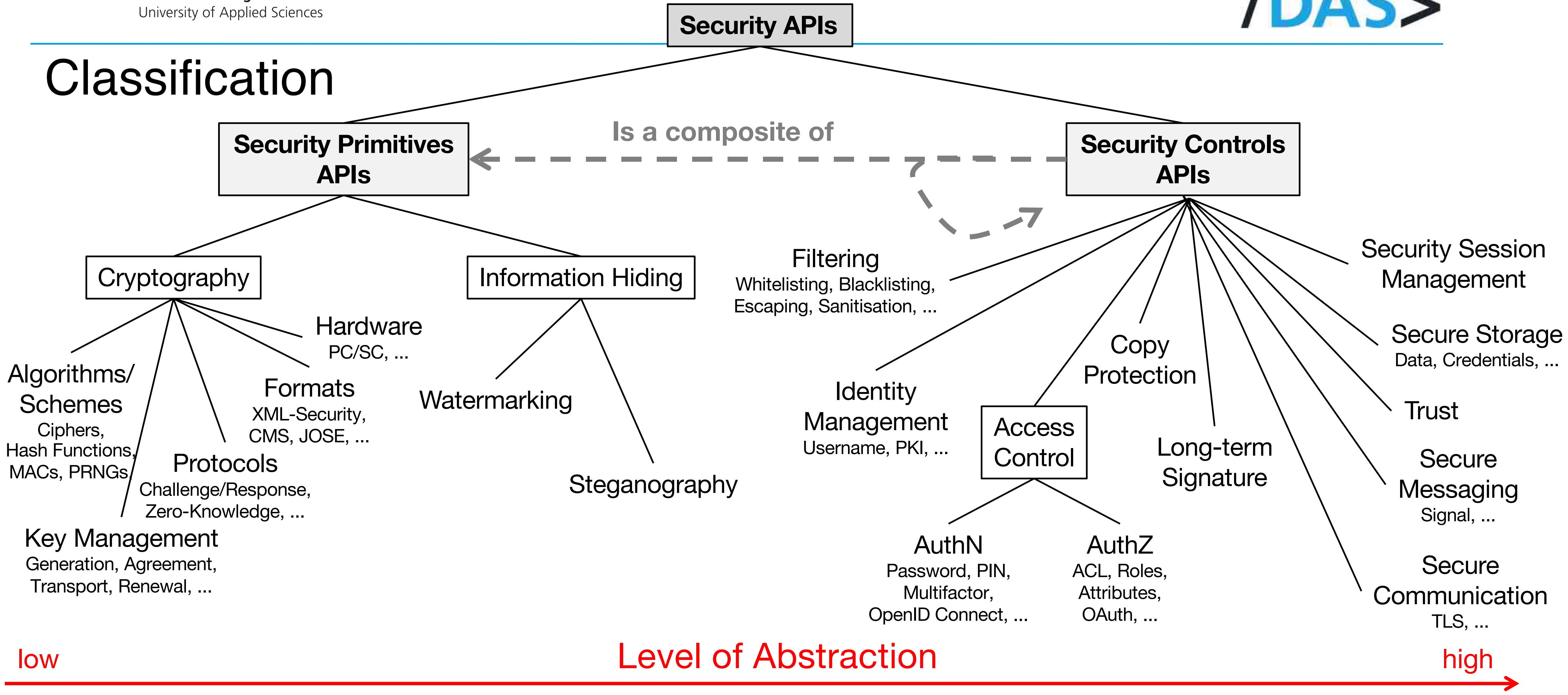
Classification



Basic Security Services
Confidentiality, Integrity, Authenticity, Non-repudiation

Concrete Security Measures
User Login, Secure File Transfer

Quelle: L. Lo Iacono und P. L. Gorski (2017). „I Do and I Understand. Not Yet True for Security APIs. So Sad." In: The 2nd European Workshop on Usable Security, EuroUSEC ´17, Paris, France.

Incorrect Use of Cryptographic APIs

Hard-to-use cryptographic APIs increase the likelihood of software vulnerabilities.

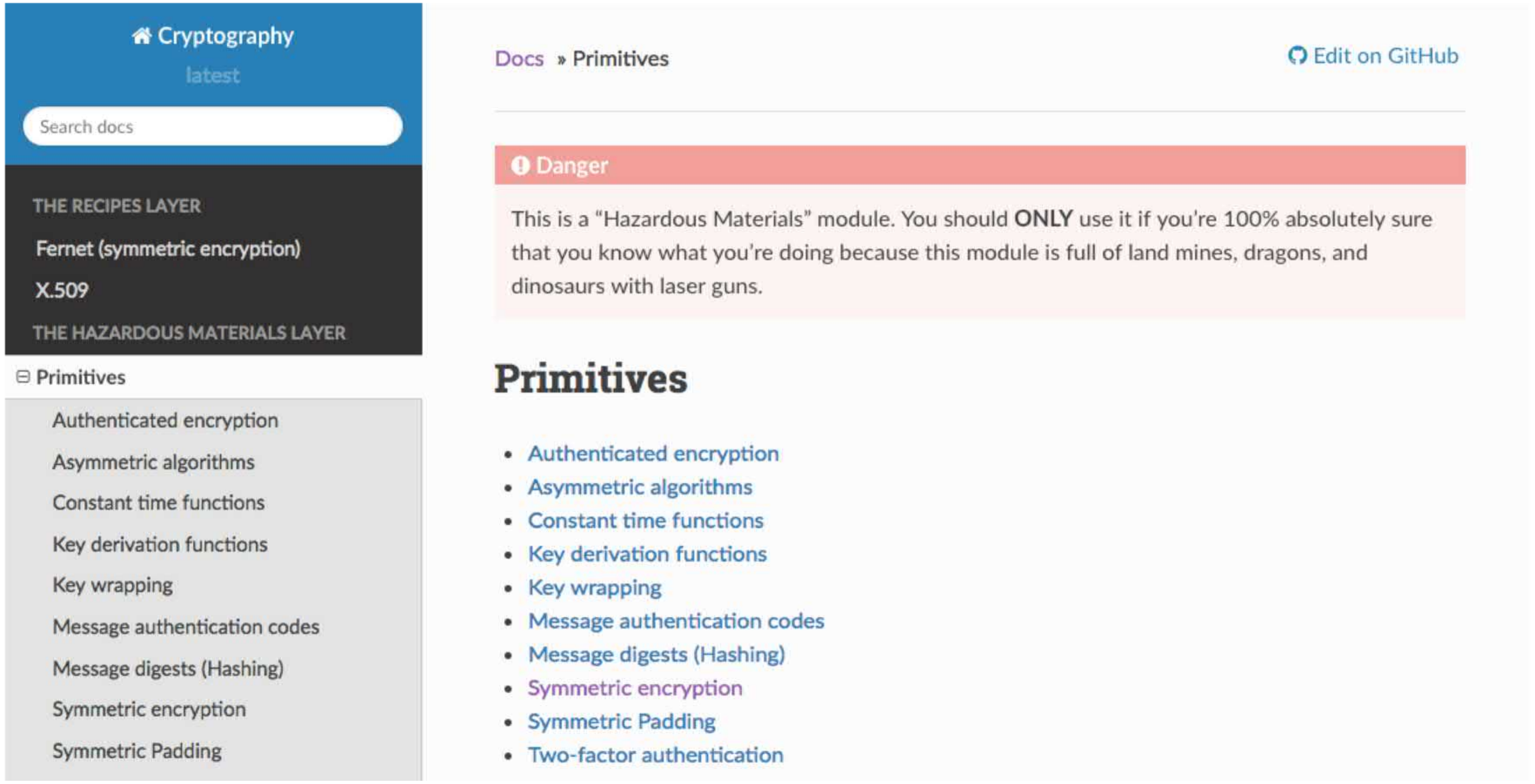
Y. Acar, M. Backes, S. Fahl, S. Garnkel, D. Kim, M. L. Mazurek, and C. Stransky.
Comparing the usability of cryptographic APIs.
In 2017 IEEE Symposium on Security and Privacy (SP), pages 154 - 171, 2017.

A. Naiakshina, A. Danilova, C. Tiefenau, M. Herzog, S. Dechand, and M. Smith.
Why do developers get password storage wrong?: A qualitative usability study.
In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS '17. ACM, 2017.

S. Nadi, S. Kruger, M. Mezini, and E. Bodden.
Jumping Through Hoops": Why do Java Developers Struggle With Cryptography APIs?
In Proceedings of the 37th International Conference on Software Engineering (ICSE 2016), 2016.

Designed by iconicbestiary / Freepik

Cryptography.io Documentation



Danger

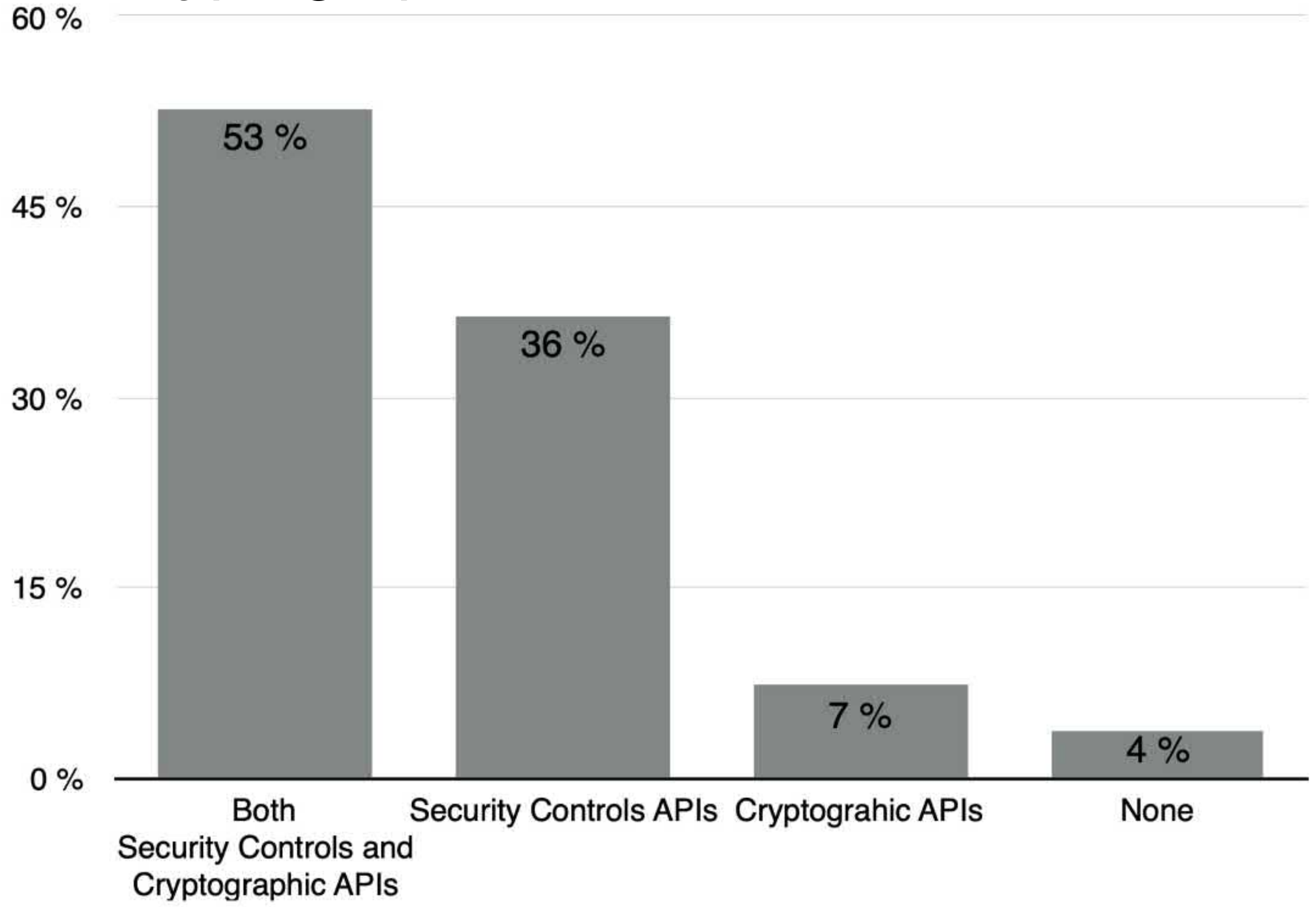
This is a "Hazardous Materials" module. You should **ONLY** use it if you're 100% absolutely sure that you know what you're doing because this module is full of land mines, dragons, and dinosaurs with laser guns.

Primitives

- Authenticated encryption
- Asymmetric algorithms
- Constant time functions
- Key derivation functions
- Key wrapping
- Message authentication codes
- Message digests (Hashing)
- Symmetric encryption
- Symmetric Padding
- Two-factor authentication

Source: https://cryptography.io/en/latest/hazmat/primitives/

Security APIs vs Cryptographic APIs



Source: L. Lo Iacono und P. L. Gorski (2017). „I Do and I Understand. Not Yet True for Security APIs. So Sad.“ In: The 2nd European Workshop on Usable Security, EuroUSEC ´17, Paris, France.

PyCrypto Documentation

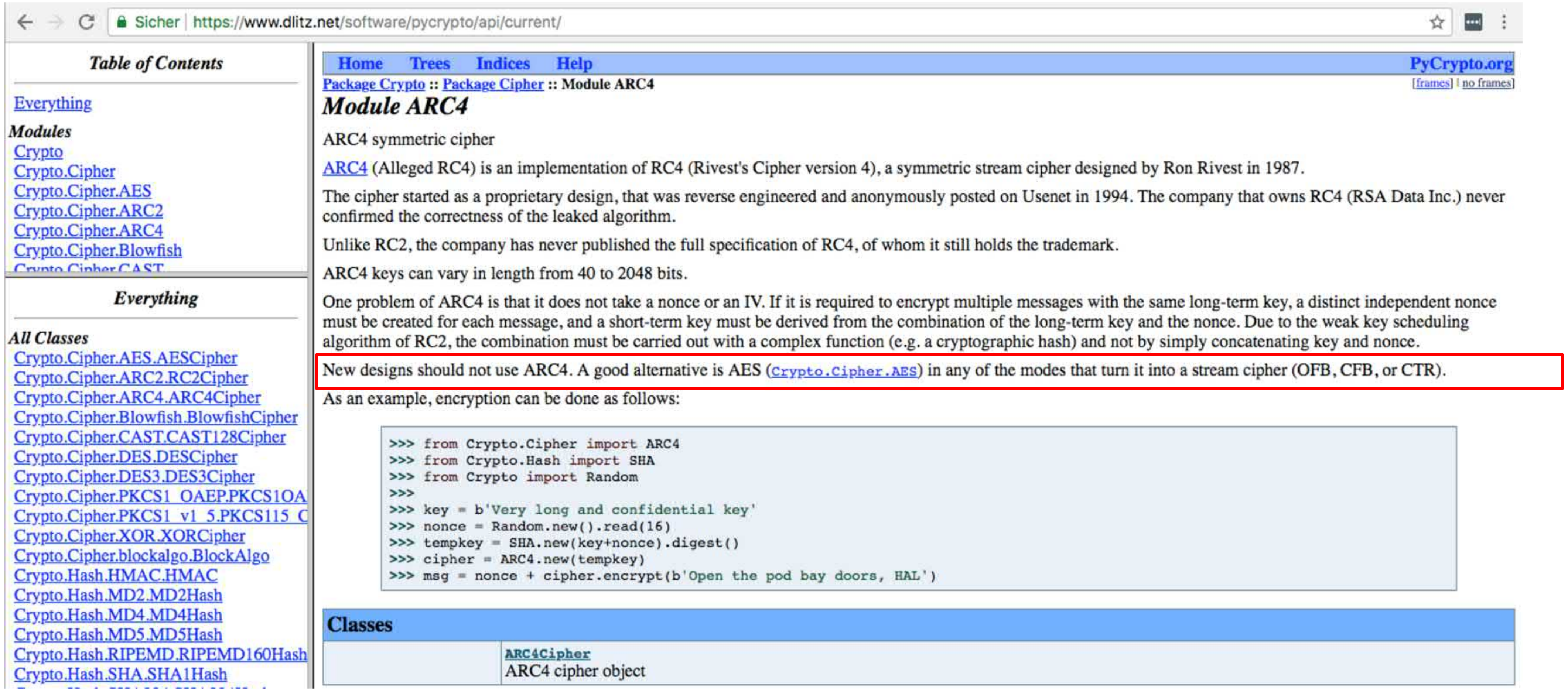


Table of Contents

Everything

Modules
Crypto
Crypto.Cipher
Crypto.Cipher.AES
Crypto.Cipher.ARC2
Crypto.Cipher.ARC4
Crypto.Cipher.Blowfish
Crypto.Cipher.CAST

Everything

All Classes
Crypto.Cipher.AES.AESCipher
Crypto.Cipher.ARC2.RC2Cipher
Crypto.Cipher.ARC4.ARC4Cipher
Crypto.Cipher.Blowfish.BlowfishCipher
Crypto.Cipher.CAST.CAST128Cipher
Crypto.Cipher.DES.DESCipher
Crypto.Cipher.DES3.DES3Cipher
Crypto.Cipher.PKCS1_OAEP.PKCS1OA
Crypto.Cipher.PKCS1_v1_5.PKCS115_C
Crypto.Cipher.XOR.XORCipher
Crypto.Cipher.blockalgo.BlockAlgo
Crypto.Hash.HMAC.HMAC
Crypto.Hash.MD2.MD2Hash
Crypto.Hash.MD4.MD4Hash
Crypto.Hash.MD5.MD5Hash
Crypto.Hash.RIPEMD.RIPEMD160Hash
Crypto.Hash.SHA.SHA1Hash

Home Trees Indices Help PyCrypto.org

Package Crypto :: Package Cipher :: Module ARC4 [frames] | no frames]

Module ARC4

ARC4 symmetric cipher

ARC4 (Alleged RC4) is an implementation of RC4 (Rivest's Cipher version 4), a symmetric stream cipher designed by Ron Rivest in 1987.

The cipher started as a proprietary design, that was reverse engineered and anonymously posted on Usenet in 1994. The company that owns RC4 (RSA Data Inc.) never confirmed the correctness of the leaked algorithm.

Unlike RC2, the company has never published the full specification of RC4, of whom it still holds the trademark.

ARC4 keys can vary in length from 40 to 2048 bits.

One problem of ARC4 is that it does not take a nonce or an IV. If it is required to encrypt multiple messages with the same long-term key, a distinct independent nonce must be created for each message, and a short-term key must be derived from the combination of the long-term key and the nonce. Due to the weak key scheduling algorithm of RC2, the combination must be carried out with a complex function (e.g. a cryptographic hash) and not by simply concatenating key and nonce.

New designs should not use ARC4. A good alternative is AES (`Crypto.Cipher.AES`) in any of the modes that turn it into a stream cipher (OFB, CFB, or CTR).

As an example, encryption can be done as follows:

```
>>> from Crypto.Cipher import ARC4
>>> from Crypto.Hash import SHA
>>> from Crypto import Random
>>>
>>> key = b'Very long and confidential key'
>>> nonce = Random.new().read(16)
>>> tempkey = SHA.new(key+nonce).digest()
>>> cipher = ARC4.new(tempkey)
>>> msg = nonce + cipher.encrypt(b'Open the pod bay doors, HAL')
```

Classes	
	`ARC4Cipher` ARC4 cipher object

Source: https://www.dlitz.net/software/pycrypto/api/current/Crypto.Cipher.ARC4-module.html

Incorrect Use of Cryptographic APIs

- ”Real-world Android developers use Stack Overflow (and other Q&A communities) as a major resource for solving programming problems, including security- and privacy relevant problems”

- **Important Factor:** Degree of instantly understandable and actionable support (e.g. instructions, guidelines)

Designed by iconicbestiary / Freepik

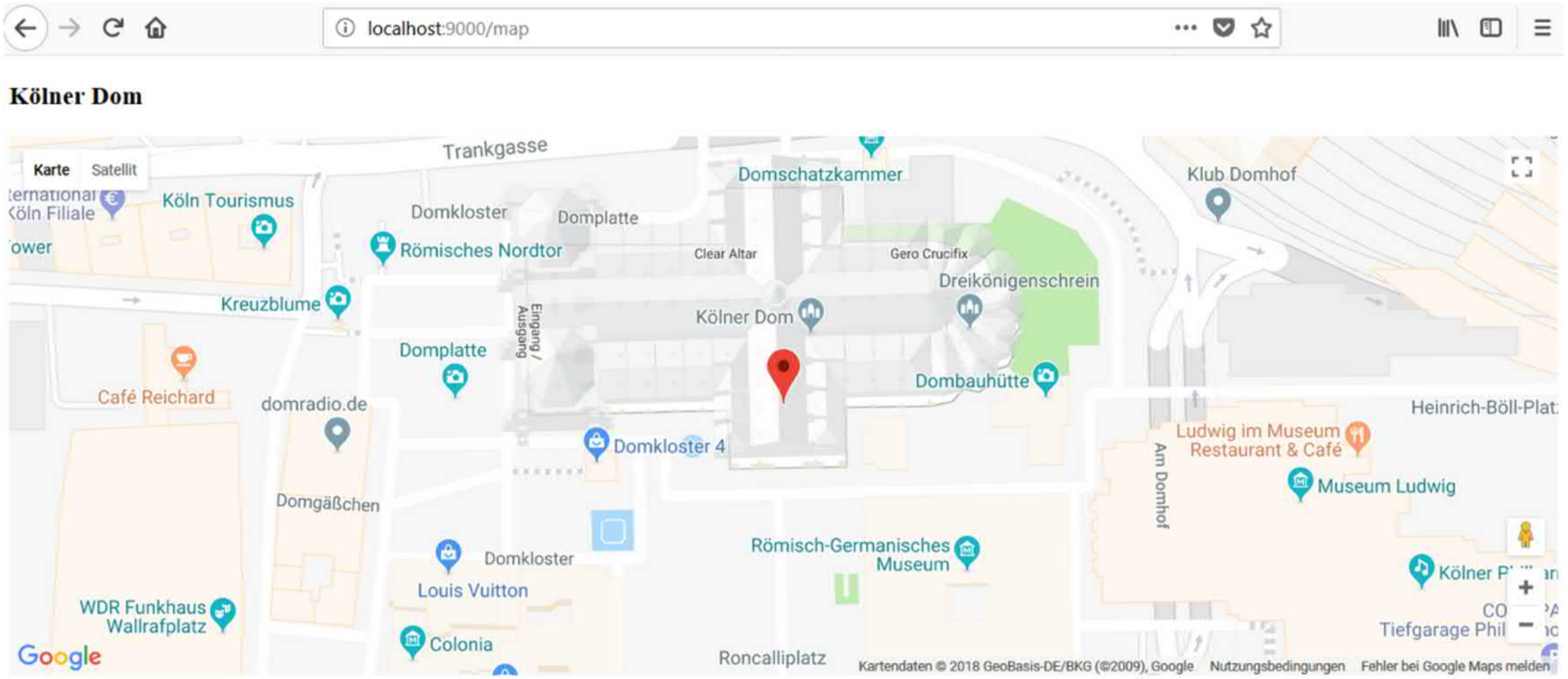
Y. Acar, M. Backes, S. Fahl, D. Kim, M. L. Mazurek, and C. Stransky.
You get where you're looking for: The impact of information sources on code security.
In 2016 IEEE Symposium on Security and Privacy (SP), pages 289 - 305, May 2016.

Why do developers make mistakes when storing passwords?

- Developers think about functionality first and security second.
- Requiring security can make a difference.
- Standards and recommendations are important.
- Opt-out rather than opt-in security.

Source: A. Naiakshina, A. Danilova, C. Tiefenau, M. Herzog, S. Dechand, and M. Smith: Why Do Developers Get Password Storage Wrong? A Qualitative Usability Study, ACM CCS 2017

Programming Task

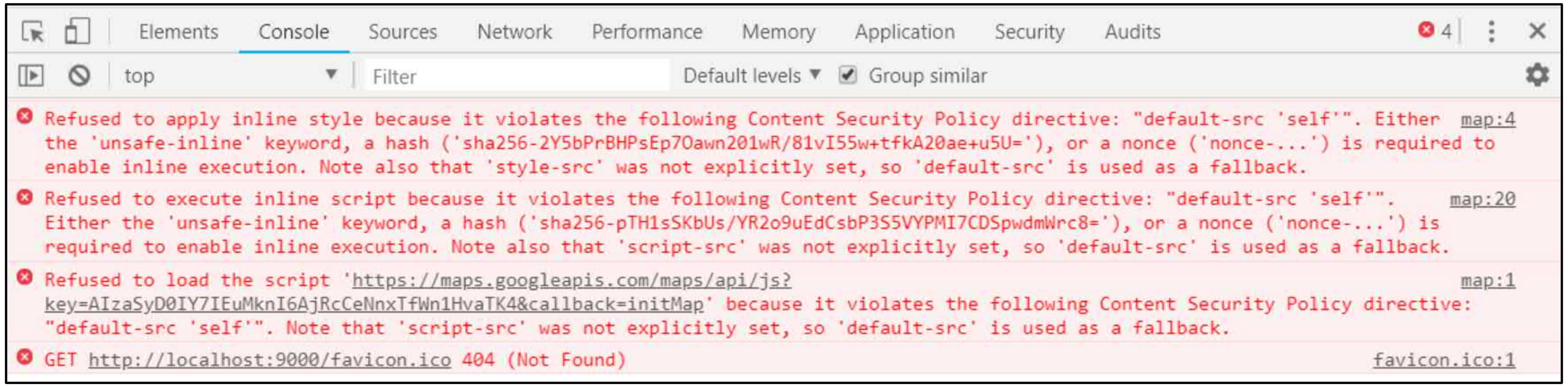

localhost:9000/map

Kölner Dom

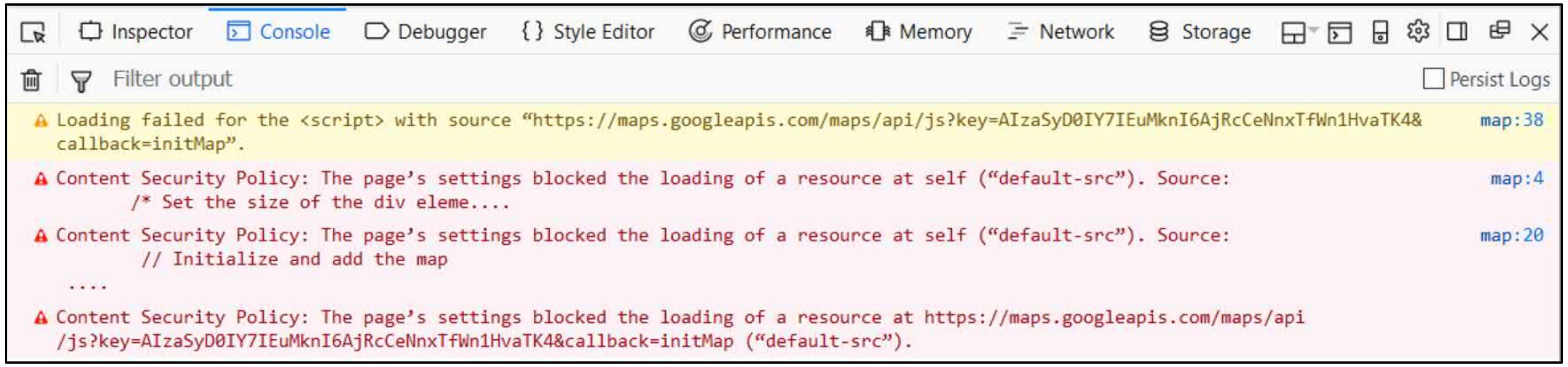
Bildquelle: https://www.google.de/maps

Source: Peter Leo Gorski, Luigi Lo Iacono, Stephan Wiefling und Sebastian Möller,

CSP Violation Messages / Warnings



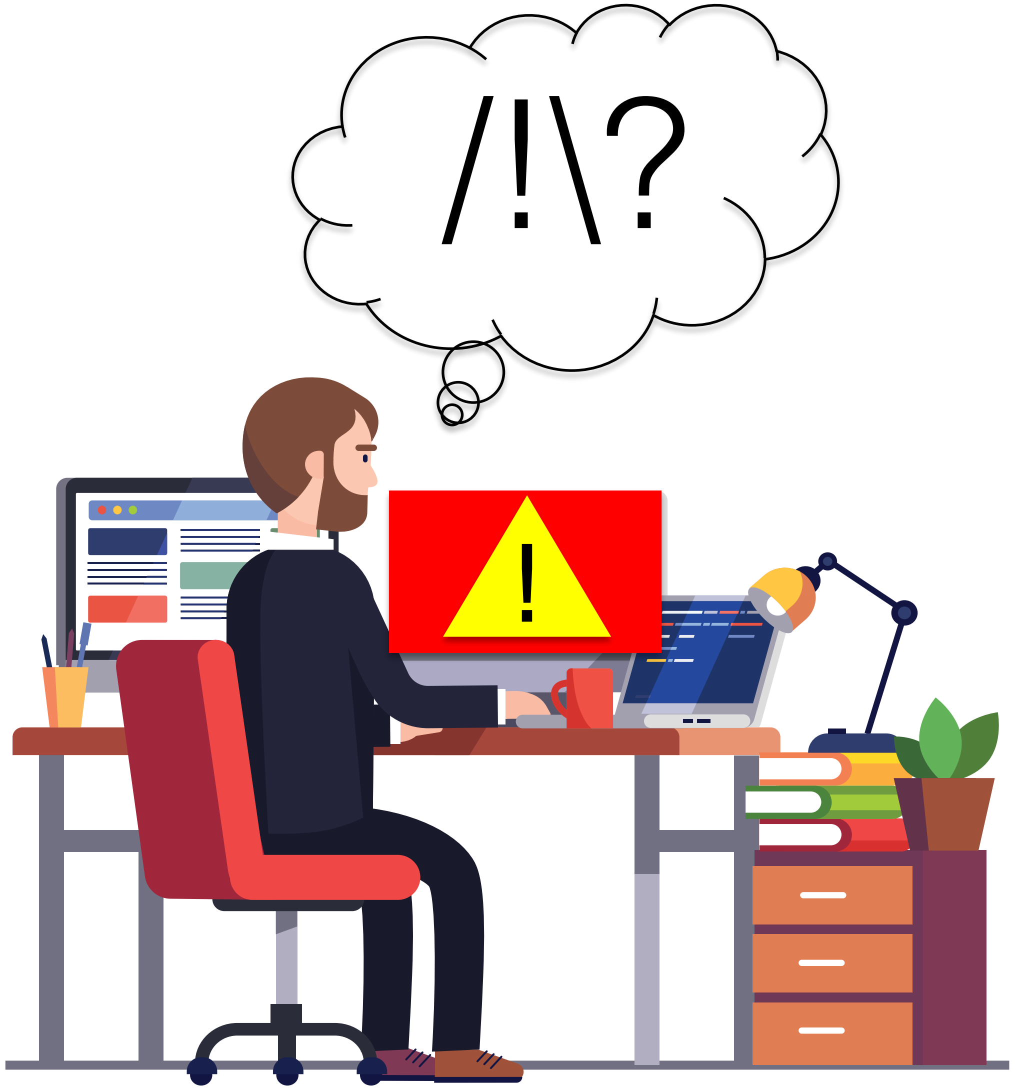
```
Refused to apply inline style because it violates the following Content Security Policy directive: "default-src 'self'". Either   map:4
the 'unsafe-inline' keyword, a hash ('sha256-2Y5bPrBHPsEp7Oawn201wR/81vI55w+tfkA20ae+u5U='), or a nonce ('nonce-...') is required to
enable inline execution. Note also that 'style-src' was not explicitly set, so 'default-src' is used as a fallback.
Refused to execute inline script because it violates the following Content Security Policy directive: "default-src 'self'".   map:20
Either the 'unsafe-inline' keyword, a hash ('sha256-pTH1sSKbUs/YR2o9uEdCsbP3S5VYPMI7CDSpwdmWrc8='), or a nonce ('nonce-...') is
required to enable inline execution. Note also that 'script-src' was not explicitly set, so 'default-src' is used as a fallback.
Refused to load the script 'https://maps.googleapis.com/maps/api/js?                                                          map:1
key=AIzaSyD0IY7IEuMknI6AjRcCeNnxTfWn1HvaTK4&callback=initMap' because it violates the following Content Security Policy directive:
"default-src 'self'". Note that 'script-src' was not explicitly set, so 'default-src' is used as a fallback.
GET http://localhost:9000/favicon.ico 404 (Not Found)                                                                 favicon.ico:1
```



```
Loading failed for the <script> with source "https://maps.googleapis.com/maps/api/js?key=AIzaSyD0IY7IEuMknI6AjRcCeNnxTfWn1HvaTK4&   map:38
callback=initMap".
Content Security Policy: The page's settings blocked the loading of a resource at self ("default-src"). Source:                     map:4
        /* Set the size of the div eleme....
Content Security Policy: The page's settings blocked the loading of a resource at self ("default-src"). Source:                     map:20
         // Initialize and add the map
 ....
Content Security Policy: The page's settings blocked the loading of a resource at https://maps.googleapis.com/maps/api
/js?key=AIzaSyD0IY7IEuMknI6AjRcCeNnxTfWn1HvaTK4&callback=initMap ("default-src").
```

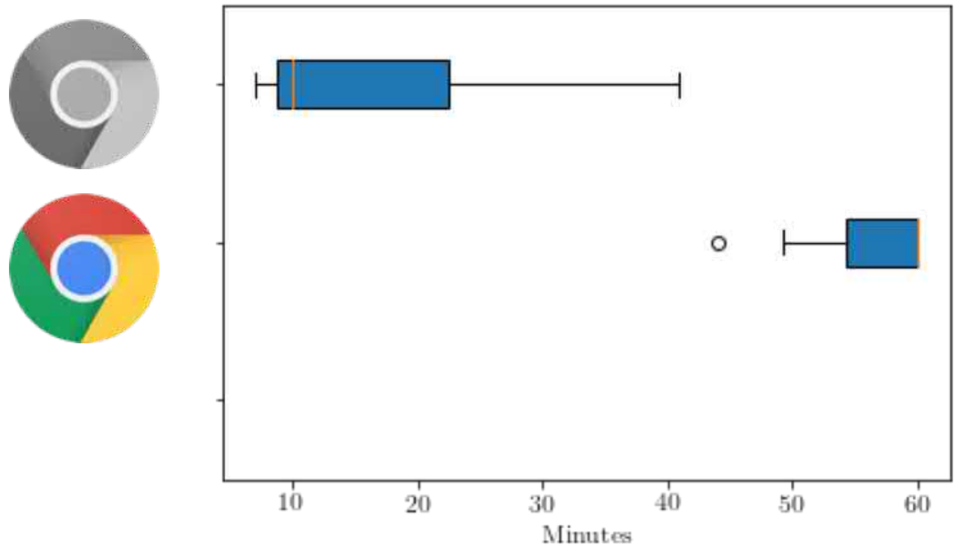
Source: Peter Leo Gorski, Luigi Lo Iacono, Stephan Wiefling und Sebastian Möller,
„Warn if Secure or How to Deal with Security by Default in Software Development?“,
12th International Symposium on Human Aspects of Information Security and Assurance (HAISA), 2018

Developers Deserve Security Warnings, Too



Designed by iconicbestiary / Freepik

Total Time



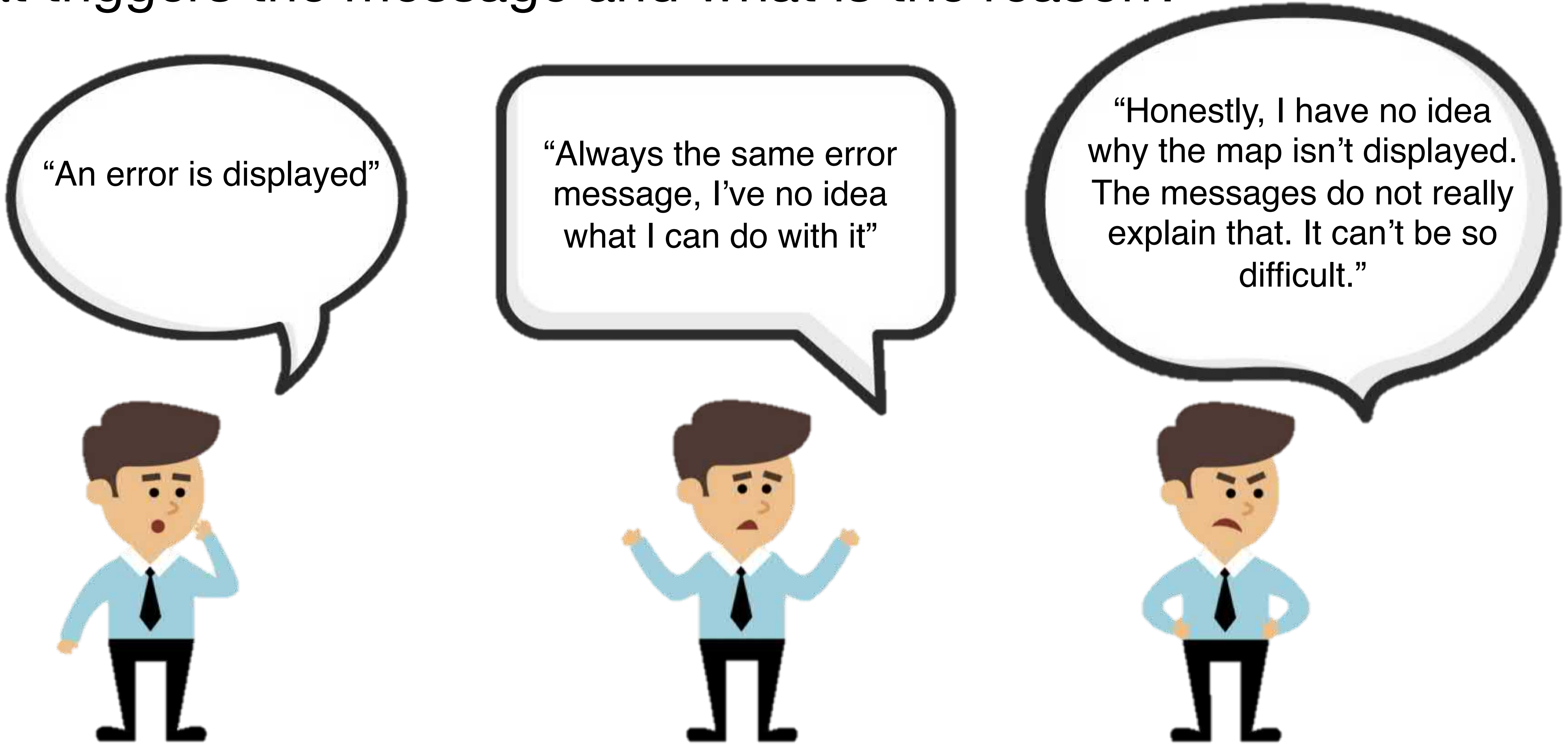
CSP	Browser	Mean	Median	60 min
No	Chrome	16 min	9.9 min	0/10
Yes	Chrome	56.6 min	60 min	7/10

Mann-Whitney U test; $U=0$; $p<0.001$

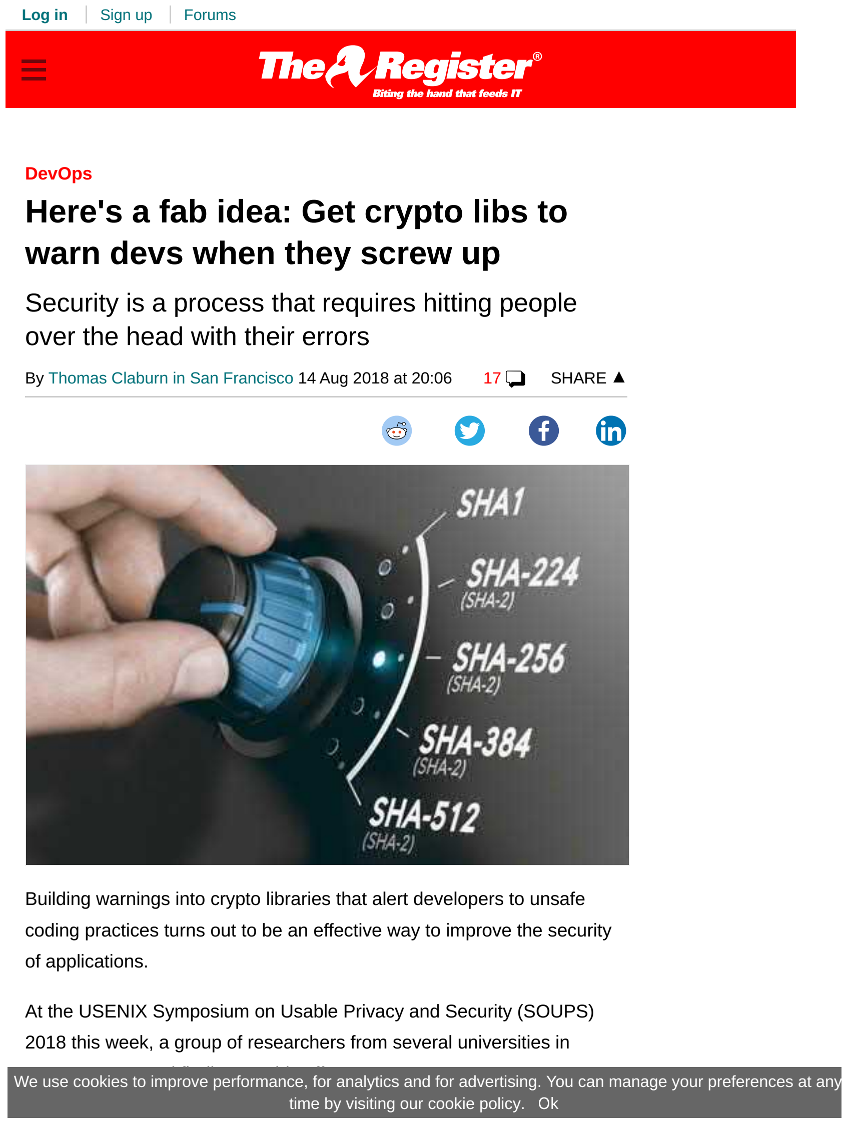
Participants trying to solve the task in the Chrome group being confronted with CSPs needed significantly more time than participants in the Control condition

Source: Peter Leo Gorski, Luigi Lo Iacono, Stephan Wiefling und Sebastian Möller,
„Warn if Secure or How to Deal with Security by Default in Software Development?“,
12th International Symposium on Human Aspects of Information Security and Assurance (HAISA), 2018

What triggers the message and what is the reason?


"An error is displayed"
"Always the same error message, I've no idea what I can do with it"
"Honestly, I have no idea why the map isn't displayed. The messages do not really explain that. It can't be so difficult."

Source: Peter Leo Gorski, Luigi Lo Iacono, Stephan Wiefling und Sebastian Möller,
„Warn if Secure or How to Deal with Security by Default in Software Development?",
12th International Symposium on Human Aspects of Information Security and Assurance (HAISA), 2018



Log in | Sign up | Forums

The Register®
Biting the hand that feeds IT

DevOps

Here's a fab idea: Get crypto libs to warn devs when they screw up

Security is a process that requires hitting people over the head with their errors

By Thomas Claburn in San Francisco 14 Aug 2018 at 20:06 17 SHARE ▲

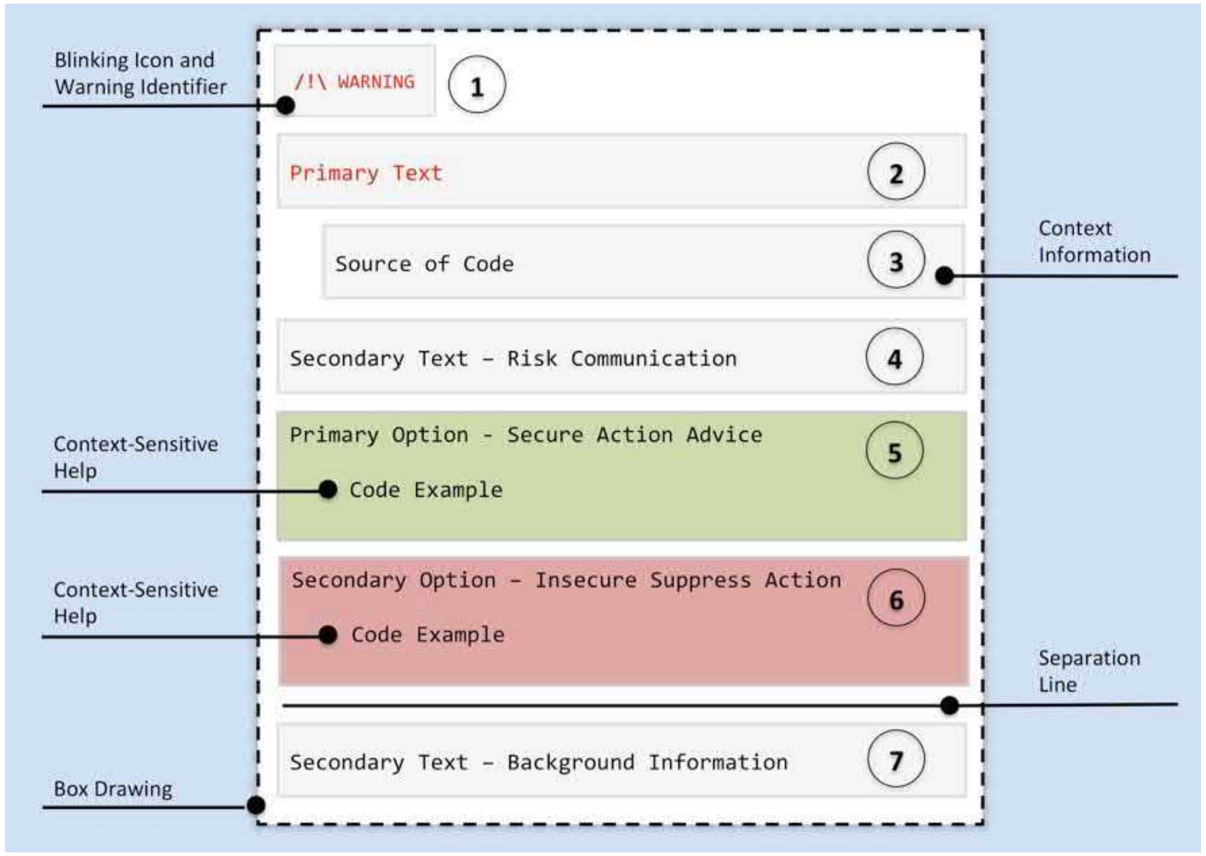
Building warnings into crypto libraries that alert developers to unsafe coding practices turns out to be an effective way to improve the security of applications.

At the USENIX Symposium on Usable Privacy and Security (SOUPS) 2018 this week, a group of researchers from several universities in

We use cookies to improve performance, for analytics and for advertising. You can manage your preferences at any time by visiting our cookie policy. Ok

Source: The Register
https://www.theregister.co.uk/2018/08/14/developers_crypto_training/

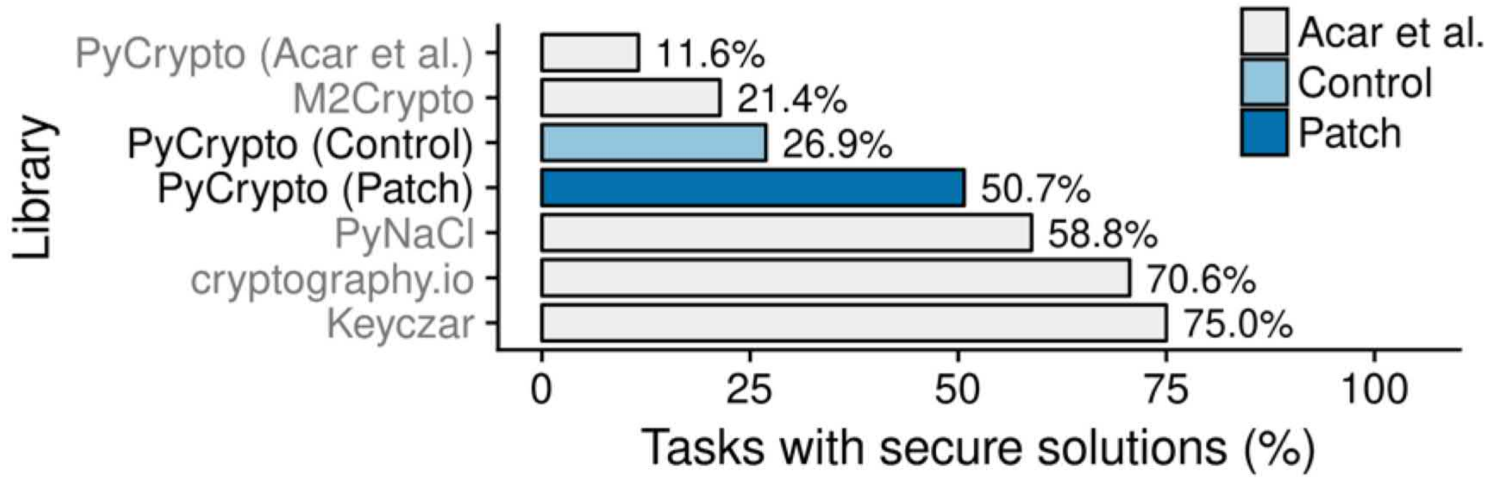
Integrated Design Approach for Security Recommendations


Blinking Icon and Warning Identifier
/!\ WARNING
1
Primary Text
2
Source of Code
3
Context Information
Secondary Text – Risk Communication
4
Primary Option - Secure Action Advice
5
Context-Sensitive Help
Code Example
Secondary Option – Insecure Suppress Action
6
Context-Sensitive Help
Code Example
Separation Line
Secondary Text – Background Information
7
Box Drawing

Source: Peter Leo Gorski, Luigi Lo Iacono, Dominik Wermke, Christian Stransky, Sebastian Möller, Yasemin Acar and Sascha Fahl „Developers Deserve Security Warnings, Too: On the Effect of Integrated Security Advice on Cryptographic API Misuse“, Fourteenth Symposium on Usable Privacy and Security (SOUPS 2018)

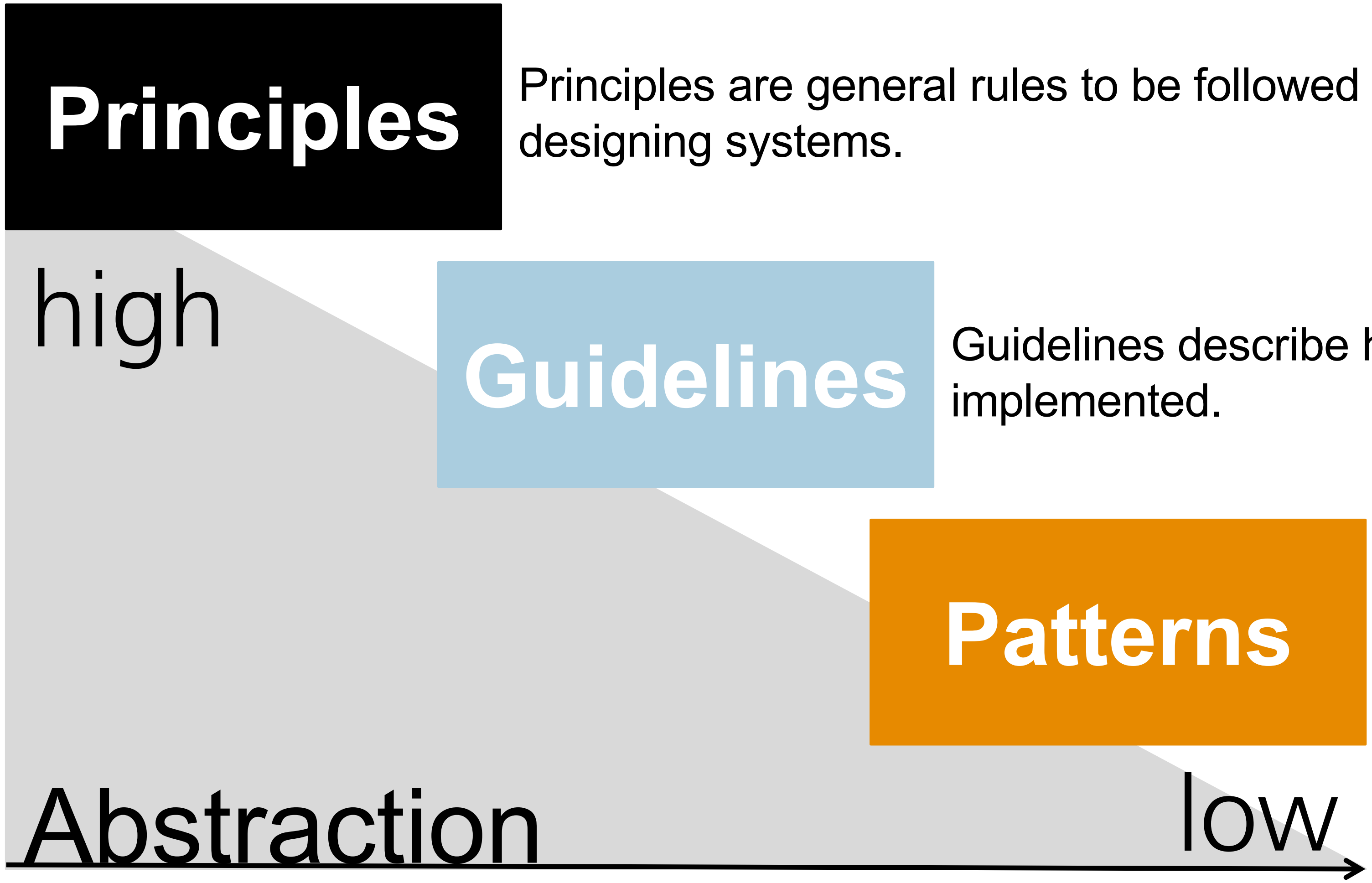
Results for Security

– 26.9% secure solutions in the PyCrypto condition
– 50.7% secure solutions in the PyCrypto patch condition



Source: Peter Leo Gorski, Luigi Lo Iacono, Dominik Wermke, Christian Stransky, Sebastian Möller, Yasemin Acar and Sascha Fahl „Developers Deserve Security Warnings, Too: On the Effect of Integrated Security Advice on Cryptographic API Misuse", Fourteenth Symposium on Usable Privacy and Security (SOUPS 2018)

GUIDANCE



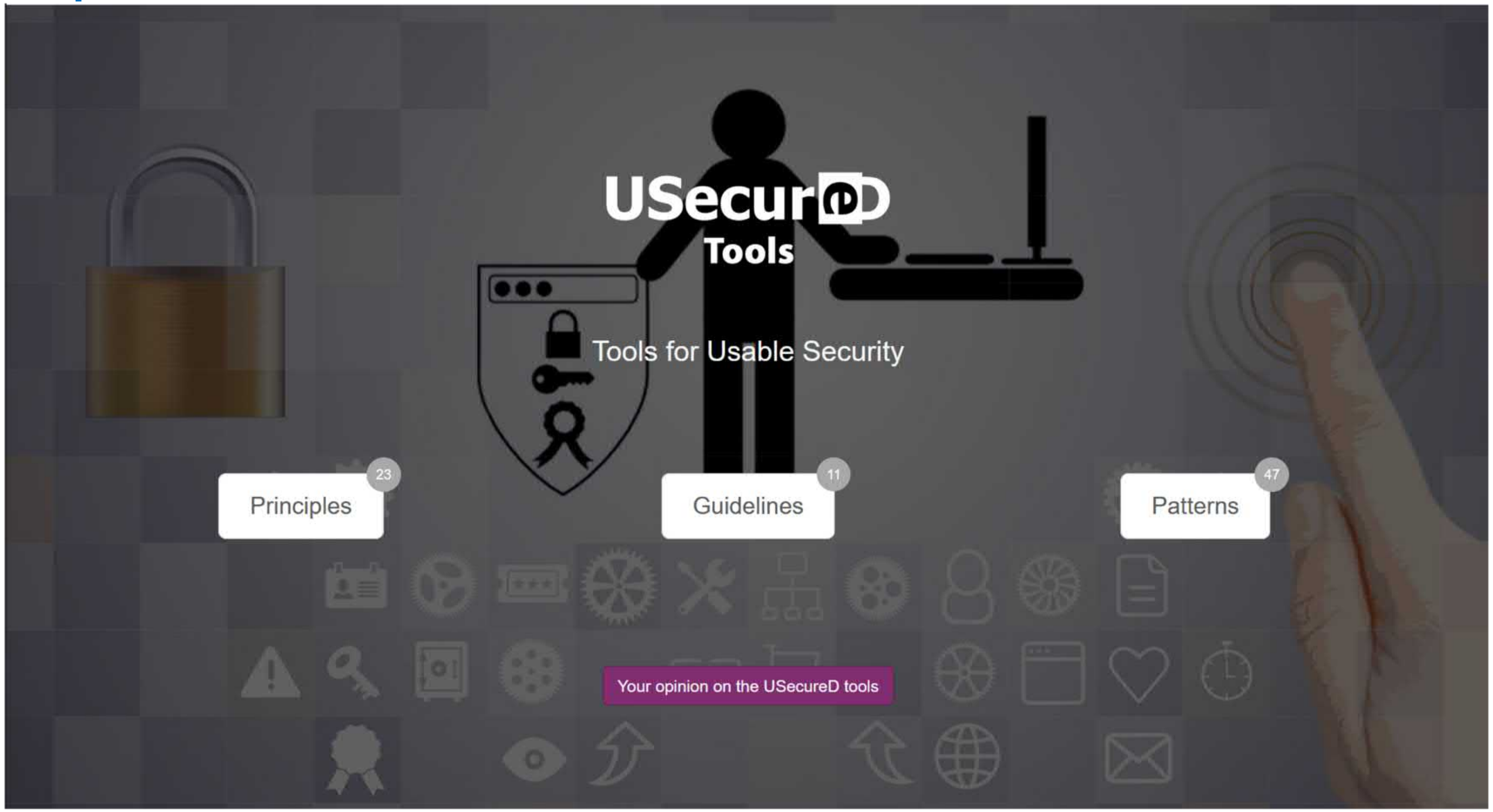
Principles are general rules to be followed when designing systems.

Guidelines describe how principles can be implemented.

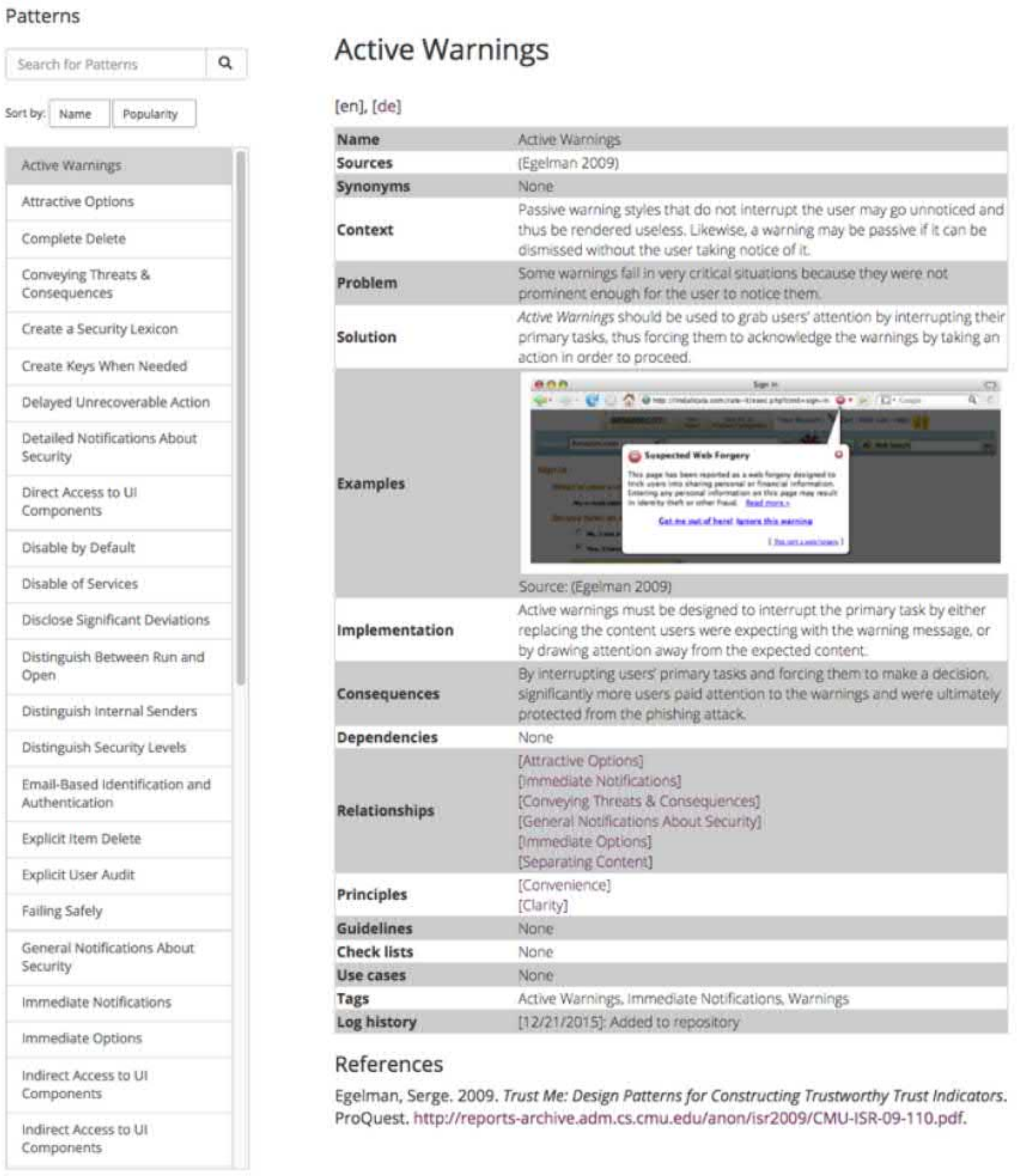
Patterns are proven solutions to recurring problems that occur in system development.

Source: L. Lo Iacono, M. Smith, E. v. Zezschwitz, P. L. Gorski, P. Nehren, “Consolidating Principles and Patterns for Human-centred Usable Security Research and Development” in: The 3rd European Workshop on Usable Security, EuroUSEC ´18, London, England, April 23, 2018.

https://das.h-brs.de/usecured



Pattern Example



Patterns

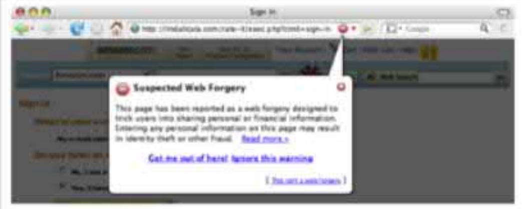
Search for Patterns

Sort by: Name Popularity

- Active Warnings
- Attractive Options
- Complete Delete
- Conveying Threats & Consequences
- Create a Security Lexicon
- Create Keys When Needed
- Delayed Unrecoverable Action
- Detailed Notifications About Security
- Direct Access to UI Components
- Disable by Default
- Disable of Services
- Disclose Significant Deviations
- Distinguish Between Run and Open
- Distinguish Internal Senders
- Distinguish Security Levels
- Email-Based Identification and Authentication
- Explicit Item Delete
- Explicit User Audit
- Failing Safely
- General Notifications About Security
- Immediate Notifications
- Immediate Options
- Indirect Access to UI Components
- Indirect Access to UI Components

Active Warnings

[en], [de]

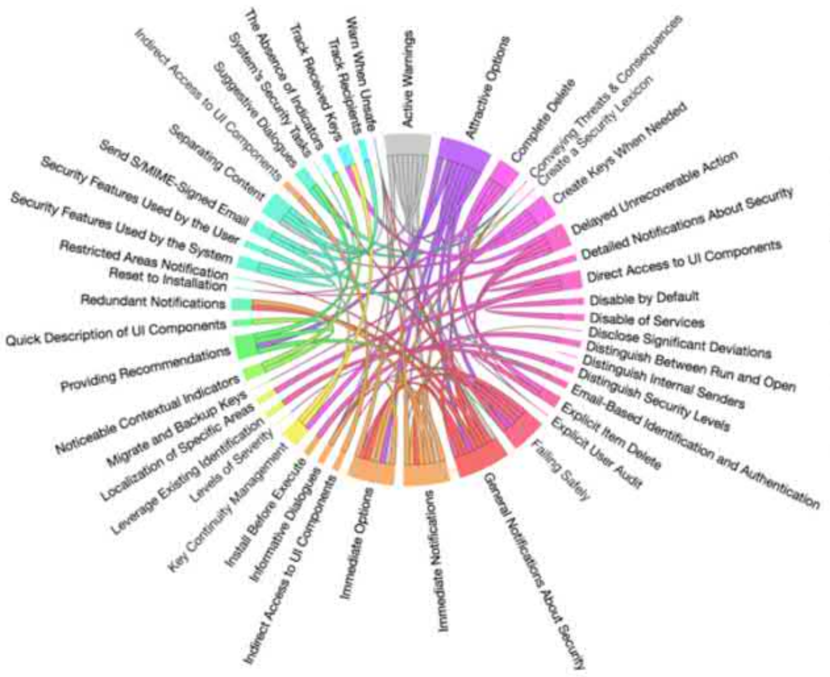
Name	Active Warnings
Sources	(Egelman 2009)
Synonyms	None
Context	Passive warning styles that do not interrupt the user may go unnoticed and thus be rendered useless. Likewise, a warning may be passive if it can be dismissed without the user taking notice of it.
Problem	Some warnings fail in very critical situations because they were not prominent enough for the user to notice them.
Solution	*Active Warnings* should be used to grab users' attention by interrupting their primary tasks, thus forcing them to acknowledge the warnings by taking an action in order to proceed.
Examples	 Suspected Web Forgery. This page has been reported as a web forgery designed to trick users into sharing personal or financial information. Entering any personal information on this page may result in identity theft or other fraud. Read more » Get me out of here! Ignore this warning [This isn't a web forgery] Source: (Egelman 2009)
Implementation	Active warnings must be designed to interrupt the primary task by either replacing the content users were expecting with the warning message, or by drawing attention away from the expected content.
Consequences	By interrupting users' primary tasks and forcing them to make a decision, significantly more users paid attention to the warnings and were ultimately protected from the phishing attack.
Dependencies	None
Relationships	[Attractive Options] [Immediate Notifications] [Conveying Threats & Consequences] [General Notifications About Security] [Immediate Options] [Separating Content]
Principles	[Convenience] [Clarity]
Guidelines	None
Check lists	None
Use cases	None
Tags	Active Warnings, Immediate Notifications, Warnings
Log history	[12/21/2015]: Added to repository

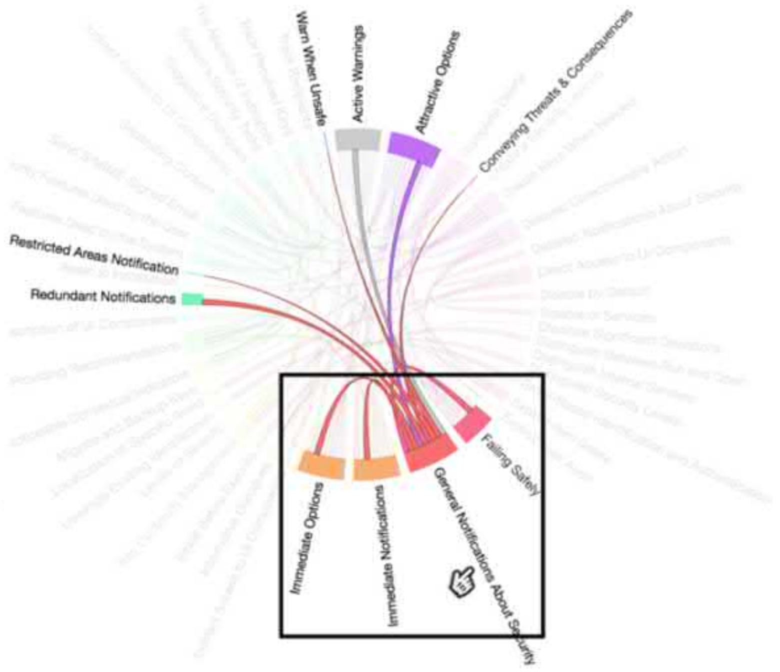
References

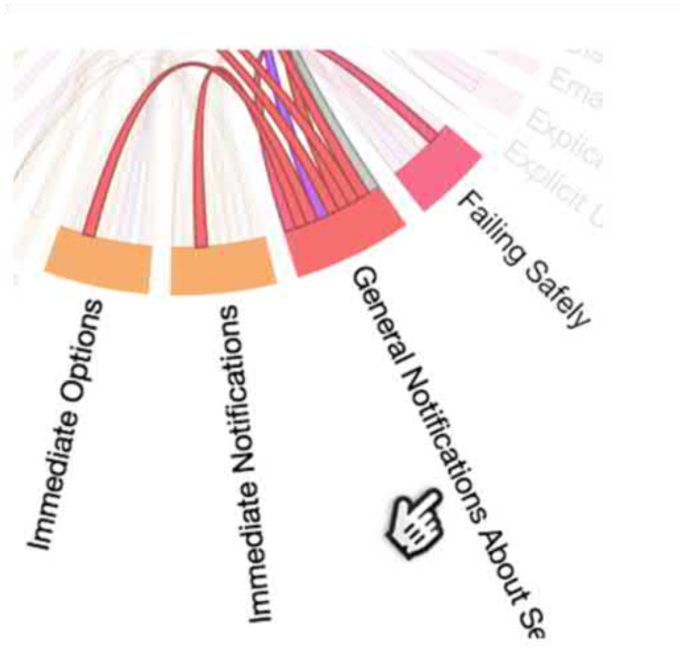
Egelman, Serge. 2009. *Trust Me: Design Patterns for Constructing Trustworthy Trust Indicators*. ProQuest. http://reports-archive.adm.cs.cmu.edu/anon/isr2009/CMU-ISR-09-110.pdf.

https://das.h-brs.de/usecured

Pattern Language


Indirect Access to UI Components
The Absence of Indications
System's Security Tasks
Suggestive Dialogues
Track Received Keys
Track Recipients
Warn When Unsafe
Active Warnings
Attractive Options
Complete Delete
Conveying Threats & Consequences
Create a Security Lexicon
Create Keys When Needed
Delayed Unrecoverable Action
Detailed Notifications About Security
Direct Access to UI Components
Disable by Default
Disable of Services
Disclose Significant Deviations
Distinguish Between Run and Open
Distinguish Internal Senders
Distinguish Security Levels
Email-Based Identification and Authentication
Explicit Item Delete
Explicit User Audit
Failing Safely
General Notifications About Security
Immediate Notifications
Immediate Options
Indirect Access to UI Components
Informative Dialogues
Install Before Execute
Key Continuity Management
Levels of Severity
Leverage Existing Identification
Localization of Specific Areas
Migrate and Backup Keys
Noticeable Contextual Indicators
Providing Recommendations
Quick Description of UI Components
Redundant Notifications
Reset to Installation
Restricted Areas Notification
Security Features Used by the System
Security Features Used by the User
Send S/MIME-Signed Email
Separating Content





Overview

Relations

Navigation

https://das.h-brs.de/usecured

Evaluation Methods

Expert Reviews

- Literature Search
- Cognitive Walkthrough
- Heuristic Evaluation
- Model-based evaluation
- …

User Reviews

- Observation
- Interviews / focus groups
- Surveys / questionnaires
- Online studies
- Laboratory experiments
- ….

Designed by iconicbestiary / Freepik

Evaluation Methods

(Expert) Heuristic Walkthrough

Prerequisite:

- At least two experts (better some more with different expertise)
  - Usability Experts
  - System designer
  - UI designer
  - Developers
  - …

Evaluation Methods

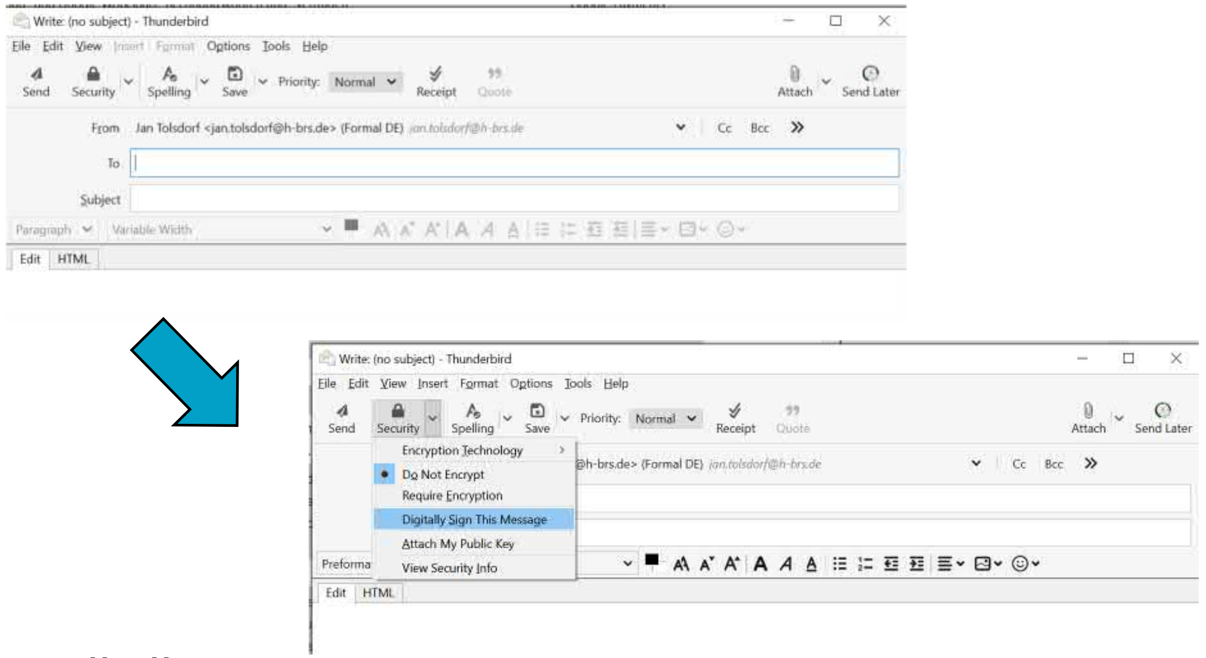
(Expert) Heuristic Walkthrough

Step 1:

- Compile a list of prioritized user tasks
- Compile a list of heuristics (principles)

1. Send an encrypted and signed email
2. Add a new public key

Evaluation Methods


Write: (no subject) - Thunderbird
File Edit View Insert Format Options Tools Help
Send Security Spelling Save Priority: Normal Receipt Quote Attach Send Later
From
To
Subject
Cc Bcc
Paragraph Variable Width
Edit HTML
Encryption Technology
Do Not Encrypt
Require Encryption
Digitally Sign This Message
Attach My Public Key
View Security Info
Preforma

From: Jan Tolsdorf <jan.tolsdorf@h-brs.de> (Formal DE) jan.tolsdorf@h-brs.de

(Expert) Heuristic Walkthrough

Step 2:

- Pass 1: Apply adapted cognitive walkthrough
  - Will the users know what they **need to do next**?
  - Will users **notice** that there is a **control available** that will allow them to accomplish the next part of their task?
  - Will users know **how** to use the **control**?
  - Will users see that **progress** is being made towards completing the task?

Evaluation Methods

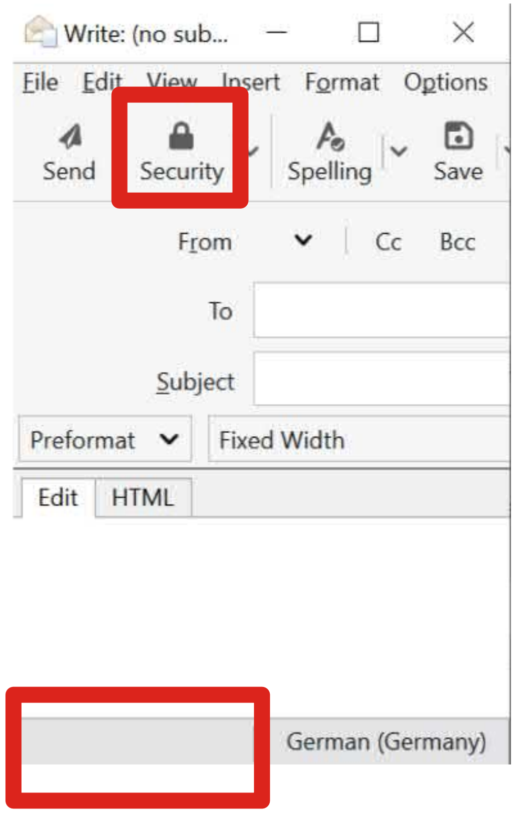
(Expert) Heuristic Walkthrough

Step 3:

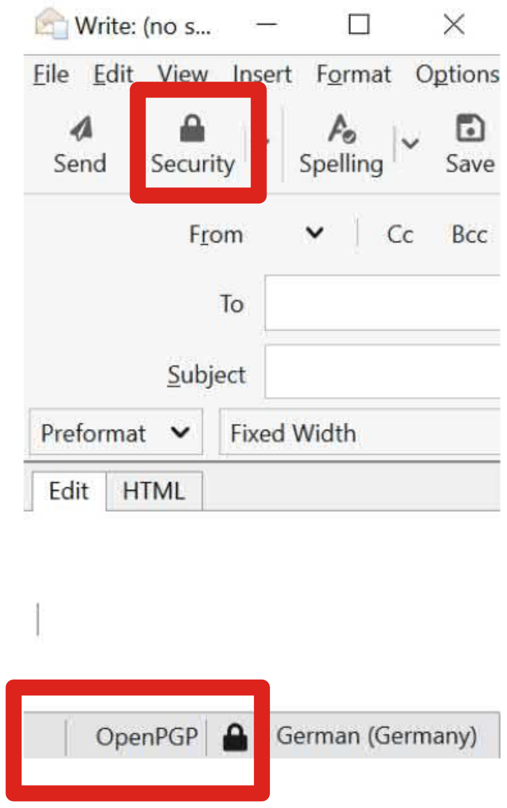
- Pass 2: Apply adapted heuristic evaluation

Name	Visibility
Sources	(Yee, 2002)
Synonyms	Visible (Furnell et al., 2006)
Intent	The system should give a clear indication of whether security is being applied.

unencrypted email



encrypted email



Evaluation Methods

(Expert) Heuristic Walkthrough

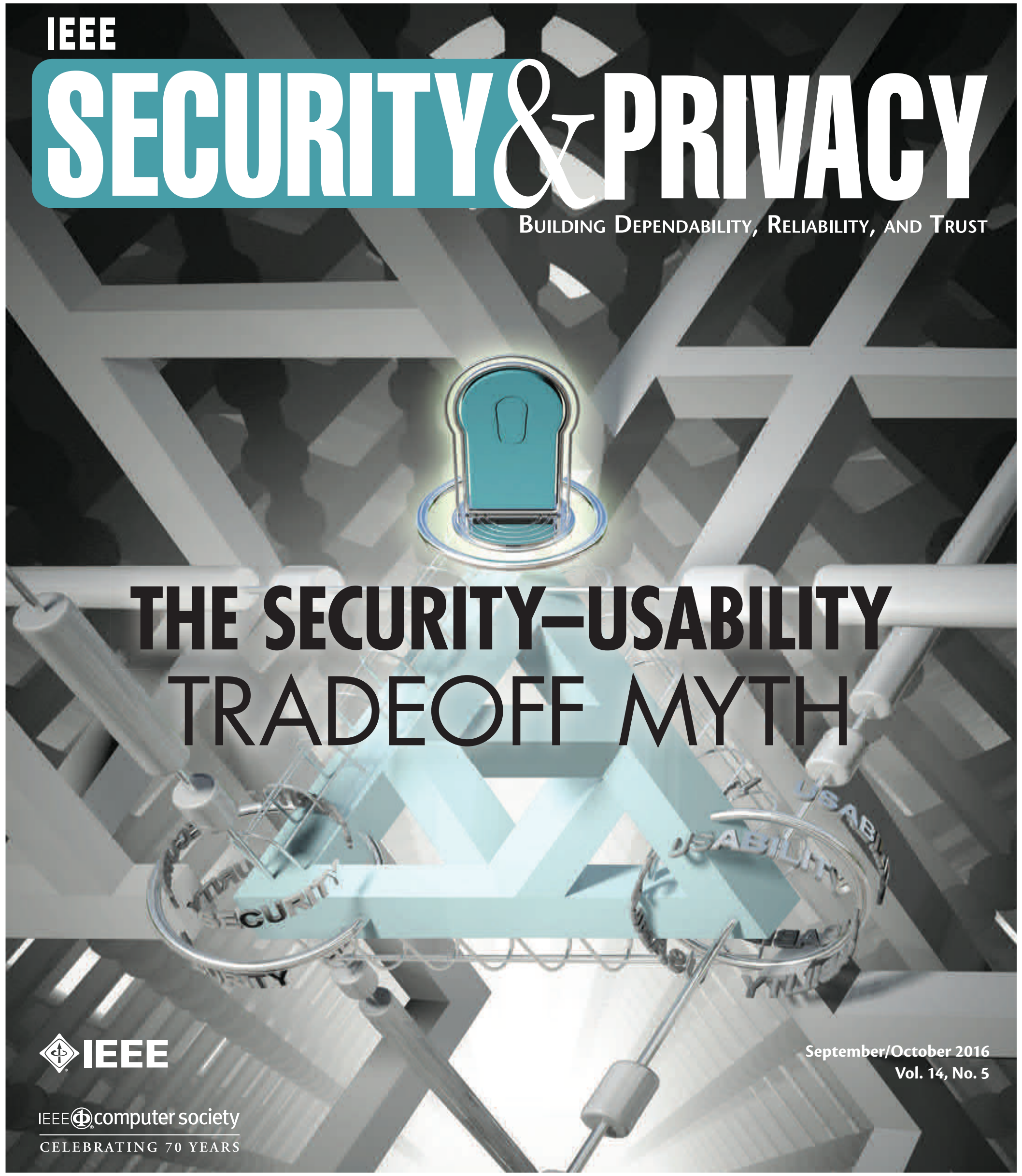
Step 4:

- Consolidate findings and compare results
- Rate problems

1. Send an encrypted and signed email

Problems:

- System status visibility is hidden: 3/5
- Menu button falsely suggests that encryption is enabled: 5/5



IEEE

SECURITY & PRIVACY

BUILDING DEPENDABILITY, RELIABILITY, AND TRUST

THE SECURITY–USABILITY TRADEOFF MYTH

IEEE

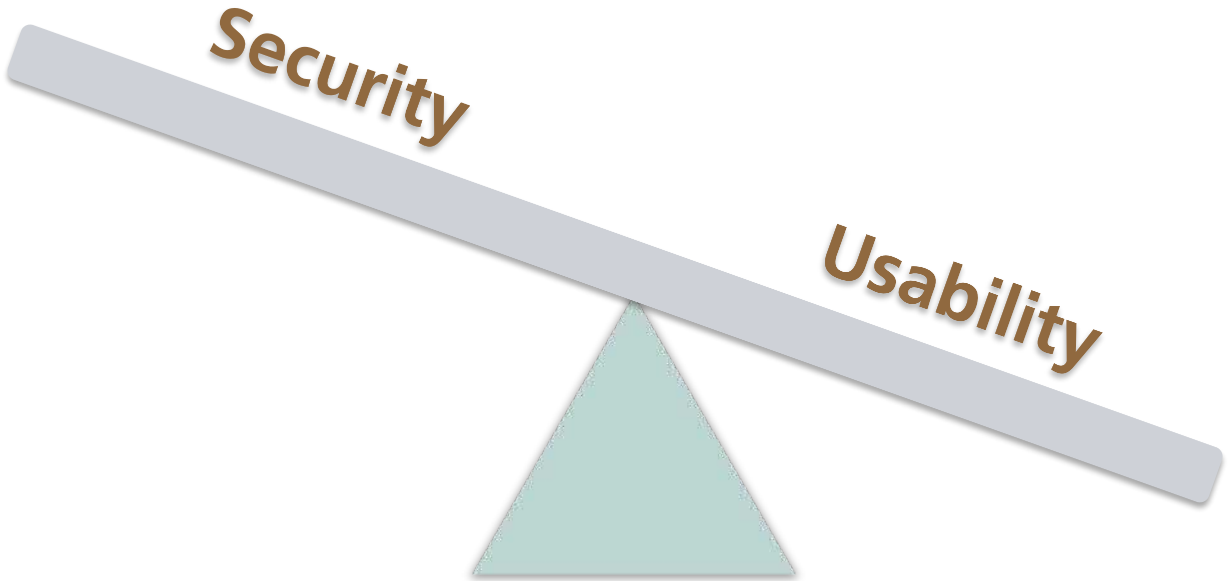
IEEE computer society
CELEBRATING 70 YEARS

September/October 2016
Vol. 14, No. 5

Source: IEEE Security & Privacy, Volume 14 Issue 5, Sept.-Oct. 2016

The Security-Usability Tradeoff Myth

1. More security means less usability?

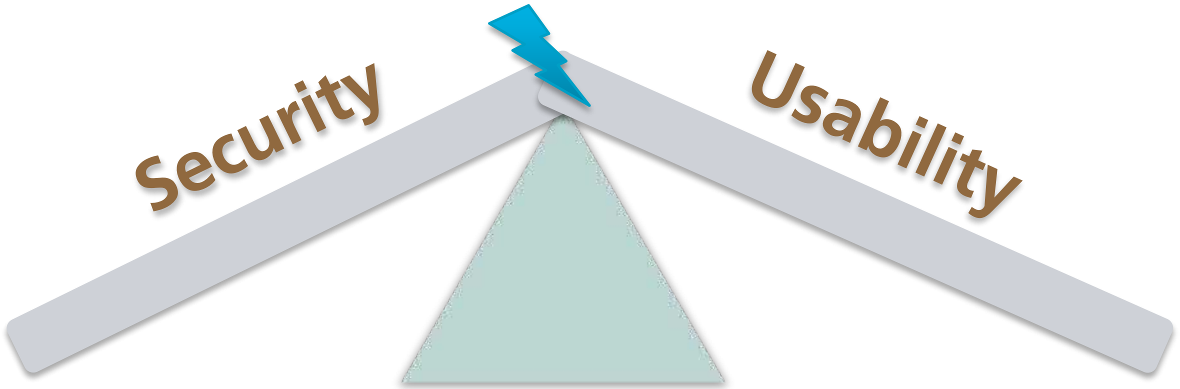


Source: M. A. Sasse, M. Smith, C. Herley, H. Lipford and K. Vaniea, "Debunking Security-Usability Tradeoff Myths," in IEEE Security & Privacy, vol. 14, no. 5, pp. 33-39, Sept. Oct. 2016

The Security-Usability Tradeoff Myth

Security breaks when usability is

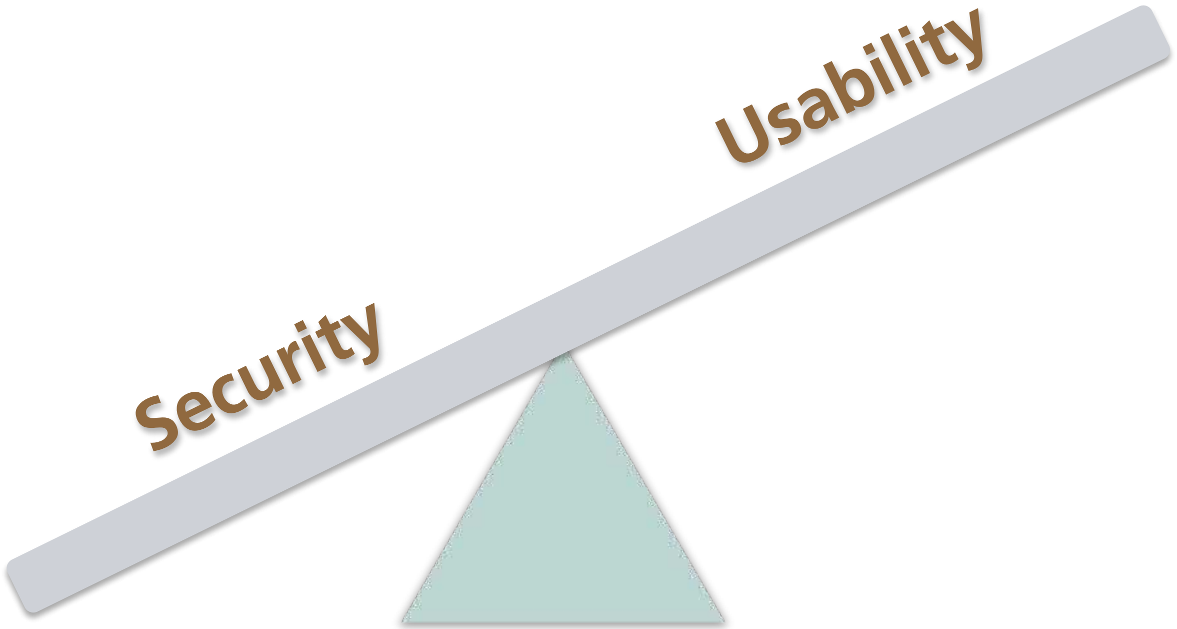
is not taken into account!



Source: M. A. Sasse, M. Smith, C. Herley, H. Lipford and K. Vaniea, "Debunking Security-Usability Tradeoff Myths," in IEEE Security & Privacy, vol. 14, no. 5, pp. 33-39, Sept. Oct. 2016

The Security-Usability Tradeoff Myth

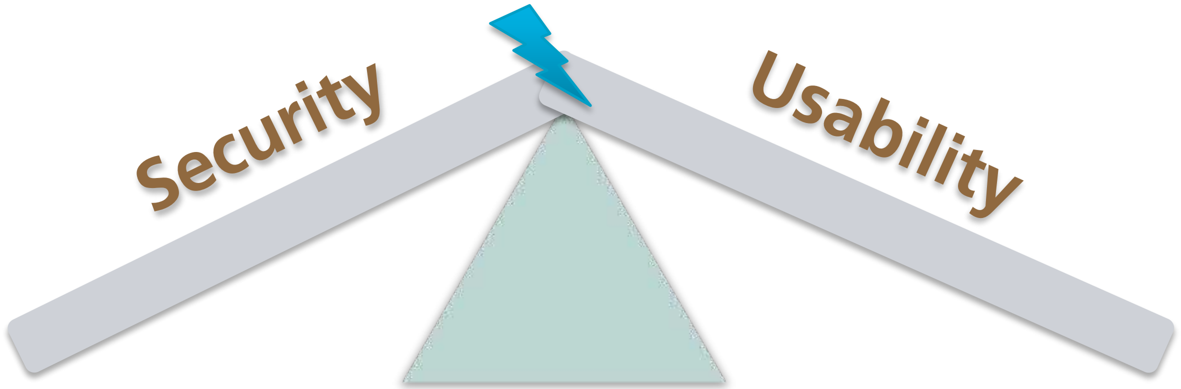
2. More usability means less security?



Source: M. A. Sasse, M. Smith, C. Herley, H. Lipford and K. Vaniea, "Debunking Security-Usability Tradeoff Myths," in IEEE Security & Privacy, vol. 14, no. 5, pp. 33-39, Sept. Oct. 2016

The Security-Usability Tradeoff Myth

Digital products cannot be used without security!

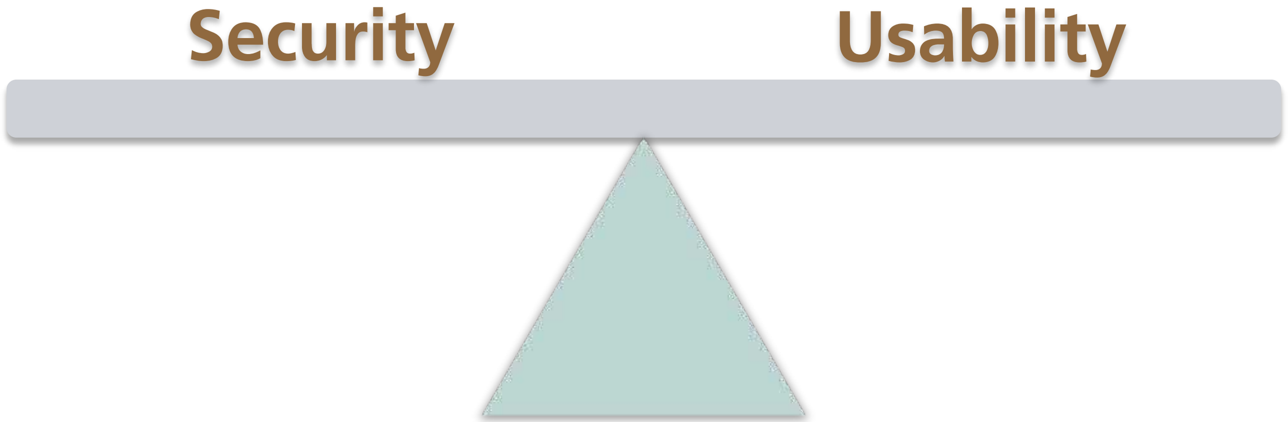


Source: M. A. Sasse, M. Smith, C. Herley, H. Lipford and K. Vaniea, "Debunking Security-Usability Tradeoff Myths," in IEEE Security & Privacy, vol. 14, no. 5, pp. 33-39, Sept. Oct. 2016

The Security-Usability Tradeoff Myth

Goal: Reflective Balance

When security features are fit for purpose, they are more likely to be used (correctly), improving overall security.



Source: M. A. Sasse, M. Smith, C. Herley, H. Lipford and K. Vaniea, "Debunking Security-Usability Tradeoff Myths," in IEEE Security & Privacy, vol. 14, no. 5, pp. 33-39, Sept. Oct. 2016

~~Elephant in the room~~
Challenges

Secondary task

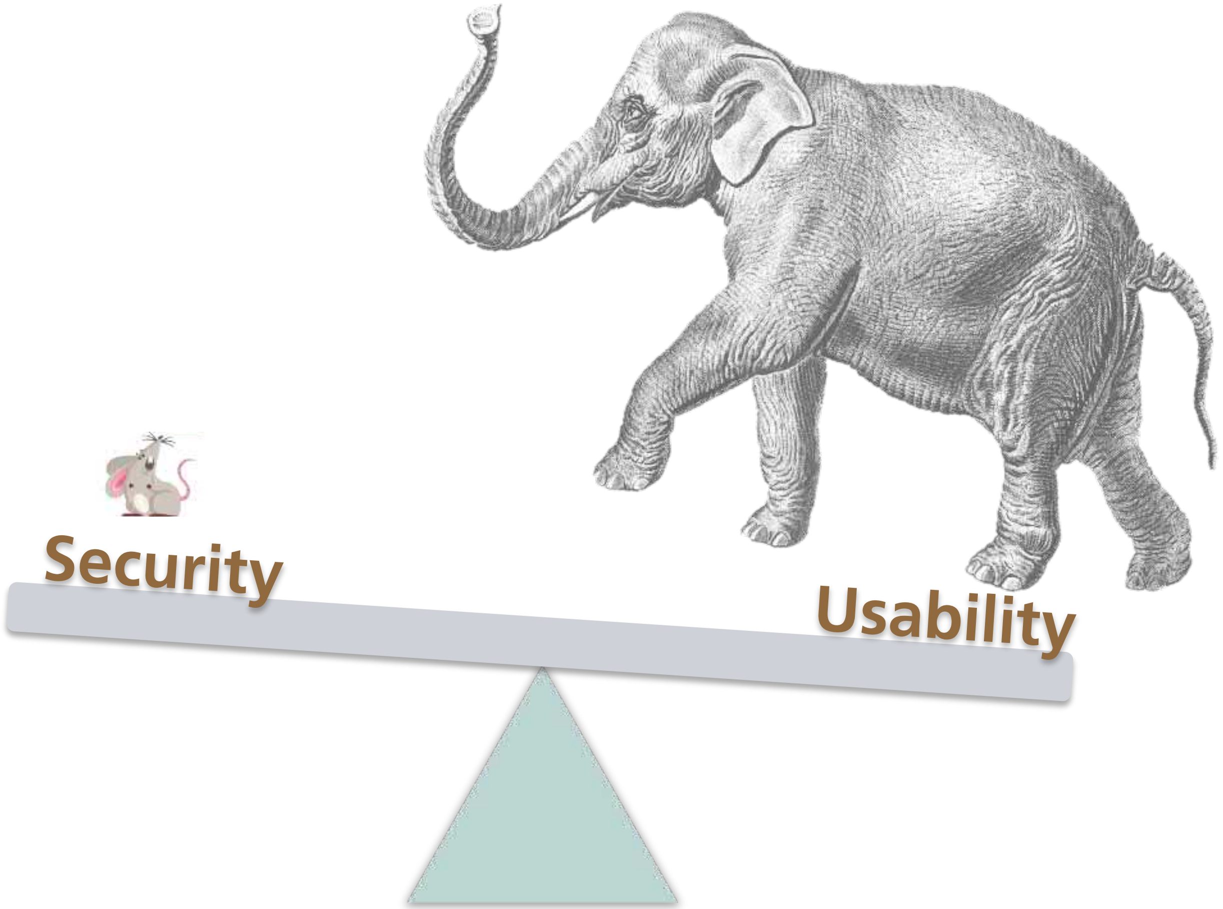
- Stress
- Physical condition
- Mental condition, concentration
- Least Effort

Resilience

- Attacking/countering
- Human characteristics
- (Cognitive abilities, habituation,
- mistakes)

Person types

- Perception of risk
- Security knowledge
- Security Behavior



Elephant: "Designed by rawpixel.com / Freepik"
Mouse: "Designed by Freepik"

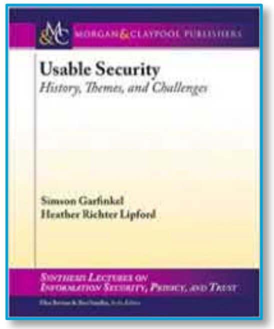
Takeaways

(1) The Security-Usability Tradeoff Myth

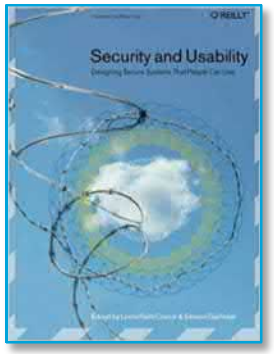
(2) Empower people to become a strong link in Security

Designed by Freepik

Recommended Readings



„Usable Security: History, Themes, and Challenges"
Simson Garfinkel and Heather Richter Lipford, 2014



„Security and Usability: Designing Secure Systems that People Can Use"
Lorrie Faith Cranor and Simson Garfinkel, 2005

Conferences

International

Symposium on Usable Privacy and Security
https://www.usenix.org/conference/soups2020

Privacy Enhancing Technologies Symposium
https://www.petsymposium.org/

Conference on Human Factors in Computing Systems
https://chi2020.acm.org/

Workshop on Usable Security and Privacy
http://www.usablesecurity.net/USEC/usec21/

Conferences

Europe

European Workshop on Usable Security
https://eusec20.cs.uchicago.edu/

Germany

MENSCH UND COMPUTER 2020
Digitaler Wandel im Fluss der Zeit

Usable Security & Privacy Workshop
https://das.h-brs.de/workshops/

Security
How was your security experience today?
Heathrow
Tell us how we can make your journey better

Picture: Peter Leo Gorski

Contact:
Luigi Lo Iacono

luigi.lo_iacono@h-brs.de

https://das.h-brs.de/