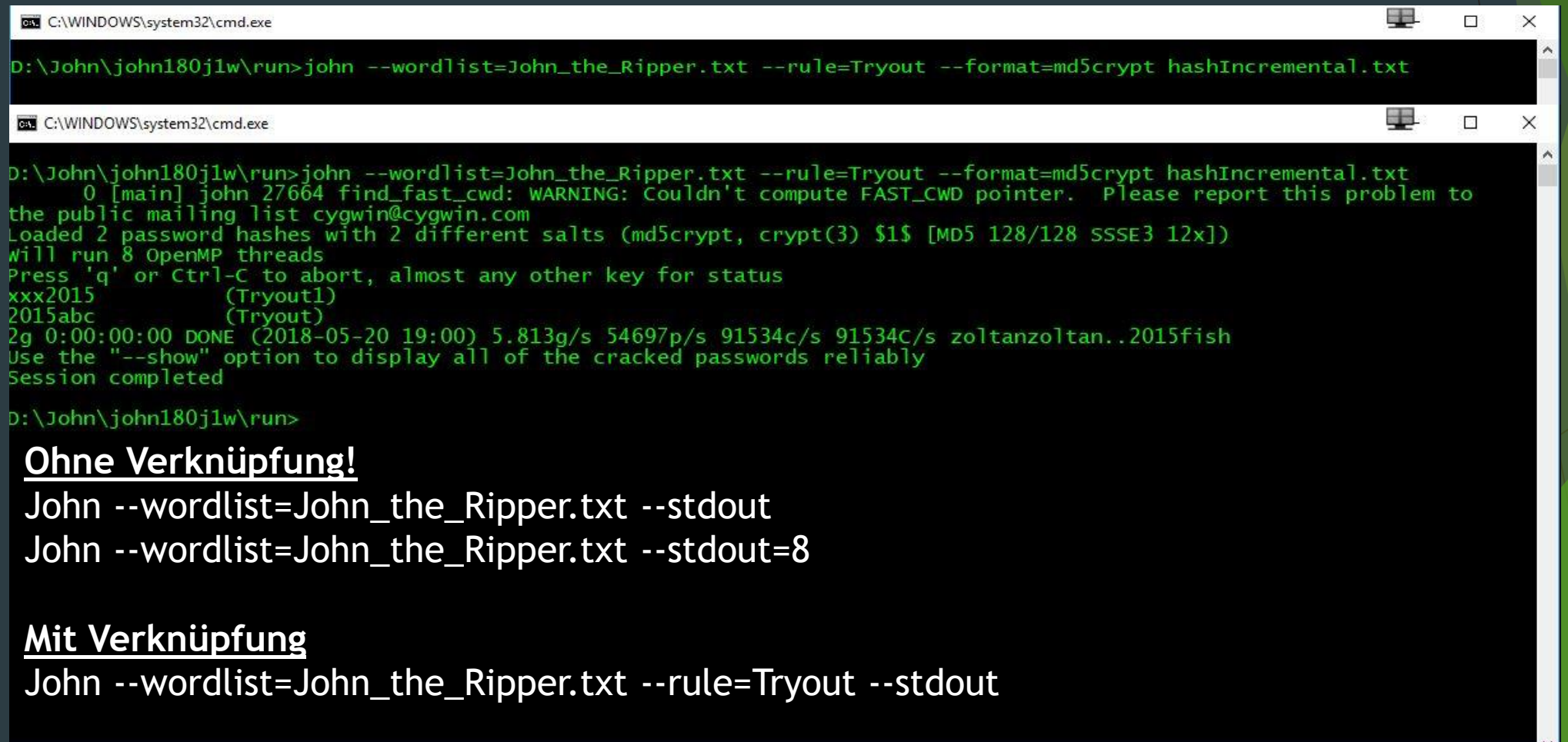
Incremental Mode mit Wordlist und Rule Beispiel :

- Im Incremental Mode ist es möglich mit einer Wordlist zu arbeiten und eine selbst definierte Rule miteinzubinden.
- Das sieht folgendermaßen aus:

```
# john.conf befindet sich im Order D://john/john180j1w/run
# A "Tryout" rule for just me to test something and it works!
[List.Rules:Tryout]
l
u
c
l r
l Az"2015"
d
l A0"2015"
A0"#"Az"#"
```

l #convert to lowercase
u #convert to uppercase
c #capitalize
l r #lowercase the word and reverse it (palindrome)
l #lowercase the word and append at end of the word
(Az) the number 2015
Az"2015"
d # duplicate
l A0"2015“ # lowercase the word and prepend at beginning of
the word (A0) the number 2015
A0"#"Az"# Add # to the beginning and end of the word

Incremental Mode mit Wordlist und Rule Beispiel :



Ohne Verknüpfung!

John --wordlist=John_the_Ripper.txt --stdout

John --wordlist=John_the_Ripper.txt --stdout=8

Mit Verknüpfung

John --wordlist=John_the_Ripper.txt --rule=Tryout --stdout

Schutz gegen Incremental Mode

- Sind Passwortvorgaben von Firmen Sinnvoll?
  - Nein!
- Schutz ist nur gewährleistet wenn:
  - Passwörter keine Folgen beinhalten
    - 1234, abcd, etc.
  - Passwörter regelmäßig geändert werden
  - Es keine vorgegebenen Kombinationen von Firmen gibt

Wordlist Mode

- Passwort raten anhand einer Liste von möglichen Passwörtern
  - Auch Wörterbuch genannt
  - Die Liste wird Stück für Stück durchgegangen und ausprobiert
  - Enthält meistens die üblichen Passwörter
  - John empfiehlt als Wordlist Openwall
    - Openwall= riesige Wordlist nach wahrscheinlichkeiten sortiert
- Beispiel
  - sschuck:6Finhqm.v$kgiVkr38bZXIUNLFJZQYa..mXmCmVYn7ohD/OkYgB568jVyhNm2ZQUrknk14kXZdMnAJsDynHQbgz7elqOQlV0:17670:0:99999:7:::
  - John --wordlist=<ausgewähltes Wörterbuch>

```
sschuck@sschuck-VirtualBox:~/Desktop/John the ripper$ john --wordlist=wordlist.txt password.txt
Loaded 1 password hash (crypt, generic crypt(3) [?/64])
Press 'q' or Ctrl-C to abort, almost any other key for status
passwort1234     (sschuck)
1g 0:00:00:00 100% 2.127g/s 204.2p/s 204.2c/s 204.2C/s 123456..pamela
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

Auszug der Wordlist

```
123456
12345
password
password1
123456789
12345678
1234567890
abc123
computer
tigger
1234
qwerty
money
carmen
mickey
secret
summer
internet
a1b2c3
123
service

canada
passwort1234
hello
ranger
shadow
baseball
donald
harley
hockey
letmein
maggie
```

Schutz vor Wordlist angriffen

- Keine Zusammenhängenden Wörter
- Keine Tastaturlayouts bei der Passwortwahl als Vorlage verwenden
  - Bsp: qwertzuio
- Keine Zahlenketten

- Vermeiden von offensichtlichen Zusammenhängen

Gegenmaßnahmen (Allgemein)

- Vermeidung von offensichtlichen Zusammenhängen
- Lange und komplexe Passwörter
- Keine Buchstabenfolgen
- Keine Zahlenfolgen

- Optimaler Schutz:
  - Passwort min. 1mal im Monat ändern
  - Nicht das gleiche Passwort
  - Nicht die selben Zahlen/Buchstaben verwenden
  - Min. Länge >= 15 Zeichen
    - Voraussetzung das System unterstützt lange Passwörter

John MPI

- Seit Version 1.7.2 Unterstützung der Multiprozessor Nutzung

Distributed John

- Der Versuch John the Ripper über eine Server und Clientartige Struktur auf mehreren Systemen gegen die selben Hashwerte strukturiert laufen zu lassen
  - Wurde leider eingestellt

Ähnliche Programme

- Hashcat
  - Multiple Hashes zeitgleich
  - Nutzung mehrerer Systemen
- Andere Lösungen auf Windows spezialisiert
  - Ophcrack
  - kon-boot

Rainbow Tables

- Eine Tabelle mit vorherberechneten Hashes
  - RainbowCrack