Introduction Problem Domain Secret Storing Properties What About the E-Voter Missing If in Doubt Analyses

# KryptonIT: The Discovery of a New Cryptographic System for Storing Secrets

Reto E. Koenig

University of Fribourg

&

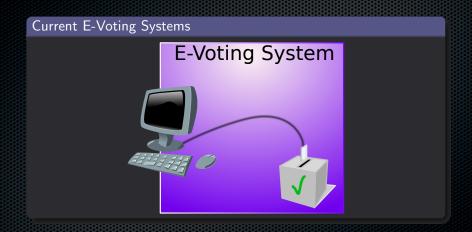
Bern University of Applied Sciences

25.03.2011





What you should know about building new cryptographic systems Ronald Rivest: "Never build your own cryptographic system!"



# Voter's view on a current E-Voting Systems

- It is Understandable
- It is Simple to Use
- Fast
- "Cheap"



# Adversary's view on a current E-Voting Systems Do not Underestimate the Power of the Coercer

- You Shall Not Vote
- You Shall Vote
- You Shall Vote As I Say
- I randomize Your Vote
- I Vote for You
- I am Watching You
- I Know How You Voted
   Last Summer
- It is Fast
- It is Scalable
- It is "Cheap"



# Overall view on a current E-Voting Systems

- It is Simple
- It is Fast
- It is Cheap
- It is completely insecure





# Cryptographer's view on their E-Voting Systems

- It is Coercion-Resistant
- It is End-To-End Verifiable
- It is secure
- The voter has to remember "some" "secrets"

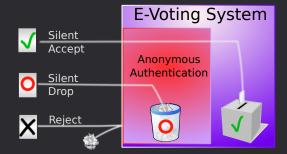


Introduction Problem Domain Secret Storing Properties What About the E-Voter Missing If in Doubt Analyses

#### Some Secrets?

# E2E verifiable coercion resistant E-Voting systems require:

- 3 different kind of credentials with very high entropy
- kept top secret by each voter

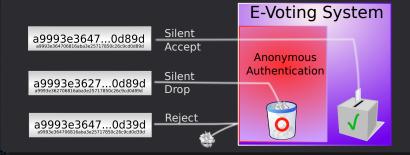


Introduction Problem Domain Secret Storing Properties What About the E-Voter Missing If in Doubt Analyses

#### Some Secrets?

E2E verifiable coercion resistant E-Voting systems require:

- 3 different kind of credentials with very high entropy
- kept top secret by each voter



duction Problem Domain Secret Storing Properties What About the E-Voter Missing If in Doubt Analyses

## A Realistic Example With 20 Credentials

In order to render the E-Voting System coercion resistant, each voter...

- ...needs to secretly store several dozens credentials
- ...has to discriminate doubtless between credentials for 'Accept' and 'Drop'.<sup>a</sup>
  - ...is not allowed to mark any credential
- ...shall never unveil the amount of possessed secrets (They vary per voter)

f-0ccd6641d45ef2efd0946c3a67773ac268e9e3 elaSSS017e3105ef0667e68e6ef773Bar26a9b32 f-088213d95897fc7e9479790c2ale664924537 f5abae583295649547c13be2c54bcbfb3268f87 3a706238df856ecce10bbb93cs317893c46691 blaa9bad3a87fe886cc610bb93cs317893c46691 blaa9bad3a87fe886cc6468730d6644750f0d88 22be0892841c37e6511e85e74432a45c6f35537594

fd8b823d985947fc7d9f470907ca18ed68243557



<sup>&</sup>lt;sup>a</sup> E-voting system accepts both without returning any hint

on **Problem Domain** Secret Storing Properties What About the E-Voter Missing If in Doubt Analyses

# A Realistic Example With 20 Credentials

In order to render the E-Voting System coercion resistant, each voter...

- ...needs to secretly store several dozens credentials
- ...has to discriminate doubtless between credentials for 'Accept' and 'Drop'.<sup>a</sup>
- ...is not allowed to mark any credential
- ...shall never unveil the amount of possessed secrets (They vary per voter)

fc0ccd641d45e72e7d946c3a6f7f3ac2669e2 elaSS5917e31d56f06fc686e68f7238a7axb532 fc8822396597fc7694f796072a18e69423337 f5abae582297649547c13be2c54bcbfb1269f8f3 3a710d284f856bccc16bbb92c3517893c46691 blas98d33a9ffc896c496373008644f59fdd88 22be80g28dc27e6511e85e7a422a45c49f3573008

fd8b823d985947fc7d9f470907ca18ed68243557

<sup>&</sup>lt;sup>a</sup> E-voting system accepts both without returning any hint

duction Problem Domain Secret Storing Properties What About the E-Voter Missing If in Doubt Analyses

## A Realistic Example With 20 Credentials

In order to render the E-Voting System coercion resistant, each voter...

- ...needs to secretly store several dozens credentials
- ...has to discriminate doubtless between credentials for 'Accept' and 'Drop'.<sup>a</sup>
- ...is not allowed to mark any credential
- ...shall never unveil the amount of possessed secrets (They vary per voter)



<sup>&</sup>lt;sup>a</sup> E-voting system accepts both without returning any hint

But How Should the Voter Store those Secrets?

Indeed it is not possible to store these secrets with some keys (passwords) in a current crypto-systems. (→ Discussion)

# Our View on the Secret-Storage System

#### The system...

- ...allows to choose freely n (usually low entropy) keys
- ...allows to choose freely n (usually high entropy) secrets
- ...has to store multiple secrets in one storage (aka cipher)
- ...has to retrieve only the secret correlated to the key
- ...has to have all properties of a (symmetric) crypto-system





# Let's get Formal...

#### Prerequisits

S = secret space, set of all possible secrets (typically high-entropy)

 $\mathcal{K}=$  key space, set of all possible keys (typically low-entropy)

 $S = (s_1, \dots, s_n)$ ,  $s_i \in S$ , an n-tuple of (not necessarily distinct) secrets  $(n \ge 1)$ 

 $\overline{\mathcal{K}} = (k_1, \ldots, k_n), \; k_i \in \mathcal{K}$ , an n-tuple of distinct keys  $n \geq 1$ 

 $\mathcal{C}$  =storage space, the set of all possible storages

c =a particular storage

# Functions of the Secret-Storing System

store:  $S^n \times \mathcal{K}^{(n)} \longrightarrow \mathcal{C}$ 

storage function, where  $\mathcal{K}^{(n)} \subseteq \mathcal{K}^n$  is the set of all

admissible key tuples (with distinct keys)

retrieve :  $\mathcal{C} \times \mathcal{K} \longrightarrow \mathcal{S}$ 

the retrieval function

store<sub>K</sub>(S) =  $c \in C$ , storing the n-tuple of the secrets  $S \in S^n$  with the n-tuple of distinct keys  $K \in K^{(n)}$ 

 $\mathsf{retrieve}_{k_i}(c) = s_i \in \mathcal{S} \mathsf{retrieval} \mathsf{with} \mathsf{key} k_i$ 

$$retrieve_{k_i}(store_K(S)) = s_i$$

# Properties of the Secret-Stroring System

Required to possess the cryptographic properties of a traditional symmetric crypto-system:

- Retrieving  $s_i$  from c does not disclose any information about the other secrets in c
- Applying K on c returns S
- Serves a conditional entropy  $\overline{H(S|c)}$  which is equal to H(S)
- Applying K' on c where  $K' \neq K$  does return S with a probability of  $\frac{1}{|S|}$

#### Definition

A secret-storing system of order n,

$$\Sigma = (S, K, C, \text{store}, \text{retrieve}),$$

consists of a secret space S, a key space K, a storage space C, and two functions store and retrieve with properties as introduced above.

$$S = (s_1, \dots, s_i, \dots, s_n)$$

$$K = (k_1, \dots, k_i, \dots, k_n)$$

$$c = \text{store}_K(S)$$

$$c = \sum_{i=1}^{t} a_i x^i$$



Key Space

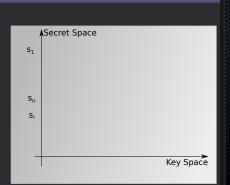
$$S=(s_1,\ldots,s_i,\ldots,s_n)$$

$$K = (k_1, \ldots, k_i, \ldots, k_n)$$

$$c = \operatorname{store}_K(S)$$

$$c = LagrangeInterpPol_K(S)$$

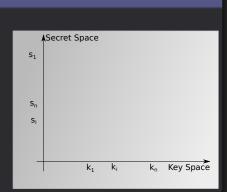
$$c = \sum_{i=1}^{t} a_{z} x^{z}$$



$$S = (s_1, \ldots, s_i, \ldots, s_n)$$
  
$$K = (k_1, \ldots, k_i, \ldots, k_n)$$

$$c = \operatorname{store}_{K}(S)$$

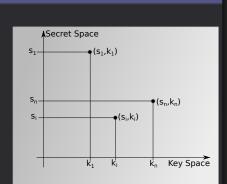
$$c = \sum a_z x^z$$



$$S = (s_1, \dots, s_i, \dots, s_n)$$
  
 $K = (k_1, \dots, k_i, \dots, k_n)$   
 $c = \text{store}_K(S)$ 

 $c = \mathsf{LagrangeInterpPol}_K(S)$ 

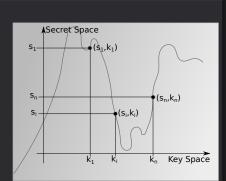
$$c = \sum_{z=1}^{L} a_z x^z$$



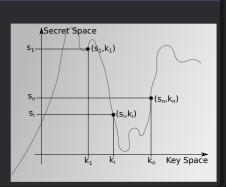
$$S = (s_1, \dots, s_i, \dots, s_n)$$
  
 $K = (k_1, \dots, k_i, \dots, k_n)$   
 $c = \text{store}_K(S)$ 

 $c = \mathsf{LagrangeInterpPol}_{\mathcal{K}}(S)$ 





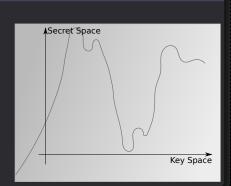
$$S = (s_1, \dots, s_i, \dots, s_n)$$
  
 $K = (k_1, \dots, k_i, \dots, k_n)$   
 $c = \text{store}_K(S)$   
 $c = \text{LagrangeInterpPol}_K(S)$   
 $c = \sum_{z=0}^t a_z x^z$ 



$$c = \sum_{z=0}^{t} a_z x^z$$
$$s_i = \text{retrieve}_{k_i}(c)$$

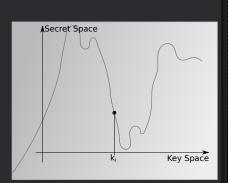
$$s_i = \text{retrieve}_{k_i}(c)$$

$$s_i = f(k_i) = \sum_{z=0}^t a_z k_i^z$$



$$c = \sum_{z=0}^{t} a_z x^z$$
  
 $s_i = \text{retrieve}_{k_i}(c)$ 

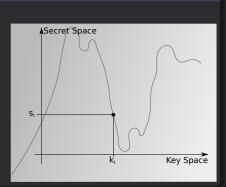
$$s_i = f(k_i) = \sum_{z=0}^{\tau} a_z k_i^z$$



$$c = \sum_{z=0}^{t} a_z x^z$$

$$s_i = \text{retrieve}_{k_i}(c)$$

$$s_i = f(k_i) = \sum_{z=0}^{t} a_z k_i^z$$

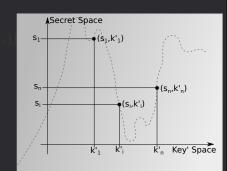


$$S = (s_1, \ldots, s_i, \ldots, s_n)$$
  
 $K = (k_1, \ldots, k_i, \ldots, k_n)$ 

$$K' = (H(k_1), \ldots, H(k_i), \ldots, H(k_i))$$

$$c = \mathsf{LagrangeInterpPol}_{\mathcal{K}'}(S)$$

$$z=0$$



$$S = (s_1, \ldots, s_i, \ldots, s_n)$$
  
 $K = (k_1, \ldots, k_i, \ldots, k_n)$ 

$$c = \mathsf{store}_{\mathcal{K}}(S)$$

$$K' = (H(k_1), \ldots, H(k_i), \ldots, H(k_n))$$

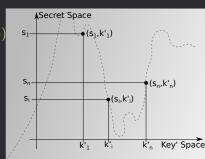
$$c = \mathsf{LagrangeInterpPol}_{\mathcal{K}'}(S)$$

$$c = \sum_{z=0}^{L} a_z x^z$$

$$s_i = \text{retrieve}_{k_i}(c)$$

$$k_i' = H(k_i)$$

$$s_i = f(k_i') = \sum_{j=0}^{L} a_z k_i'$$



$$S = (s_1, \ldots, s_i, \ldots, s_n)$$
  
$$K = (k_1, \ldots, k_i, \ldots, k_n)$$

$$c = \operatorname{store}_{\mathcal{K}}(S)$$

$$K' = (H(k_1), \dots, H(k_i), \dots, H(k_n))$$

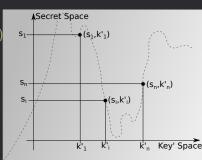
$$c = \mathsf{LagrangeInterpPol}_{\mathcal{K}'}(S)$$

$$c=\sum_{z=0}^t a_z x^z$$

$$s_i = \text{retrieve}_{k_i}(c)$$

$$k_i' = H(k_i)$$

$$s_i = f(k_i^l) = \sum_{j=0}^t a_z k_i^{lz}$$



$$c = \sum_{z=0}^{t} a_z x^z$$

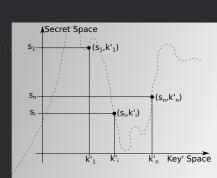
$$s_i = \text{retrieve}_{k_i}(c)$$

$$k_i' = H(k_i)$$

$$k'_{i} = H(k_{i})$$

$$k'_{i} = H(k_{i})$$

$$s_{i} = f(k'_{i}) = \sum_{z=0}^{t} a_{z}k'_{i}^{z}$$



Sometimes  $\mathbb{F}_p$  is really small... Let's say 1000|K|

#### Problem

How can collisions of k' in small  $\mathbb{F}_p$  be omitted

#### Solution:

Use a collision free mapping function per storage  $H_K: K \hookrightarrow K' \subseteq \mathbb{R}$  where |K| = |K'|

Sometimes  $\mathbb{F}_p$  is really small... Let's say 1000|K|

#### Problem:

How can collisions of k' in small  $\mathbb{F}_p$  be omitted

#### Solution

Use a collision free mapping function per storage  $H_K: K \mapsto K' \in \mathbb{F}_p$  where |K| = |K'|

Sometimes  $\mathbb{F}_p$  is really small... Let's say 1000|K|

#### Problem:

How can collisions of k' in small  $\mathbb{F}_p$  be omitted

# Solution:

Use a collision free mapping function per storage

$$H_K: K \mapsto K' \in \mathbb{F}_p$$
 where  $|K| = |K'|$ 

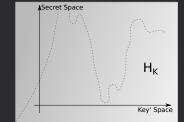
$$S = (s_1, \ldots, s_i, \ldots, s_n)$$
  
 $K = (k_1, \ldots, k_i, \ldots, k_n)$ 

$$c = \operatorname{store}_K(S)$$

$$\Lambda = (\Pi_K(\kappa_1), \dots, \Pi_K(\kappa_i), \dots$$

$$c = \text{LagrangeInterpPol}_{K'}(3), H$$

$$c = \sum_{z=0}^{\infty} a_z x^z, H_k$$



#### Considerations

$$S = (s_1, \ldots, s_i, \ldots, s_n)$$

$$K = (k_1, \ldots, k_i, \ldots, k_n)$$

$$c = \operatorname{store}_K(S)$$

$$H_K: K \mapsto K' \in \mathbb{F}_p[|K| = |K'|]$$

$$K' = (H_K(k_1), \dots, H_K(k_i), \dots, H_K(k_n))$$

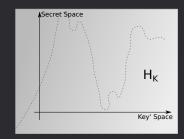
 $c = \text{LagrangeInterpPol}_{K'}(S), H_K$ 

$$c = \sum_{z=0} a_z x^z, H_K$$

$$s_i = \text{retrieve}_{k_i}(c)$$

$$k_i' = H_K(k_i)$$

$$s_i = f(k_i') = \sum_{z=0}^t a_z k_i'^z$$

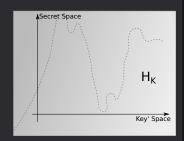


#### Considerations

$$\begin{split} S &= (s_1, \dots, s_i, \dots, s_n) \\ K &= (k_1, \dots, k_i, \dots, k_n) \\ c &= \mathsf{store}_K(S) \\ H_K : K &\mapsto K' \in \mathbb{F}_p ||K| = |K'| \\ K' &= (H_K(k_1), \dots, H_K(k_i), \dots, H_K(k_n)) \\ c &= \mathsf{LagrangeInterpPol}_{K'}(S), H_K \\ c &= \sum_{z=0}^t a_z x^z, \mathsf{H}_K \end{split}$$

$$s_i = \text{retrieve}_{k_i}(c)$$
  
 $k'_i = H_K(k_i)$ 

$$s_i = f(k_i') = \sum_{z=0}^{t} a_z k_i'^z$$



#### Considerations

$$K = (k_1, \ldots, k_n)$$

$$H_K: K \mapsto K' \in \mathbb{F}_p[|K| = |K'|]$$

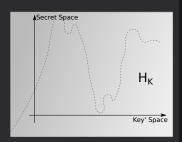
$$c = \text{LagrangeInterpPol}_{K'}(S), H_{\ell}$$

$$c = \sum_{z=0}^{t} a_z x^z, H_K$$

$$s_i = \mathsf{retrieve}_{k_i}(c)$$

$$k_i' = H_K(k_i)$$

$$s_i = \text{retrieve}_{k_i}(c)$$
  
 $k'_i = H_K(k_i)$   
 $s_i = f(k'_i) = \sum_{z=0}^t a_z k'_i^z$ 



Problem Domain Secret Storing Properties What About the E-Voter Missing If in Doubt Introduction

# How to Use It For the 20-Credentials

# A simple example (it works, but...)

924f61661a3472da74387a35f2c8d22e87e84a4d O cbf019b764b9477080c5a9a748a2911a5fa6d614 O fc8ccd6641d45et2etdd926c3a6t7t3ac268e9e3 a29965fbb2954c8a66d856e8eb891bce5f49dacf 3a718d2a84f856bc4e1c8bbb93ca517893c48691 ela5b5d17e51d56f8d6fc868968ff238afba9b32 cbf819b764b9477880caa9a748a2911a5fa6d614 a29965fbb2954c8a66d8b6e8eb891bce5f49dacf 22eb602811c37e6611e85e7a432a45c8f3525749 f5abae583297649847c13be2c54bcbfb3260f0f3

- 924f61661a34d2da74387a35f2c8d22e87e84a4d h1aa98ad3a82ffe896c49687388d8644f58fdd88 2789a4eb84e43e4d5b60d87b3d1edec02d4c449e 2789a4eb84e43e4d5b68d87d3d1edec82d4c449e
- fc8ccd6641d45ef2efdd946c3a6f7f3ac268e9e3 elaSbSd17e51d56f0d6fc868e68ff238afba9b32 fd8b823d965947fc7d9f470907ca18ed68243557 f5abae503297649547c13be2c54bcbfb3268f8f3 3a719d2a84f856bcce1c9bbb93ca517893c48691 blaa98ad3a02ffe896c49687300d8644f50fdd88

22eb602841c37e6611e85e7a432a45c8f3525749 fd8b823d985947fc7d9f470907ca18ed68243557



# How to Use It For the 20-Credentials

A simple example (it works, but...)

- ...use consonants as keys for the 'Drop' credentials
- ...use vowels as keys for 'Accept' credentials
- ...use some other keys for some credentials
- ...store them in the secret-storage system

924f61661a3472da74307a35f2c8d22e07e84a4d O cbf819b764b9477888c5a9a748a2911a5fa6d614 O a29965fbb2954c8a66d856e8eb891bce5f49dacf d 3a718d2a84f856br4e1r8bbb93ca517893c48691 e1a5b5d17e51d56f0d6fc868968ff238afba9b32 h cbf019b764b9477080caa9a748a2911a5fa6d614 ġ a29965fbb2954c8a66d8b6e8eb891bce5f49dacf 22eb602811c37e6611e85e7a432a45c8f3525749 924f61661a34d2da74307a35f2c8d22e07e84a4d 15abae503297649047c13be2c54bcbfb3260f0f3 hlaa98ad3a82ffe896c49687398d8644f58fdd88 2789a4eb84e43e4d5b69d87b3d1edec92d4c449e m 2789a4eb84e43e4d5b60d87d3d1edec02d4c449e n fc8ccd6641d45ef2efdd946c3a6f7f3ac268e9e3 p ela5b5d17e51d56f8d6fc060e68ff238afba9b32 a fd8b823d965947fc7d9f470907ca18ed60243557 f5abae583297649547c13be2c54bcbfb3268f8f3 3a718d2a84f856bcce1c0bbb93ca517893c48691 blaa98ad3a02ffe896c49687308d8644f58fdd88 22eb602841c37e6611e85e7a432a45c8f3525749 S fd8h873d985947fc7d9f478987ca18ed68743557

#### How to Use It For the 20-Credentials

A simple example (it works, but...)

- ...use consonants as keys for the 'Drop' credentials
- ...use vowels as keys for 'Accept' credentials
- ...use some other keys for some credentials
- ...store them in the secret-storage system

chf919h764b9477989c5a9a748a2911a5fa6d614 fc0ccd6641d4Sef2efdd926c3a6f7f3ac268e9e3 a29965fbb2954c8a66d856e8eb891bce5f49dacf 3a718d2a84f856bc4e1c8bbb93ca517893c48691 elaShSd17eSldS6f8d6fc868968ff238afha9h32 cbf819b764b9477088caa9a748a2911a5fa6d614 ġ k a29965fbb2954c8a66d8b6e8eb891bce5f49dacf 22eb602811c37e6611e85e7a432a45c8f3525749 924f61661a34d2da74387a35f2c8d22e87e84a4d f5abae583297649847c13be2c54bcbfb3269f8f3 u blaa98ad3a82ffe896c4968738bd8644f50fdd88 2789x4+b84e43e4d5b69d87b3d1edec82d4c449e 2789a4ab84a43a445b684874341a4ac8244c449a n fc8ccd6641d45ef2efdd946c3a6f7f3ac268e9e3 p e1aSb5d17e51d56f8d6fc868e68ff238afba9b32 ģ fd8b823d965947fc7d9f470907ca18ed68243557 f5abae503297649547c13be2c54bcbfb3268f0f3 \square

3a71847a84f856hcce1c6hhh93ca517893c48691

blaa98ad3a92ffe896c49687399d8644f59fdd88 22eb692841c37e6611e85e7a432a45c8f3525749 fd8b823d985947fc7d9f470997ca18ed69243557

92416166133472437438733512c8422e87e84344

a

S

## How to Use It For the 20-Credentials

A simple example (it works, but...)

- ...use consonants as keys for the 'Drop' credentials
- ...use vowels as keys for 'Accept' credentials
- ...use some other keys for some credentials
- ...store them in the secret-storage system

chf919h764b9477989c5a9a748a2911a5fa6d614 fc0ccd6641d4Sef2efdd926c3a6f7f3ac268e9e3 a29965fbb2954c8a66d856e8eb891bce5f49dacf 3a718d2a84f856bc4e1c8bbb93ca517893c48691 elaShSd17eSldS6f8d6fc868968ff238afha9h32 cbf819b764b9477088caa9a748a2911a5fa6d614 ġ k a29965fbb2954c8a66d8b6e8eb891bce5f49dacf 22eb602811c37e6611e85e7a432a45c8f3525749 924f61661a34d2da74387a35f2c8d22e87e84a4d f5abae503297649647c13be2c54bcbfb3260f0f3 u blaa98ad3a82ffe896c4968738bd8644f50fdd88 2789a4eb84e43e4d5b69d87b3d1edec82d4c449e 2789a4ab84a43a445b684874341a4ac8244c449a fc8ccd6641d45ef2efdd946c3a6f7f3ac268e9e3 p e1aSb5d17e51d56f8d6fc868e68ff238afba9b32 ģ fd8b823d965947fc7d9f470907ca18ed68243557 3a71847a84f856hcce1c6hhh93ca517893c48691 a 0 blas98ad3a92ffe896c49687199d8644f59fdd88

92416166133472437438733512c8422e87e84344

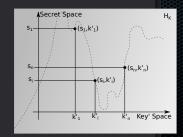
22eb602841c37e6611e85e7a432a45c8f3525749 fd8b823d985947fc7d9f470907ca18ed68243557



### How to Use It For the 20-Credentials

# A simple example (it works, but...)

- 'Drop' credentials
- ...use vowels as keys for 'Accept credentials
- ...use some other keys for some credentials
- ...store them in the secret-storage system



The secret-storing system brings other very desirable features:

Typoo Resistance via Secret-Storing System | The OR-Function

Either  $key_i$  OR  $key_n$  unveils the single secret

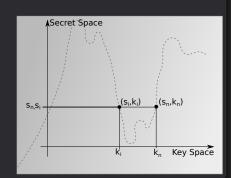
```
s_n = \text{retrieve}_{k_i}(c)
```

 $verifv(s_i = s_n)$ 

The secret-storing system brings other very desirable features:

Typoo Resistance via Secret-Storing System | The OR-Function Either  $key_i$  OR  $key_n$  unveils the single secret

$$s_i = \text{retrieve}_{k_i}(c)$$
  
 $s_n = \text{retrieve}_{k_n}(c)$   
 $verify(s_i = s_n)$ 



# Key-Hinting via Secret-Storing System | The AND Function

Only  $key_i$  AND  $key_n$  unveil the single secret:

- k<sub>a</sub> "HintMeWith...:"
- k<sub>b</sub> "GoodCredentials"
- k<sub>c</sub> "BadCredentials"
- $k_d$   $s_a \oplus s_b$
- $k_e$   $s_a \oplus s_c$

- s<sub>a</sub> "098z71adf4383498"
- *s<sub>b</sub>* "aer9393fads932sv3"
- *s<sub>c</sub>* "598nnja2devm24v3a"
- *s<sub>d</sub>* "vowel"
- se "consonant"

This still is secure as long as the keys carries some entropy! It is definitely more secure than "What is your mother's maiden name" used nowadays!

#### And this one for free:

# Secret-Sharing via Secret-Storing System | The THRESHOLD Function

Only n out of m keys unveil the single secret...

$$k_a$$
 "Chief-1"

$$k_c$$
 "Chief-3"

$$k_x$$
  $s_a \oplus s_b$ 

$$k_v s_a \oplus s_c$$

$$k_z s_b \oplus s_c$$

# Randomized Secret-Storing System

If some data points R not representing not in (K, S) are chosen arbitrary, the system changes from a deterministic secret-storing system into a randomized secrets-storing System, where two stores containing the same (K, S) 'look' completely different.

#### Definition

A randomized secret-storing system of order n,

$$\Gamma = (S, K, R, C, randomStore, retrieve),$$

consists of a secret space S, a key space K, a randomization space R, a storage space C, and two functions randomStore and retrieve with properties as introduced above.

# Randomized Secret-Storing System

If some data points R not representing not in (K, S) are chosen arbitrary, the system changes from a deterministic secret-storing system into a randomized secrets-storing System, where two stores containing the same (K, S) 'look' completely different.

#### Definition

A randomized secret-storing system of order n,

$$\Gamma = (S, K, R, C, randomStore, retrieve),$$

consists of a secret space S, a key space K, a randomization space R, a storage space C, and two functions randomStore and retrieve with properties as introduced above.

# The Thing is a Beast

#### Field-homomorphism

Every operation (+,\*) can be applied using two secret-stores as operands. The same operations will then be applied on all secrets of the same key.

This is exactly the most desired feature in order to work in a secure cloud system on the internet.

A CPU in the cloud can now do calculations using secret-stores, hence not knowing the operands nor the result.

Discussion

This is left as an exercise for the reader...

...One more Thing...

The Asymmetric variant of the secret-storing system with the same properties as shown above.

# Please feel Free to Break the Secret-Storing System!

As it is a rather easy concept (even for a young math-student), it is easy to understand the system and thus to look for the show-stopper.... So, if you ever... Please let me know too!

# Happy Hacking!



# KryptonIT is Not Secure in Theory

The probability p that any  $s_x \notin S$  matches to at least one  $k_x \in K$  is equal to 0.

$$p(s_x \notin S \text{ matches to at least one } k_x \notin K) = \frac{1}{|S|}$$
.

$$p(s_x \notin S \text{ matches to no } k_x \notin K) = 1 - \frac{1}{|S|}.$$

$$p(|\mathcal{S}|s_x \notin S \text{ do not match any } k_x \notin K) = (1 - \frac{1}{|\mathcal{S}|})^{\mathcal{S}}.$$

So: 
$$\lim_{n\to\infty} (1-\frac{1}{S})^n = \frac{1}{e}$$

This leads to the following entropy equation:

$$H(s_{\scriptscriptstyle X}|c) = H(s_{\scriptscriptstyle X}) - \frac{1}{e}$$

### KryptonIT is Not Secure in Theory

The minuend of  $H(s_x|c) = H(s_x) - \frac{1}{e}$  can be reduced if the mapping-space not not equal to the secret-space.

So K would not map to S directly but to M whereas M is a multiple  $\phi$  of S.

A sub-space-mapping (modulo) then maps from M to S.

This leads to the following entropy equation:

$$H(s_{\scriptscriptstyle X}|c) = H(s_{\scriptscriptstyle X}) - (rac{1}{e})^{rac{1}{\phi}}$$

And...with  $\lim_{\phi \to \infty}$ 

$$H(s_{\scriptscriptstyle X}|c) = H(s_{\scriptscriptstyle X})$$

#### KryptonIT is Secure in Practical Usage

As long as  $S \times \phi$  is chosen big enough (>200bit+(1bit per anno)) the system is considered to be secure in practice.

This is true as the analyses require to create a 'rainbow-table' for each c. This is considered infeasible for c spanning a big space.