Rainbow Tables

Yukai Zang

Division of Science and Mathematics
University of Minnesota, Morris
Morris, Minnesota, USA 56267
zangx040@morris.umn.edu

Table of contents

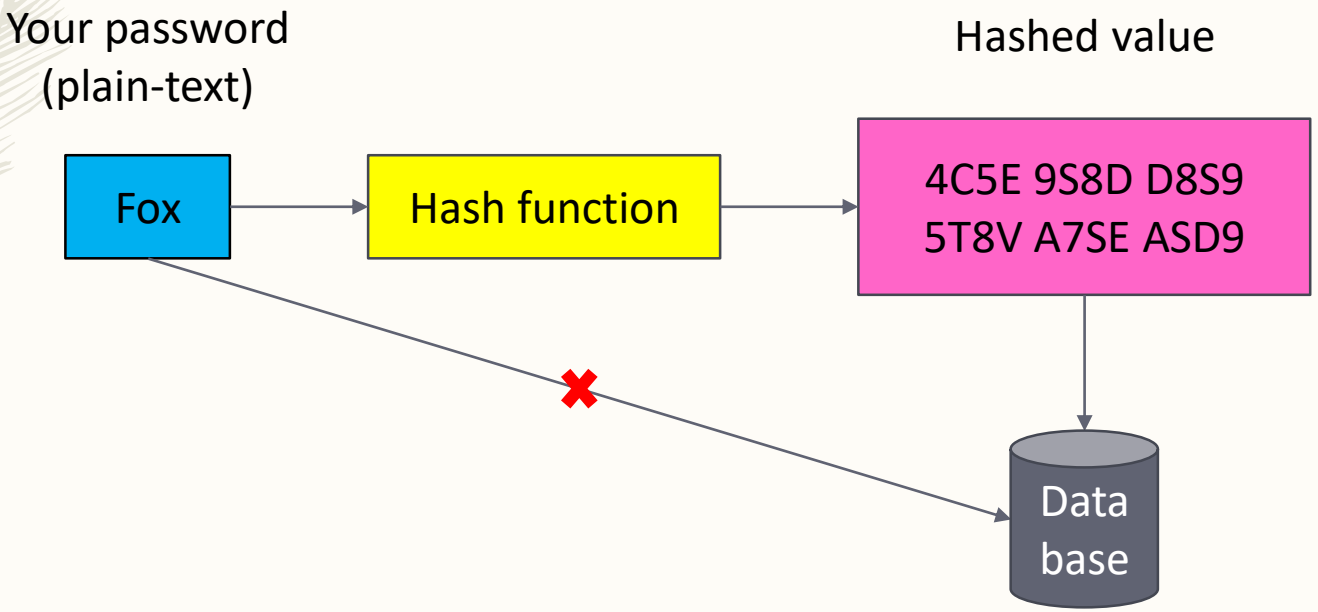
- Introduction & Background
- Rainbow table
- Create rainbow tables (offline stage)
- Use rainbow tables (online stage)
- Tests
- Conclusion

Table of contents

- Introduction & Background
- Rainbow table
- Create rainbow tables (offline stage)
- Use rainbow tables (online stage)
- Tests
- Conclusion

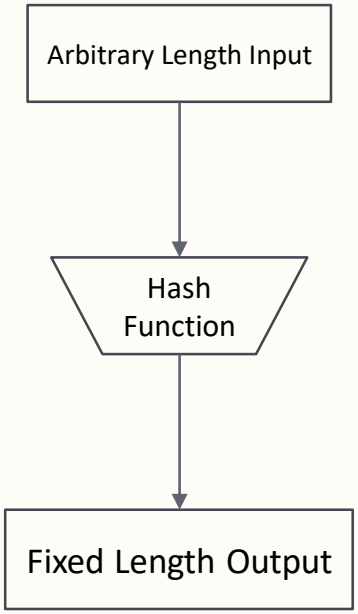
Introduction & Background

Introduction & Background



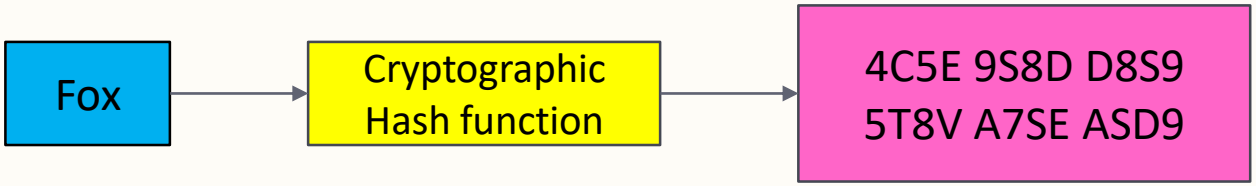
Introduction & Background

- Hash function
  - Map data of arbitrary size onto data of fixed size



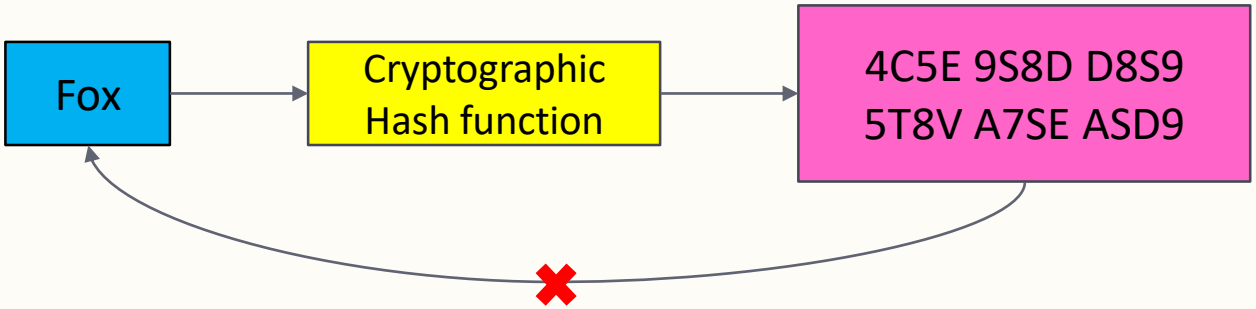
Introduction & Background

- Cryptographic hash function
  - Same plain-text result in same hashed value;



Introduction & Background

- Cryptographic hash function
  - Same plain-text result in same hashed value;
  - Fast to compute;
  - Infeasible to revert back to plain-text from hashed value;

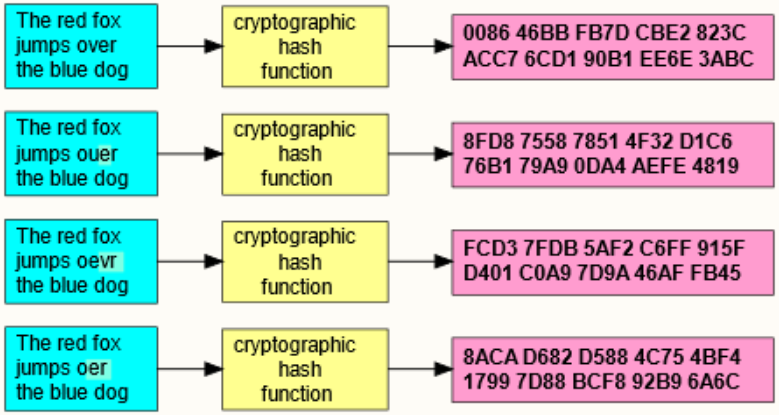


Introduction & Background

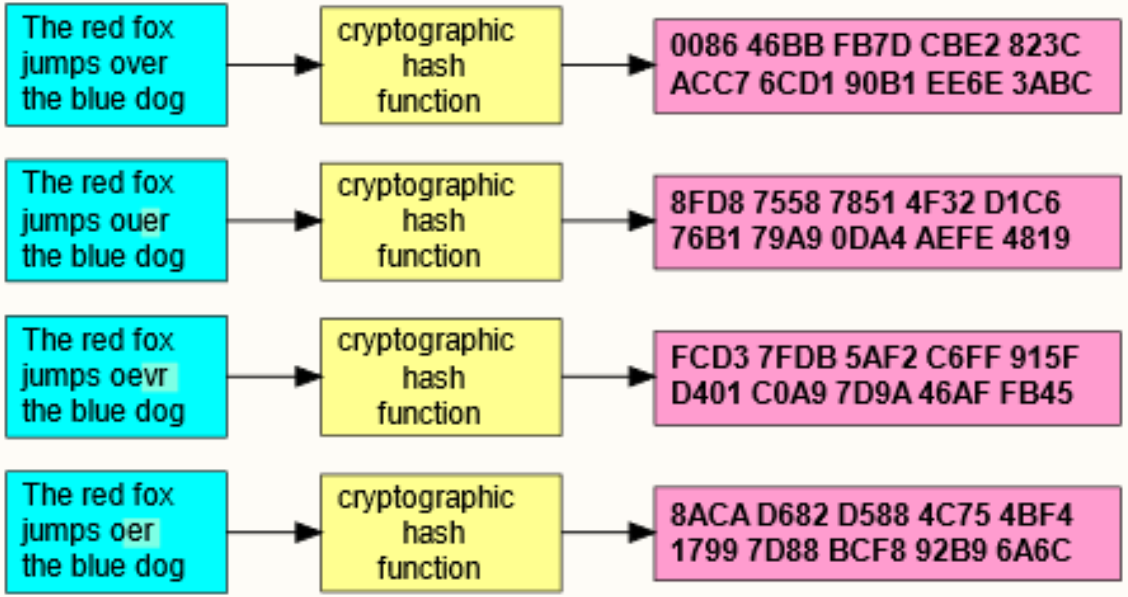
- Cryptographic hash function
  - Same plain-text result in same hashed value;
  - Fast to compute;
  - Infeasible to revert back to plain-text from hashed value;
  - Small change(s) in plain-text will cause huge changes in hashed value;

Introduction & Background

- Cryptographic hash function
  - Small change(s) in plain-text will cause huge changes in hashed value;



Introduction & Background

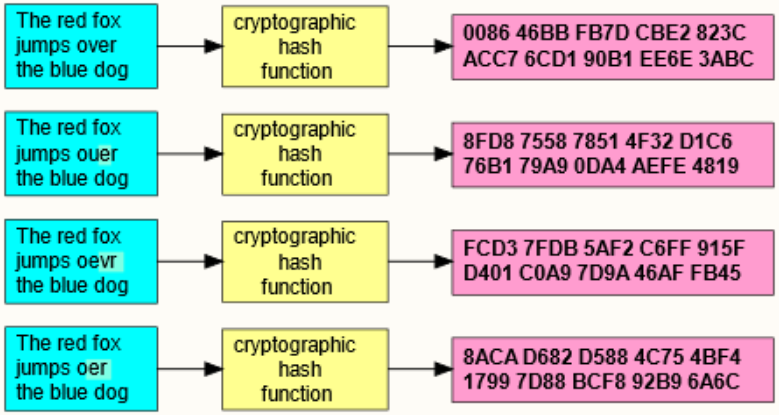


Introduction & Background

- Cryptographic hash function
  - Same plain-text result in same hashed value;
  - Fast to compute;
  - Infeasible to revert back to plain-text from hashed value;
  - Small change(s) in plain-text will cause huge changes in hashed value;
  - Infeasible to find two different plain-text with the same hashed value.

Introduction & Background

- Cryptographic hash function
  - Same plain-text result in same hashed value;
- Brute force



Introduction & Background

Password length	Alphanumeric	Days
1	62	0
2	3844	0
3	238328	0
4	14776336	0
5	916132832	0
6	56800235584	0
7	3.52 e+12	1
8	2.18 e+12	42
9	1.35 e+12	2599
10	8.39 e+12	161156

- Lookup table
- 8 digits; 62 characters
- 222 trillion combinations
- 2 quadrillion bytes
  = 1800 terabytes
- 32 days with i7

Plain-text	Hash value
aaaaaaaa	7D9SXF
aaaaaaab	7WS4G5
aaaaaaac	5F2V6D
…	…
00000000	8CVIDF
00000001	1QSD9F
…	…

Table of contents

- Introduction & Background
- Rainbow table
- Create rainbow tables (offline stage)
- Use rainbow tables (online stage)
- Tests
- Conclusion

Rainbow table

- Based on the idea introduced by Martin Hellman in 1980
- Improved by Philippe Oechslin in 2003
- Two stages
  - Offline stage
  - Online stage

Table of contents

- Introduction & Background
- Rainbow table
- Create rainbow tables (offline stage)
- Use rainbow tables (online stage)
- Tests
- Conclusion

Rainbow table

- Offline stage
  - Create tables
    - *Chains*
    - *Cryptographic hash function (h)*
    - *Reduction function (r)*
    - *Collision*

Rainbow table

- Chains
  - Only start point t_0 and end point h_2 matters
  - Tables look like:

$$S_0 = t_0 \xrightarrow{h} h_0 \xrightarrow{r_0} t_1 \xrightarrow{h} h_1 \xrightarrow{r_1} t_2 \xrightarrow{h} h_2 = E_0$$

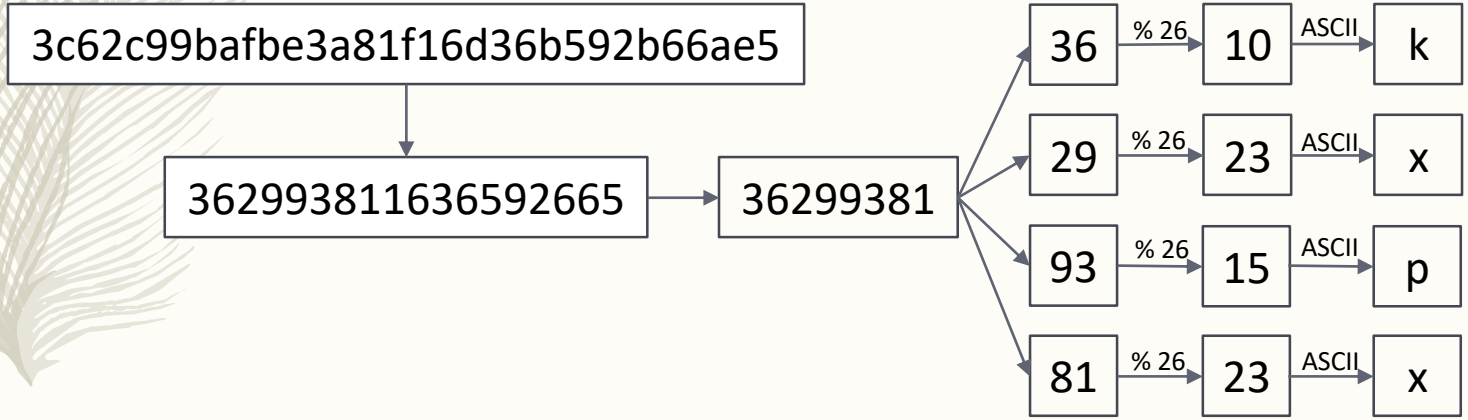
Text		Hash
aaaa	⇝	3c62c99bafbe3a81f16d36b592b66ae5
aaab	⇝	7991ef7c61d85c505adb63b70553b0ce
aaac	⇝	ef834a79dc0ac78e355859ec4f698d0b
aaad	⇝	3bb75d8a8effe7c5e18aC6670a3c3346
⋮	⇝	⋮

Rainbow table

- Cryptographic hash function (h)
  - MD5, SHA-1, SHA-2, SHA-3, BLAKE2, and etc.
  - Slight different in time due to different algorithm used

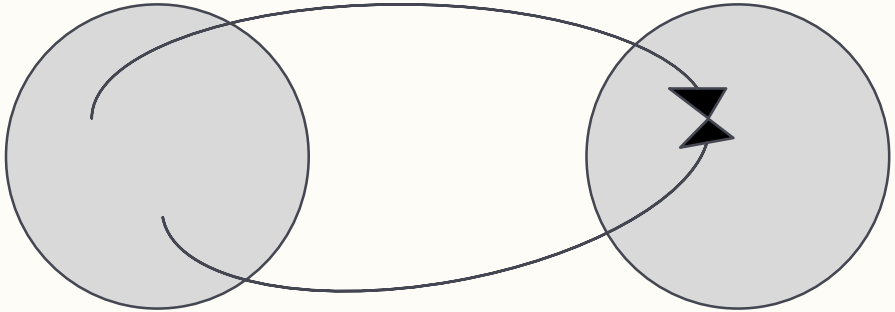
Rainbow table

- Reduction function (r)
  - Map data from the set of hashed values to the set of all plain-texts
  - Example:



Rainbow table

- Collision
  - When we map two different thing into the same value, no matter which way, it is called collision

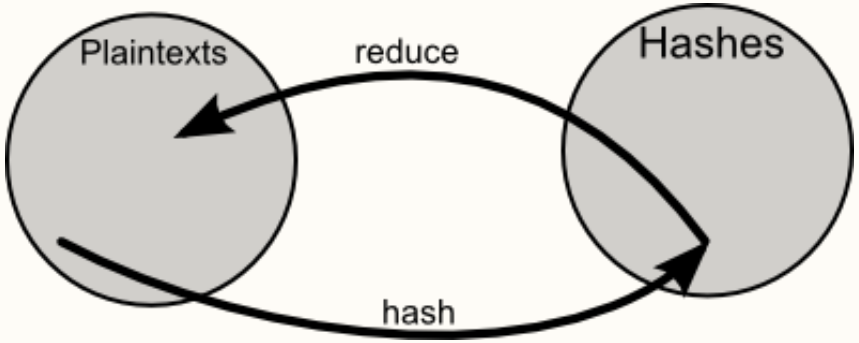


Rainbow table

- Collision
  - The size of the set of plain-texts and the set of hashed value will be different normally
  - When we try to map from a larger set onto a smaller set, collisions occur more frequently (Pigeonhole principle)

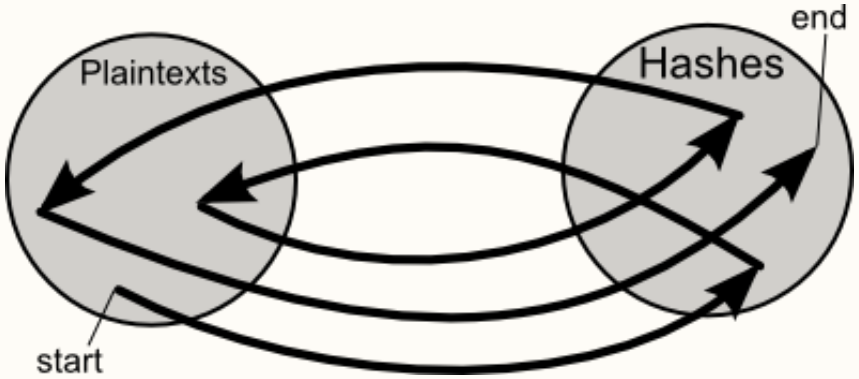
Rainbow table

– Generate steps:



Rainbow table

- Generate steps:



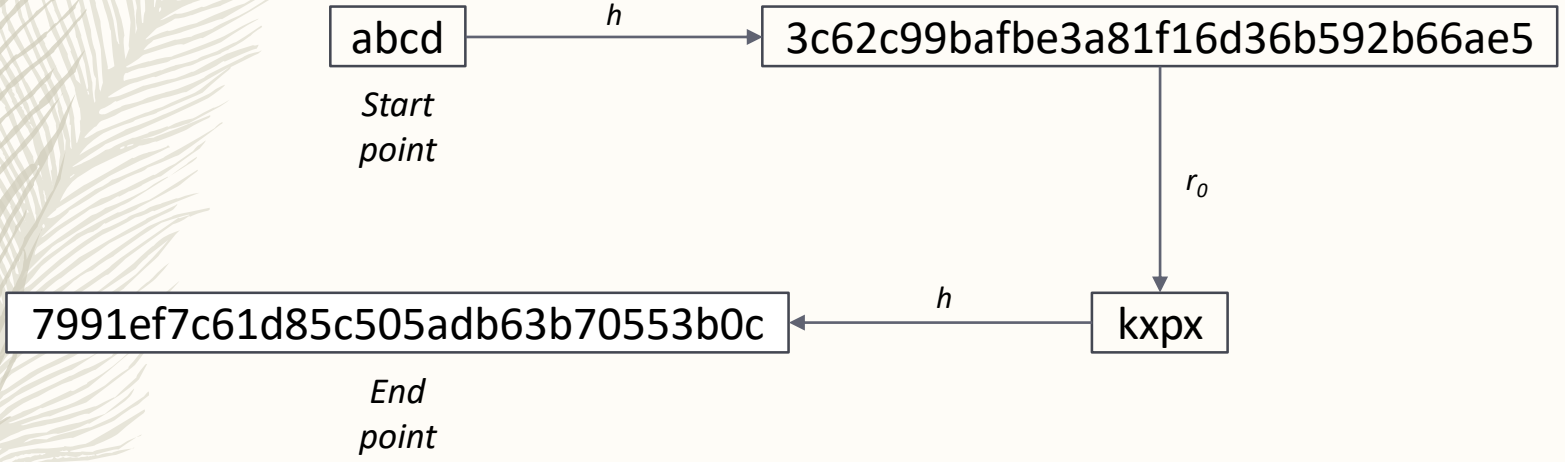
Rainbow table

- The structure of a rainbow table:

$$
\begin{array}{|c|c|c|c|c|c|c|c|c|c|}
\hline
S_0 = t_{0,0} & \xrightarrow{h} & h_{0,0} & \xrightarrow{r_0 \circ h} & h_{0,1} & \xrightarrow{r_1 \circ h} & \dots & \xrightarrow{r_{n-2} \circ h} & h_{0,n-1} & \xrightarrow{r_{n-1} \circ h} & h_{0,n} = E_0 \\
S_1 = t_{1,0} & \xrightarrow{h} & h_{1,0} & \xrightarrow{r_0 \circ h} & h_{1,1} & \xrightarrow{r_1 \circ h} & \dots & \xrightarrow{r_{n-2} \circ h} & h_{1,n-1} & \xrightarrow{r_{n-1} \circ h} & h_{1,n} = E_1 \\
\vdots & & \vdots & & \vdots & & \vdots & & \vdots & & \vdots \\
S_j = t_{j,0} & \xrightarrow{h} & h_{j,0} & \xrightarrow{r_0 \circ h} & h_{j,1} & \xrightarrow{r_1 \circ h} & \dots & \xrightarrow{r_{n-2} \circ h} & h_{j,n-1} & \xrightarrow{r_{n-1} \circ h} & h_{j,n} = E_j \\
\vdots & & \vdots & & \vdots & & \vdots & & \vdots & & \vdots \\
S_m = t_{m,0} & \xrightarrow{h} & h_{m,0} & \xrightarrow{r_0 \circ h} & h_{m,1} & \xrightarrow{r_1 \circ h} & \dots & \xrightarrow{r_{n-2} \circ h} & h_{m,n-1} & \xrightarrow{r_{n-1} \circ h} & h_{m,n} = E_m \\
\hline
\end{array}
$$

Rainbow table

– Example:



Rainbow table

- Example (collision):

  t_0 = *"abcd"* → $r_0(h(t_0))$ = *"defg"*

  t_1 = *"abdc"* → $r_0(h(t_1))$ = *"defg"*

  we will have two chains with different start points but the same end point.

  - Clean table
    - Only keep one chain with the same end point.

Table of contents

- Introduction & Background
- Rainbow table
- Create rainbow tables (offline stage)
- Use rainbow tables (online stage)
- Tests
- Conclusion

Rainbow table

- Online stage
  - Actual search
  - Example:

    We have a rainbow table with chain length = 3:

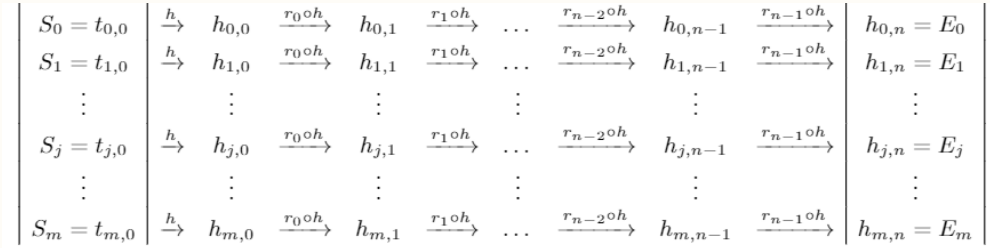
    Rainbow table: $t_0 \sim \rightarrow h_2$

    Chain: $t_0 \overset{h}{\rightarrow} h_0 \overset{r_0}{\rightarrow} t_1 \overset{h}{\rightarrow} h_1 \overset{r_1}{\rightarrow} t_2 \overset{h}{\rightarrow} h_2$

    Assuming the password we want to search for hash value:

    3C62C9 & 77CC7F

Text	Hash
aaaa	3C62C9
aaab	7991EF
aaac	EF834A
spkn	3BB75D

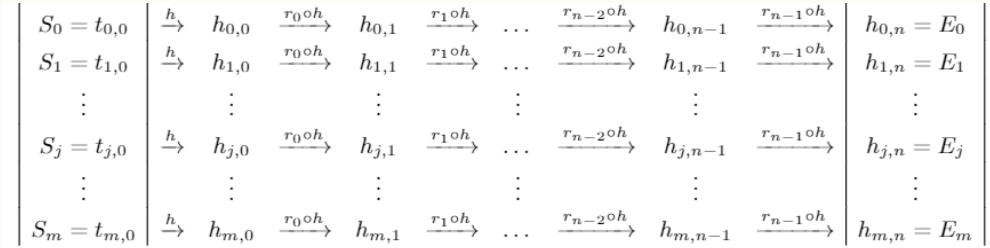


Chain: $t_0 \xrightarrow{h} h_0 \xrightarrow{r_0} t_1 \xrightarrow{h} h_1 \xrightarrow{r_1} t_2 \xrightarrow{h} h_2$

Text	Hash
aaaa	3C62C9
aaab	7991EF
aaac	EF834A
spkn	3BB75D

1. According to the given hash value y=h(x)= 3C62C9, search through the column for the ending points in all tables;
2. We find it at the first row with the plain-text “aaaa”;
3. We create a chain from “aaaa”, and search for the original hash (3C62C9) in the chain; if we find it, it returns the password at a given index (indexHash -1). The chain is:

   $aaaa \xrightarrow{h}$ 584C19 $\xrightarrow{r_0}$ kcyl $\xrightarrow{h}$ 48950E $\xrightarrow{r_1}$ **puwr** $\xrightarrow{h}$ 3C62C9
4. In this case, the plain-text we wanted is “puwr”.



Chain: $t_0 \xrightarrow{h} h_0 \xrightarrow{r_0} t_1 \xrightarrow{h} h_1 \xrightarrow{r_1} t_2 \xrightarrow{h} h_2$

Text	Hash
aaaa	3C62C9
aaab	7991EF
aaac	EF834A
spkn	3BB75D

1. According to the given hash value y=h(x)= 77CC7F, search through the column for the ending points in all tables;
2. There is no matched E_j.
3. Compute the $h(r_1(77CC7F))$ = 48950E;
4. 48950E is not in the table;
5. Compute the $h(r_1(h(r_0(77CC7F))$ = 3BB75D;
6. We have a matched E_j, which is the last row with plain-text “spkn”;
7. We create a chain from “spkn”, and search for the original hash (77CC7F) in the chain; if we find it, it returns the password at a given index (indexHash -1). The chain is:

   spkn $\xrightarrow{h}$ 77CC7F $\xrightarrow{r_0}$ puwr $\xrightarrow{h}$ 584C19 $\xrightarrow{r_1}$ uhtc $\xrightarrow{h}$ 3BB75D
8. In this case, the plain-text we wanted is “spkn”.

The location of y=h(x)

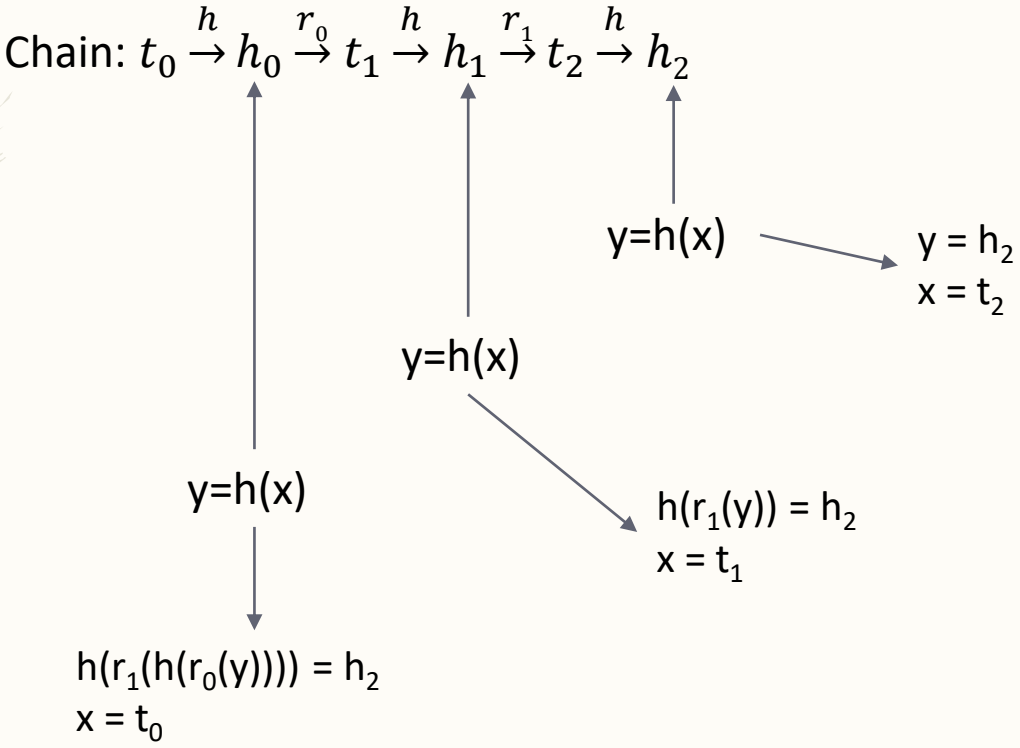


Table of contents

- Introduction & Background
- Rainbow table
- Create rainbow tables (offline stage)
- Use rainbow tables (online stage)
- Tests
- Conclusion

Tests

- Chain length;

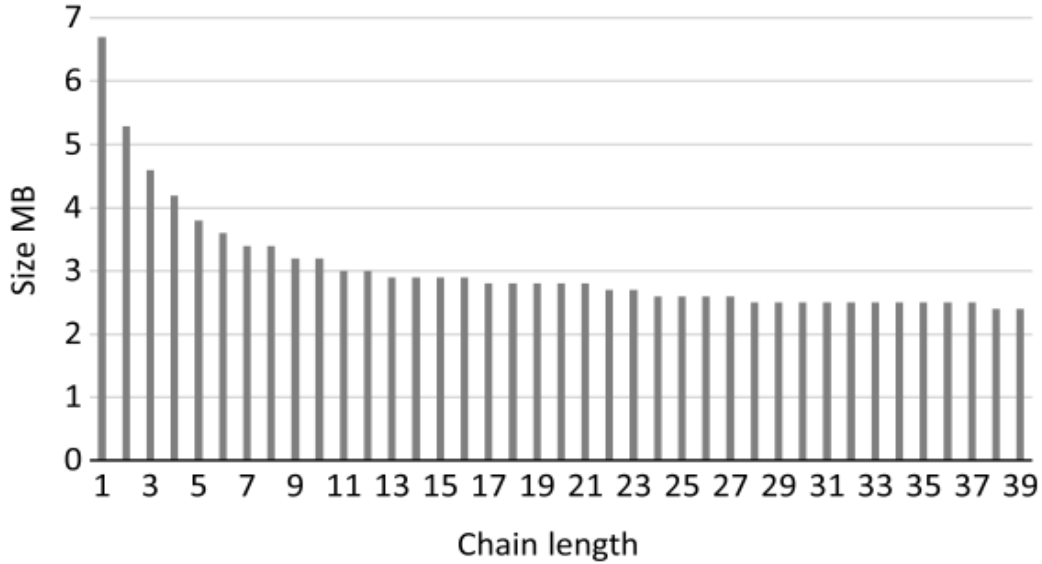


Fig. 2. The size of Rainbow Tables in dependence on the length of chain.

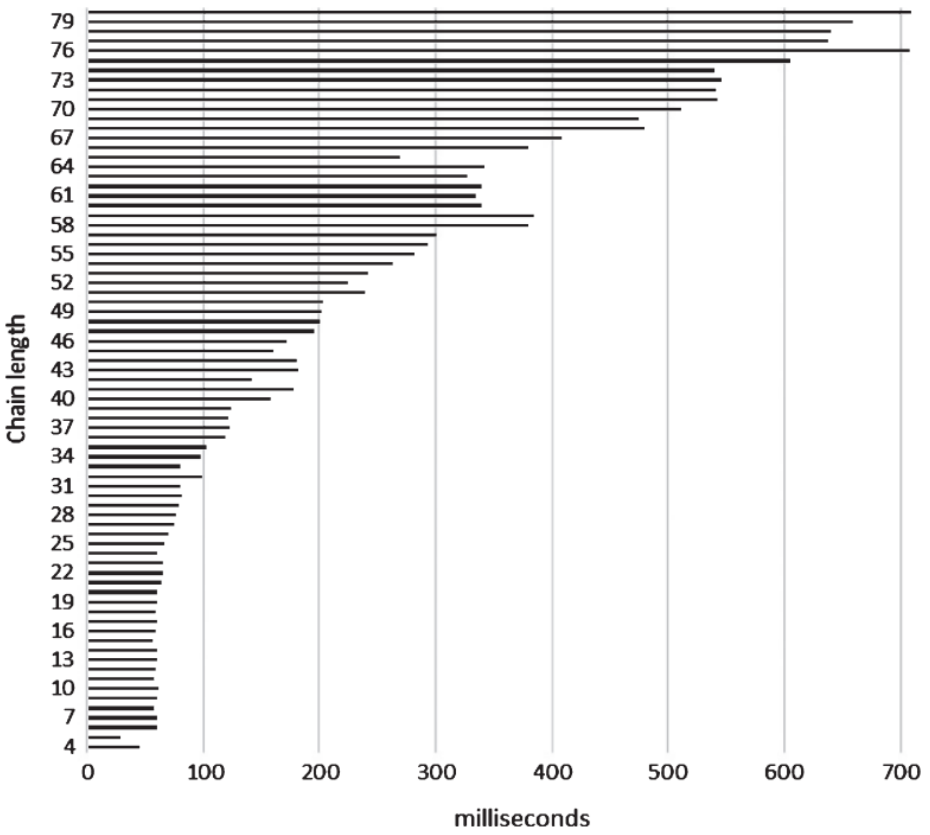


Fig. 1. Dependence on the chain length of the average time to find the password.

Table of contents

- Introduction & Background
- Rainbow table
- Create rainbow tables (offline stage)
- Use rainbow tables (online stage)
- Tests
- Conclusion

Conclusion

- Time & space;

Acknowledgement

- Great thanks to Peter Dolan, KK Lamberty, and Elena Machkasova.

Reference

1. Horáleka, J., Holík, F., Horák, O., Petr, L., & Sobeslav, V. (2017). Analysis of the use of Rainbow Tables to break hash. Journal of Intelligent & Fuzzy Systems,32(2):1523 – 1534, 2017
2. G. Avoine and X. Carpent. Heterogeneous rainbowtable widths provide faster cryptanalyses. InProceedings of the 2017 ACM on Asia Conference onComputer and Communications Security, ASIA CCS'17, pages 815–822, New York, NY, USA, 2017. ACM.
3. Cryptographic hash function. Cryptographic hashfunction — Wikipedia, the free encyclopedia, 2019.[Online; accessed 15-March-2019]

Q&A