SSDEEPER: EVALUATING AND IMPROVING SSDEEP

By
Carlo Jakobs, Martin Lambertz and Jan-Niclas Hilgert
carlo.jakobs@fkie.fraunhofer.de, martin.lambertz@fkie.fraunhofer.de, jan-niclas.hilgert@fkie.fraunhofer.de

From the proceedings of

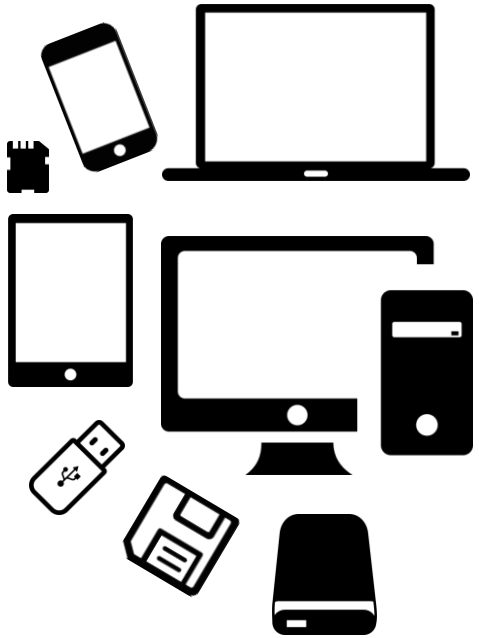
The Digital Forensic Research Workshop

DFRWS 2022 VIRTUAL USA

(July 11th - 14th)

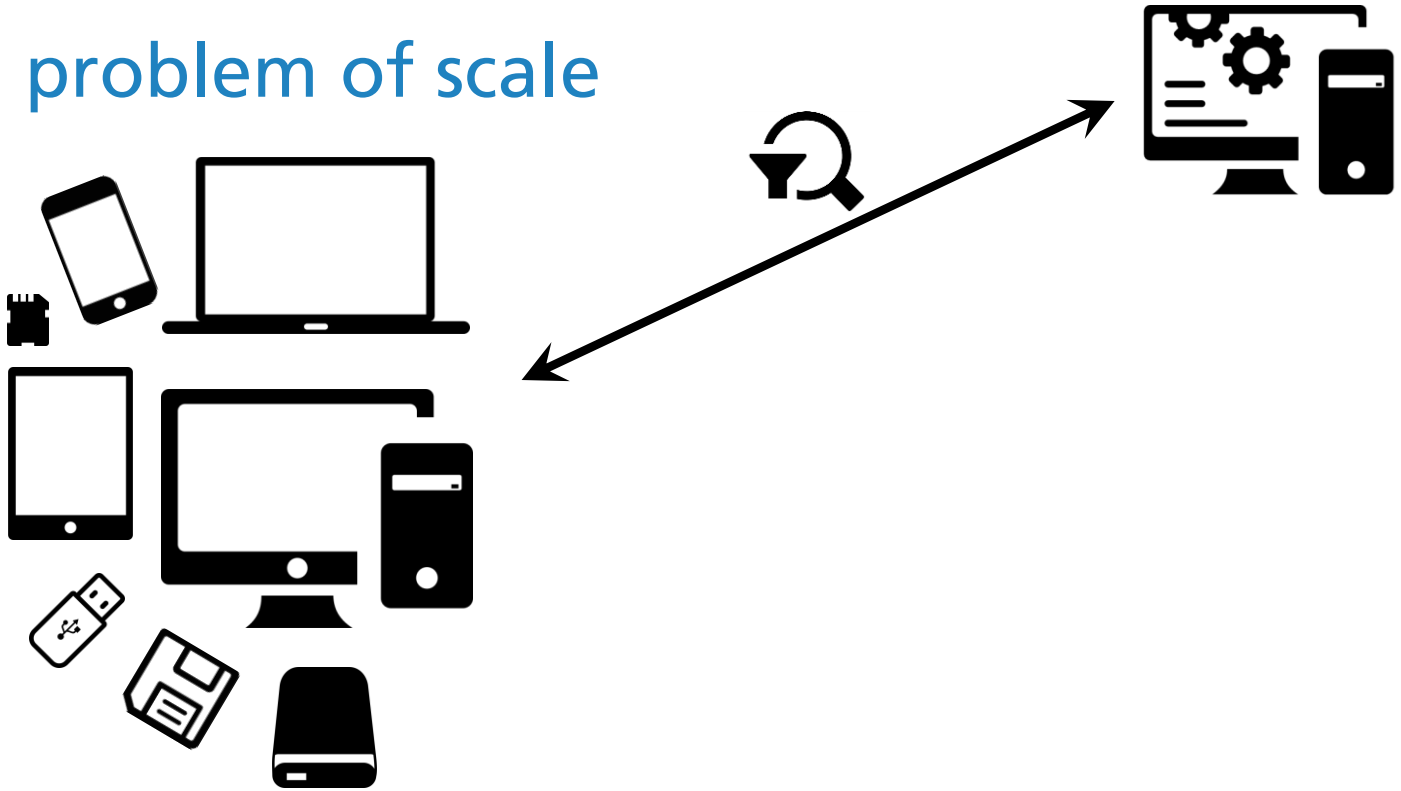
Introduction

The problem of scale



Introduction

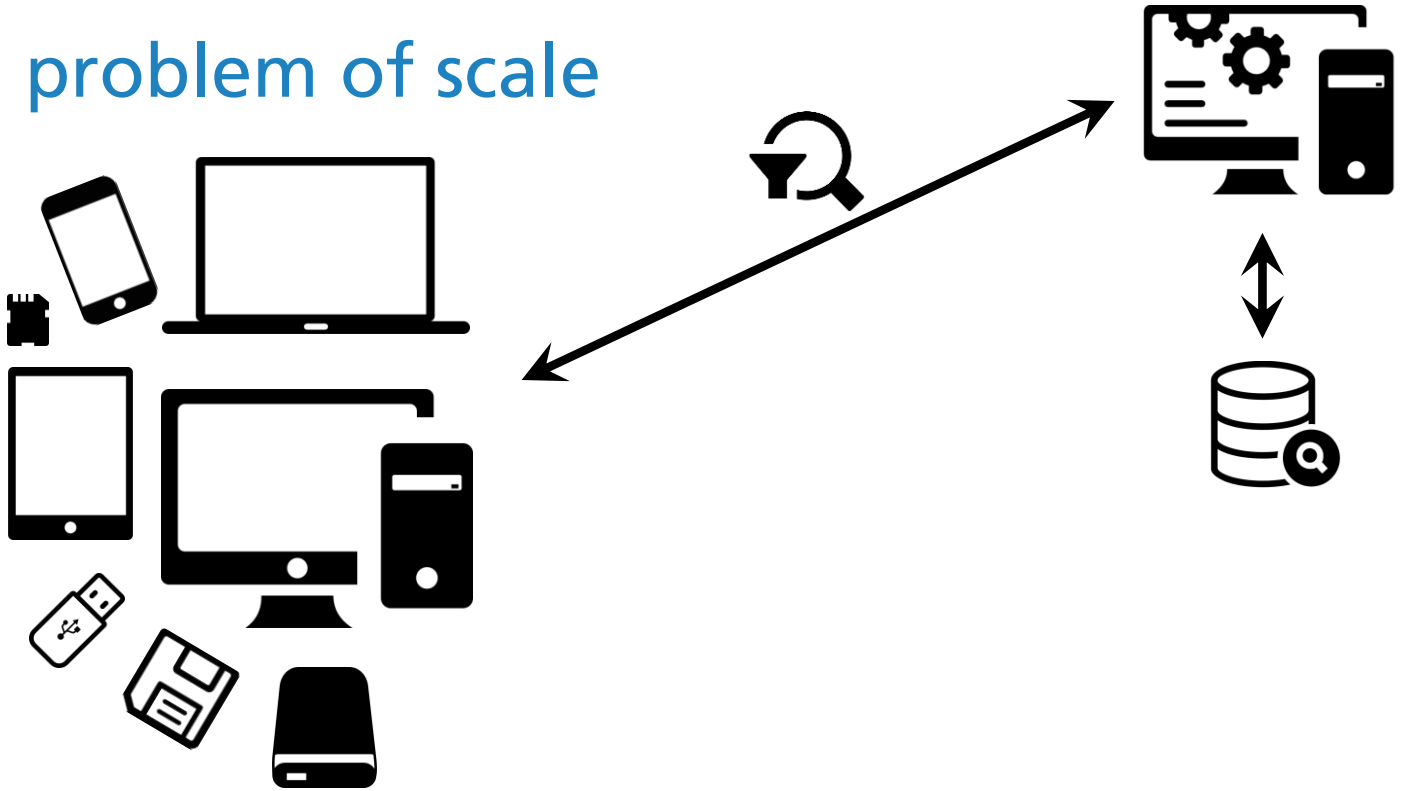
The problem of scale



Automated systems

Introduction

The problem of scale

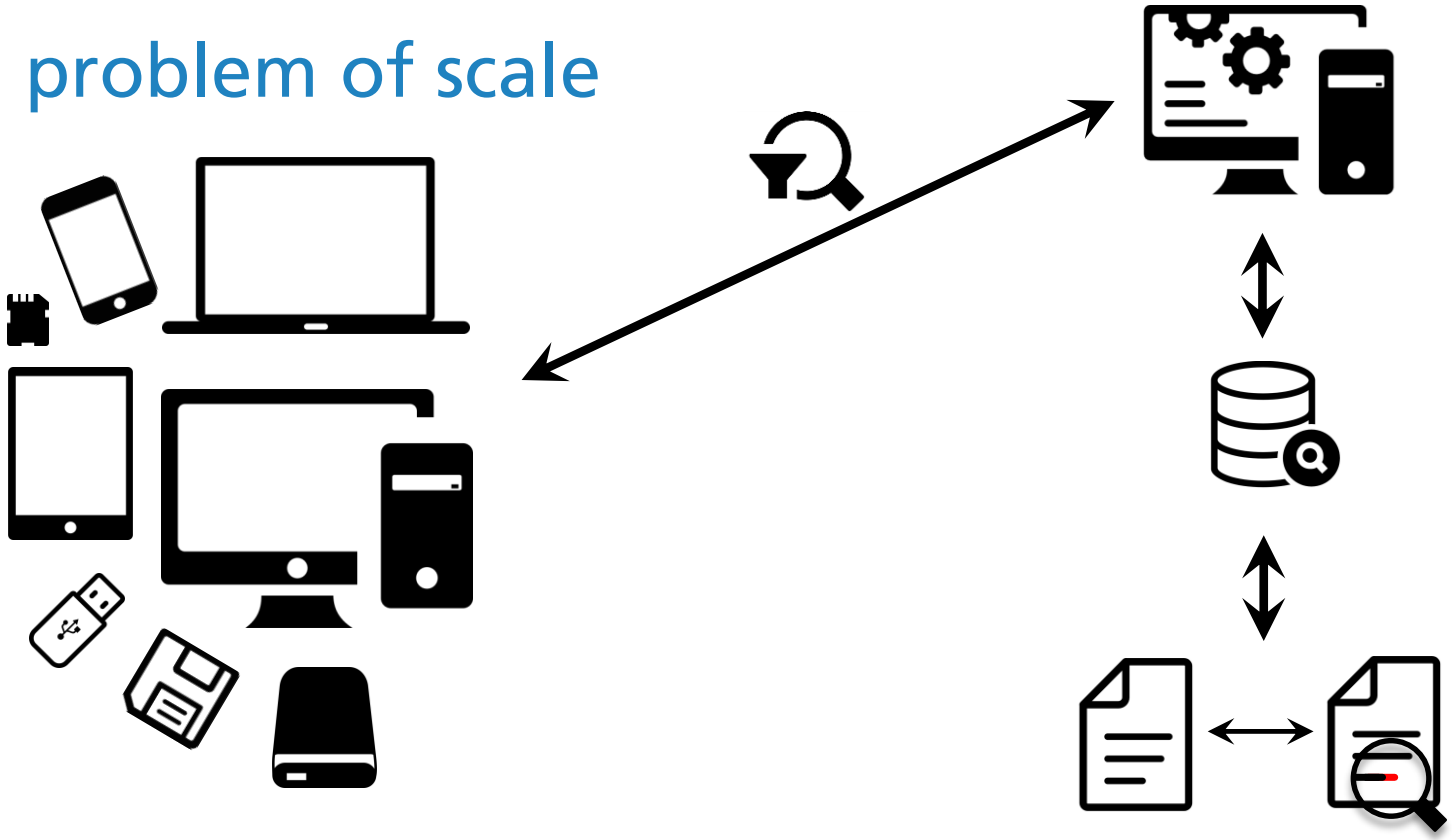


Automated systems

Traditional technique

Introduction

The problem of scale



Automated systems

Traditional technique

Limitation

Introduction – Similarity

- What is similarity?
  - Lower level approximates the similarity between two byte sequences
    - File type agnostic and fast
  - Higher level attempts to interpret the actual content
    - File type specific, requires parsing and interpretation of bytes

Introduction – Similarity Hashing

- What is similarity?
  - Lower level approximates the similarity between two byte sequences
    - File type agnostic and fast
  - Higher level attempts to interpret the actual content
    - File type specific, requires parsing and interpretation of bytes
- Similarity Hashing
  - On the lower level of abstraction
  - Similarity Preserving Hash Function (SPHF)
  - Comparison Function (CF)

Similarity Hashing – Properties

efficiency & usability

security

Similarity Hashing – Properties

efficiency & usability

Runtime

security

Fraunhofer FKIE

Similarity Hashing – Properties

efficiency & usability

Runtime

Compression

security

Fraunhofer FKIE

Similarity Hashing – Properties

efficiency & usability	Runtime
	Compression
	Accuracy
security	

Fraunhofer FKIE

Similarity Hashing – Properties

efficiency & usability	Runtime
	Compression
	Accuracy
security	Coverage

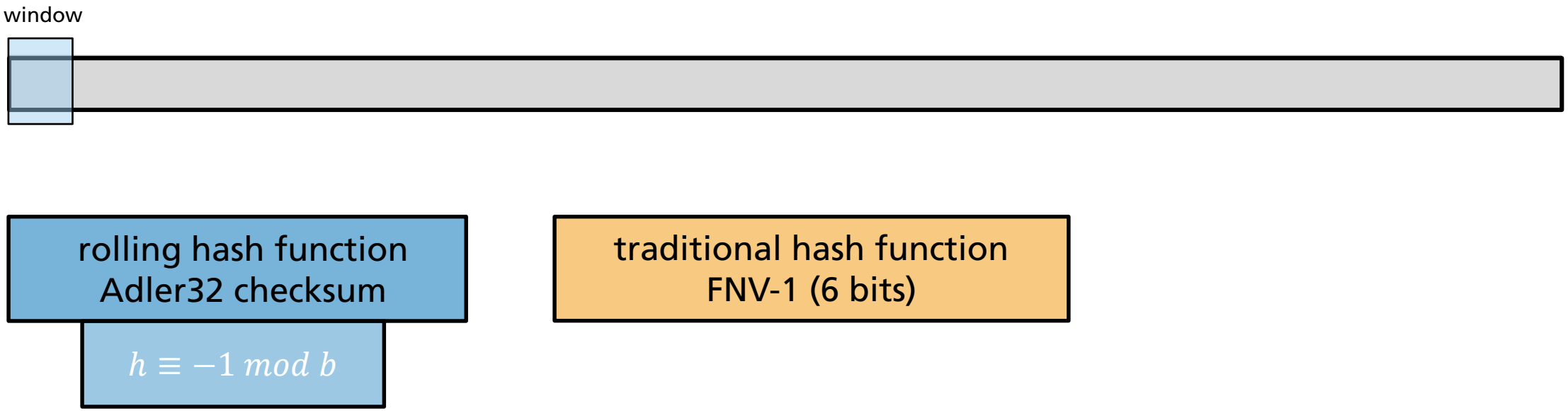
Fraunhofer FKIE

Similarity Hashing – Properties

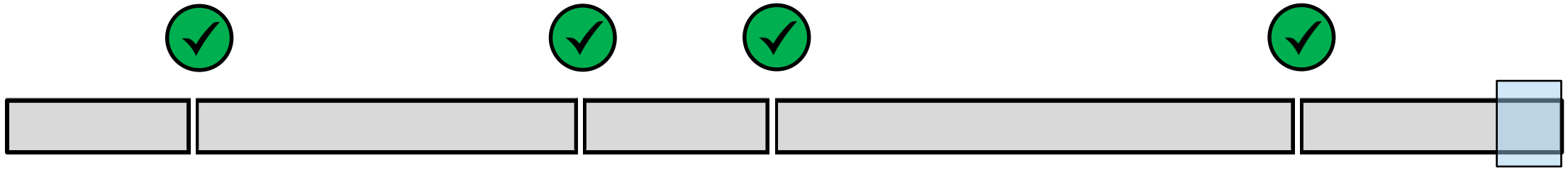
efficiency & usability	Runtime
	Compression
	Accuracy
security	Coverage
	Obfuscation resistance

Fraunhofer FKIE

Similarity Hashing – ssdeep – SPHF



Similarity Hashing – ssdeep – SPHF



rolling hash function
Adler32 checksum

$h \equiv -1 \; mod \; b$

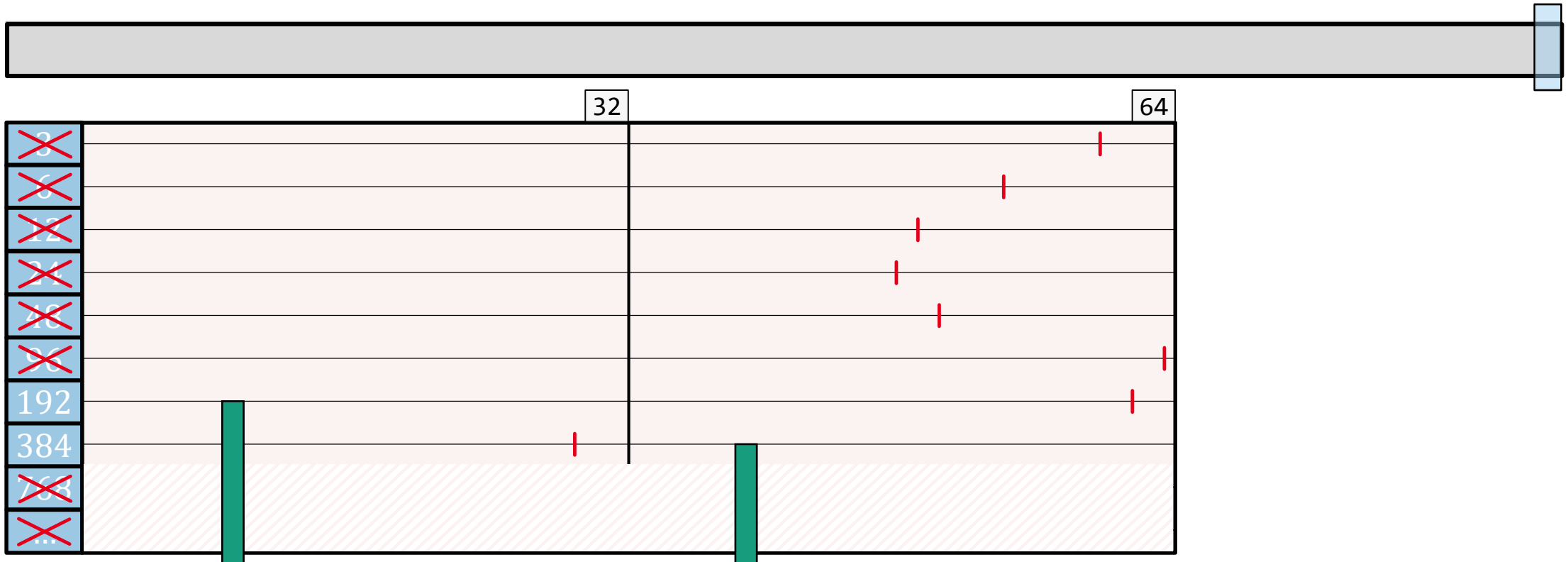
traditional hash function
FNV-1 (6 bits)

b	t	Q	y	N	F

- Recalculation if $len < 32$
  - Use $b = b/2$
  - `blocksize:hash:hash,filename`
- Variety of improvements

Fraunhofer FKIE

Similarity Hashing – ssdeep – SPHF

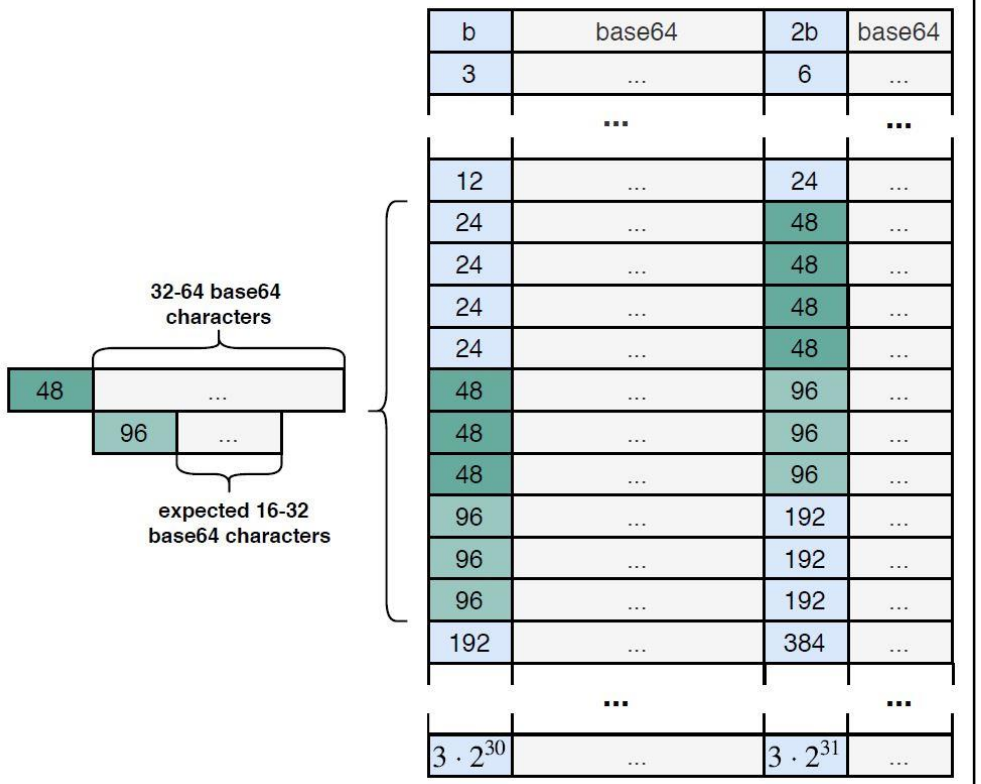



```
$ ssdeep somefile.jpg
ssdeep,1.1–blocksize:hash:hash,filename
192:tQyNFgeWKVwVyF18TYjEiHRElsOMuKyN46LouR1dPbCNXf4pMt9lj3IOYEVF:clxA1+0OAy1LVXPbAXf7t9ZbYOF,“somefile.jpg“
```


Fraunhofer FKIE

Similarity Hashing – ssdeep

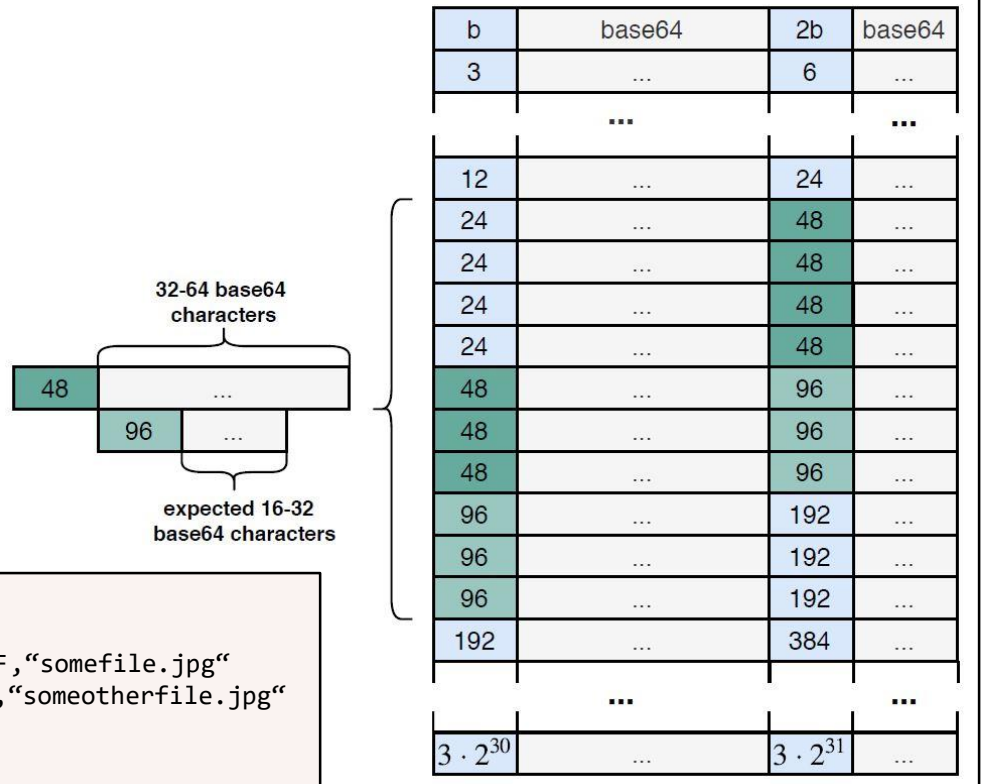
- CF
  - Matching block sizes
  - Levensthein edit distance – string comparison



Fraunhofer FKIE

Similarity Hashing – ssdeep

- CF
  - Matching block sizes
  - Levensthein edit distance – string comparison




```
$ ssdeep somefile.jpg someotherfile.jpg
ssdeep,1.1–blocksize:hash:hash,filename
192:tQyNFgeWKVwVyF18TYjEiHRElsOMuKyN46LouR1dPbCNXf4pMt9lj3IOYEVF:clxA1+0OAy1LVXPbAXf7t9ZbYOF,“somefile.jpg“
192:tQyNFgeWKVuyF18TYjEiHRElsOMuKyN46LouR1dPbCNXf4pMt9lj3IOYEVF:clxA1+0OAy1LVXPbAXf7t9ZbYOF,“someotherfile.jpg“

$ ssdeep somefile.jpg someotherfile.jpg > known_hashes.txt
$ ssdeep –m known_hashes.txt someotherfile.jpg
someotherfile.jpg matches somefile.jpg (99)
someotherfile.jpg matches someotherfile.jpg (100)
```


Fraunhofer FKIE

Similarity Hashing – ssdeep – Properties

efficiency & usability

Runtime

Compression

Accuracy

security

Coverage

Obfuscation resistance

Fraunhofer FKIE

Similarity Hashing – ssdeep – Properties

efficiency & usability

Runtime

SPHF and CF components fast in calculation

Compression

Accuracy

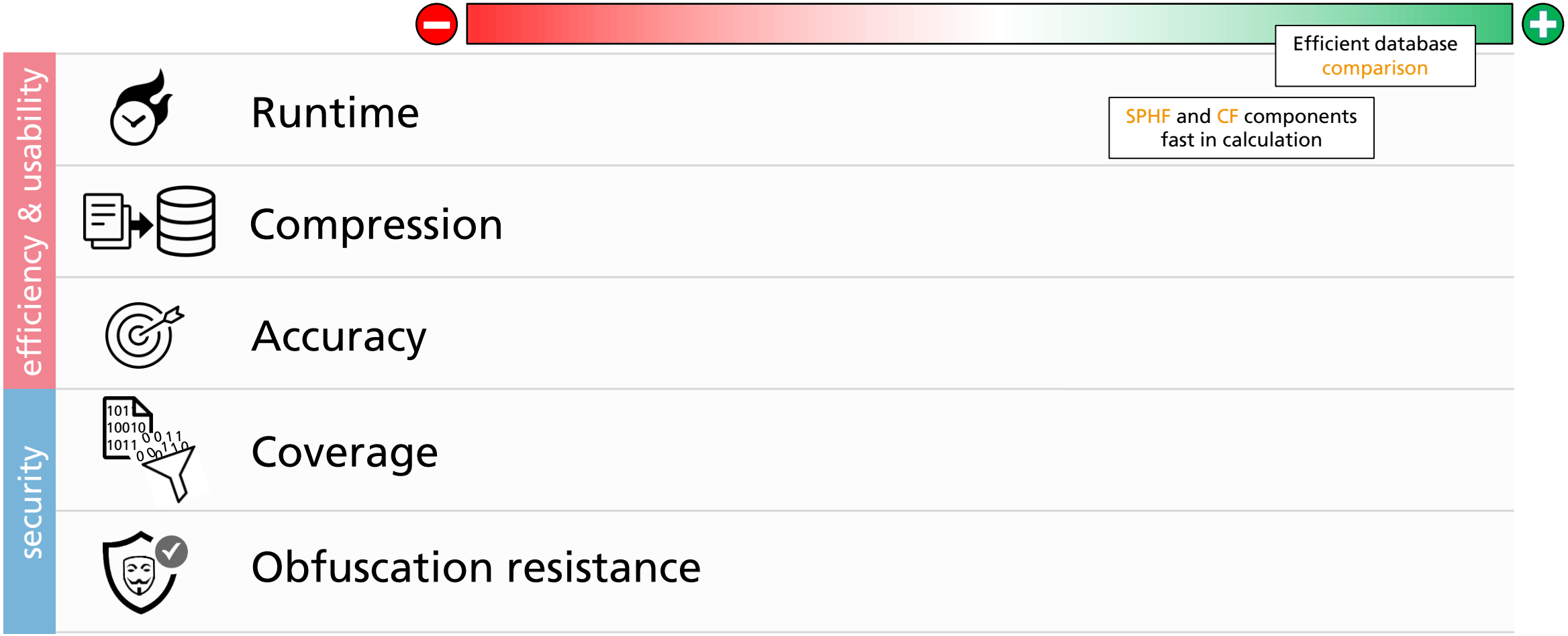
security

Coverage

Obfuscation resistance

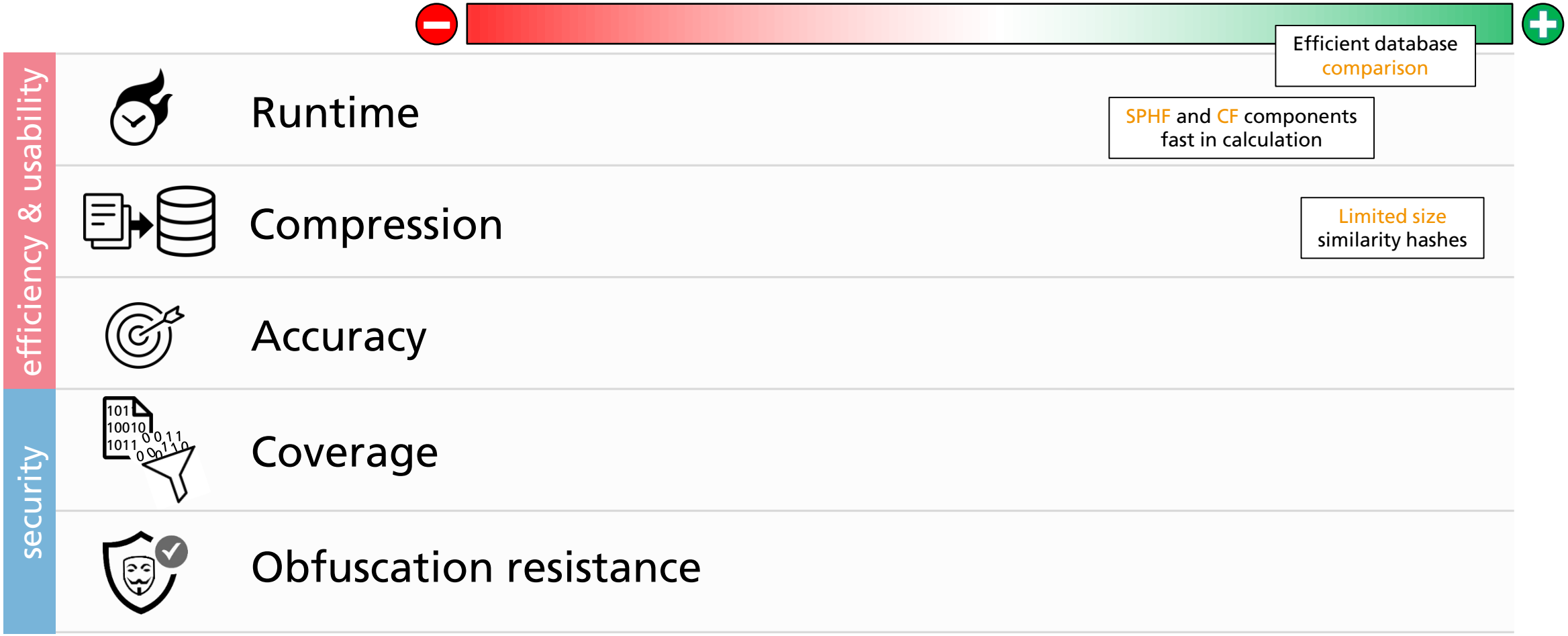
Fraunhofer FKIE

Similarity Hashing – ssdeep – Properties



Fraunhofer FKIE

Similarity Hashing – ssdeep – Properties



Fraunhofer FKIE

Similarity Hashing – ssdeep – Properties

efficiency & usability

security

Runtime

Compression

Accuracy

Coverage

Obfuscation resistance

Efficient database comparison

SPHF and CF components fast in calculation

Limited size similarity hashes

Hash contains little information

Fraunhofer FKIE

Similarity Hashing – ssdeep – Properties

efficiency & usability

security

Runtime	SPHF and CF components fast in calculation; Efficient database comparison
Compression	Limited size similarity hashes
Accuracy	Hash contains little information; False positives
Coverage	
Obfuscation resistance	

Fraunhofer FKIE

Similarity Hashing – ssdeep – Properties

efficiency & usability

Runtime

Efficient database comparison

SPHF and CF components fast in calculation

Compression

Limited size similarity hashes

Accuracy

Hash contains little information

Limited comparability

False positives

security

Coverage

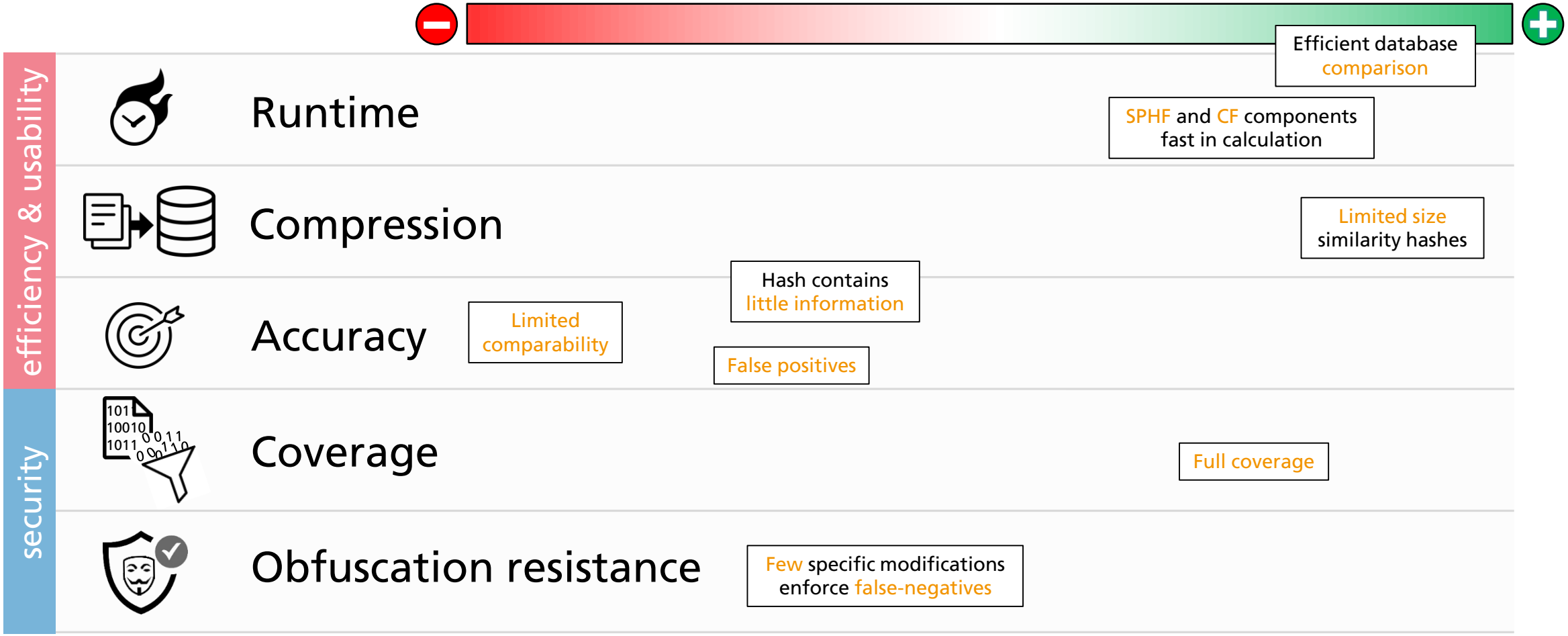
Obfuscation resistance

Fraunhofer FKIE

Similarity Hashing – ssdeep – Properties

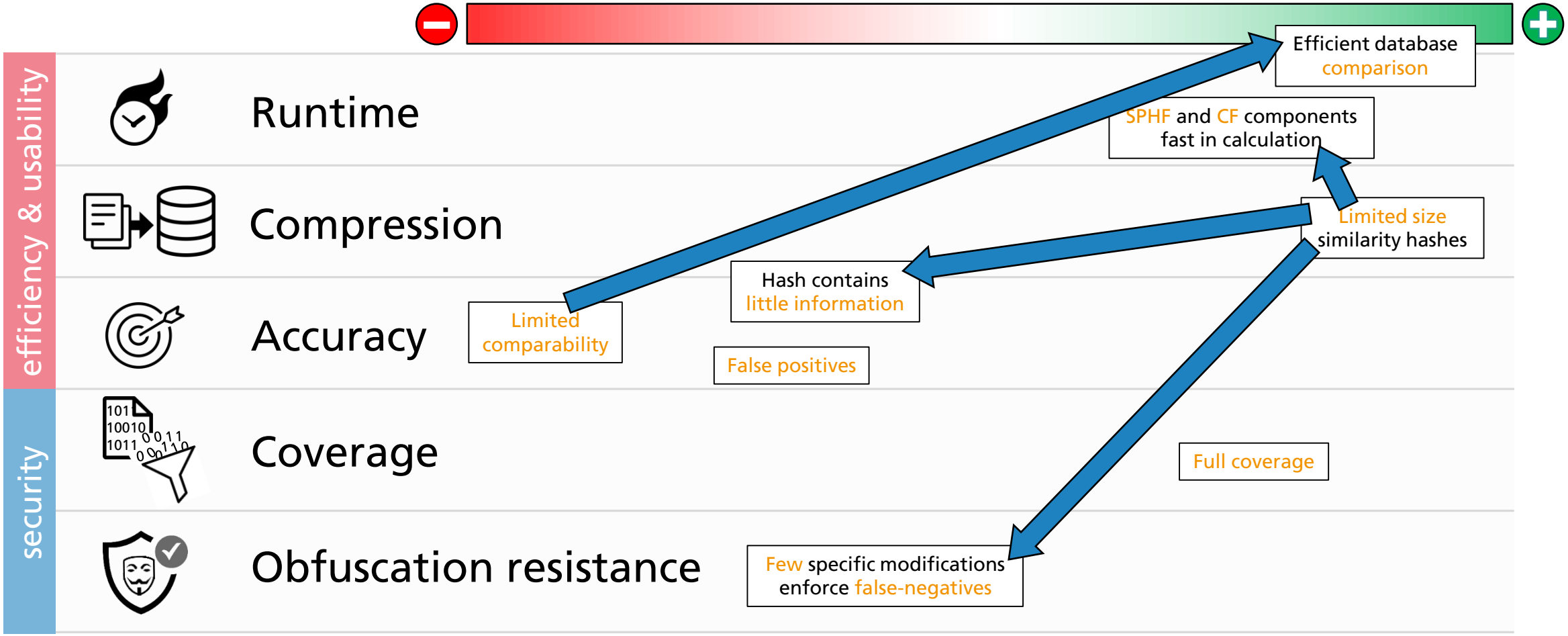
Fraunhofer FKIE

Similarity Hashing – ssdeep – Properties



Fraunhofer FKIE

Similarity Hashing – ssdeep – Properties



Fraunhofer FKIE

Analysis and improvement of ssdeep

Category	Inconsistencies	Code bugs	Possible approaches for an optimization
Modification	Modifications		
Goal	Assessment	Errors & Optimizations	Use Case

Fraunhofer FKIE

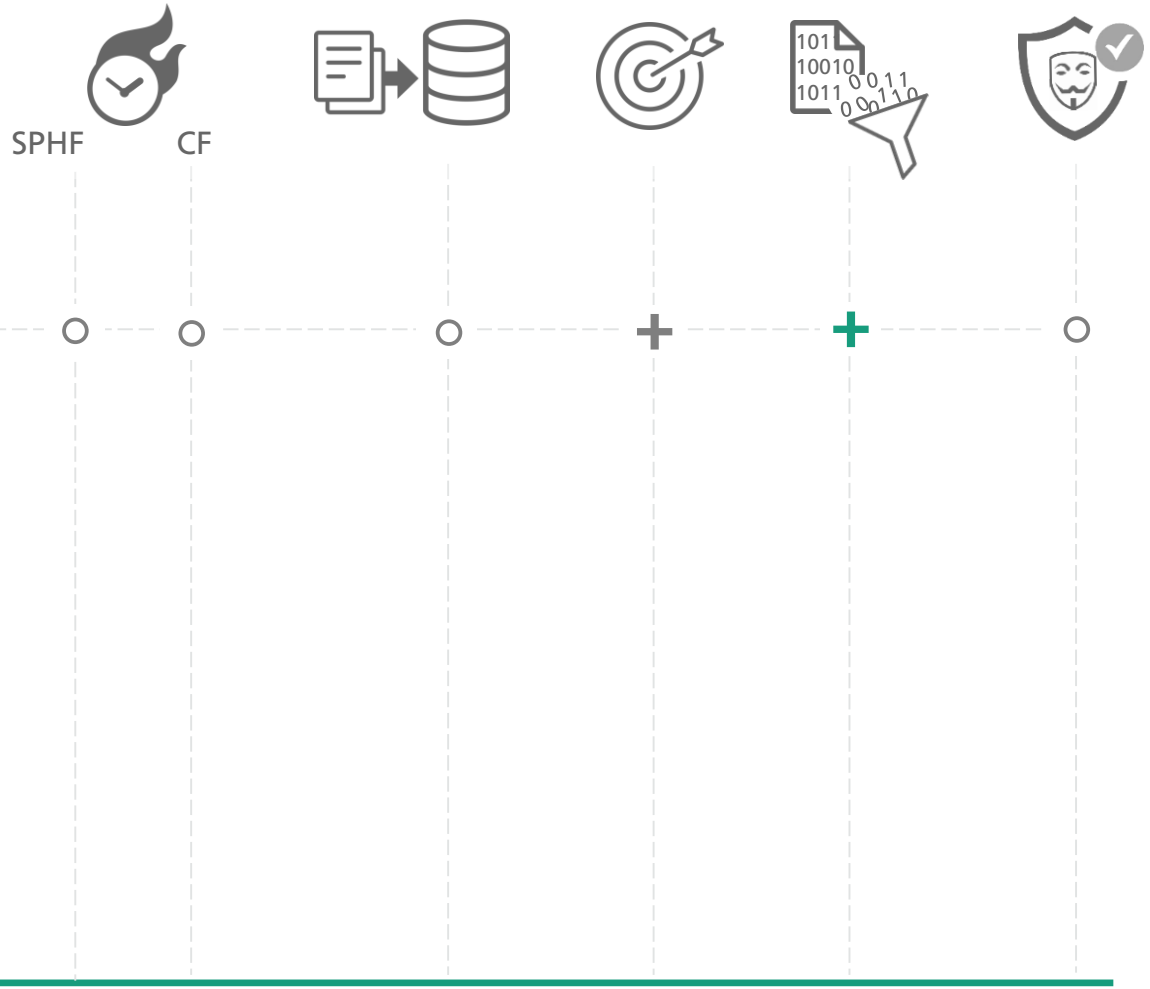
Analysis – Inconsistencies

	SPHF	CF				
Restructuring of the SPHF						
Compatibility: 32 character-limitation						
Modification `-no32lim`	+	-	-	+	+	+
CF						
Maximum matching score						
Modification `-nomax`	○	+	○	+	○	-
Distance calculation via position array						
Modification `–nopa`	○	–	○	○	○	○
Common substring						
Modification `-nocommonsub`	○	○	○	+	○	+

Fraunhofer FKIE

Analysis – Errors

- Last segment bug
  - Empty or missing segment
    - Modification `-bugfix`



Fraunhofer FKIE

Analysis – Optimizations

	SPHF	CF				
Increased trigger progression constant						
Modification `-4b`	+	-	-	+	○	+
Exchanging the rolling hash function						
Modification `-djb2`	+	○	○	○	○	○
Modification `-polynomial`	+	○	○	○	○	○

Fraunhofer FKIE

Analysis – Summarized

	SPHF	CF				
Modification -nocommonsub	○	○	○	+	○	+
Modification -nomax	○	+	○	+	○	-
Modification -nopa	○	-	○	○	○	○
Modification -no32lim, -bugfix (-refactored)	+	-	-	+	+	+
Modification -4b	+	-	-	+	○	+
Modification -djb2	+	○	○	○	○	○
Modification -polynomial	+	○	○	○	○	○

Fraunhofer FKIE

Evaluation

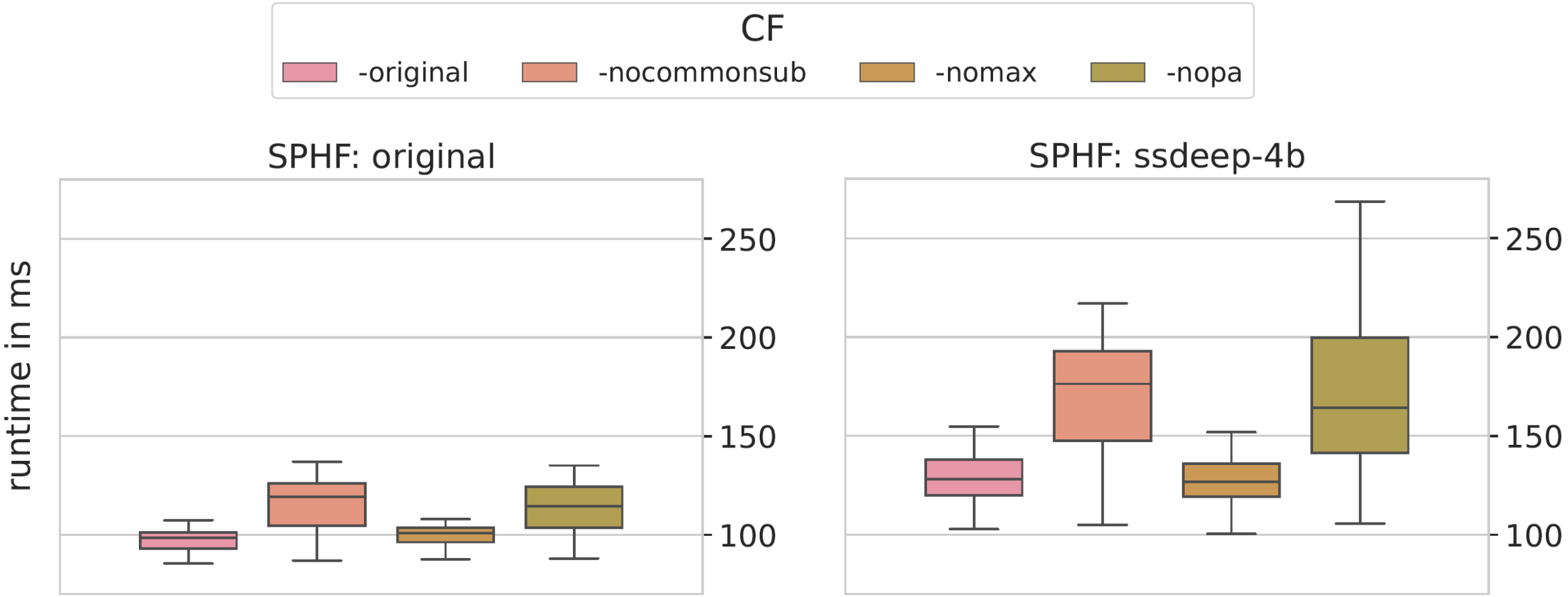
- Similarity Hashing properties
- Most promising candidates
  - SPHF
    - `-refactored-djb2`
    - `-refactored-4b-djb2`
  - SPHF & CF
    - `-refactored-djb2-nomax`
    - `-refactored-4b-djb2-nocommonsub`
- Publicly available on our GitHub repository:
  `https://github.com/fkie-cad/ssdeeper`

Evaluation – Runtime – SPHF

	Properties of the hash database					Average throughput (KB/ms)	
SPHF	Compression ratio	Avg(len(s1))	Avg(len(s2))	Signature ratio	Backward-compatibility	Real data	Synthetic data
-original						48.629	69.619
-refactored-djb2						**57.608**	**89.068**
-refactored-4b-djb2						**58.876**	**89.751**

Fraunhofer FKIE

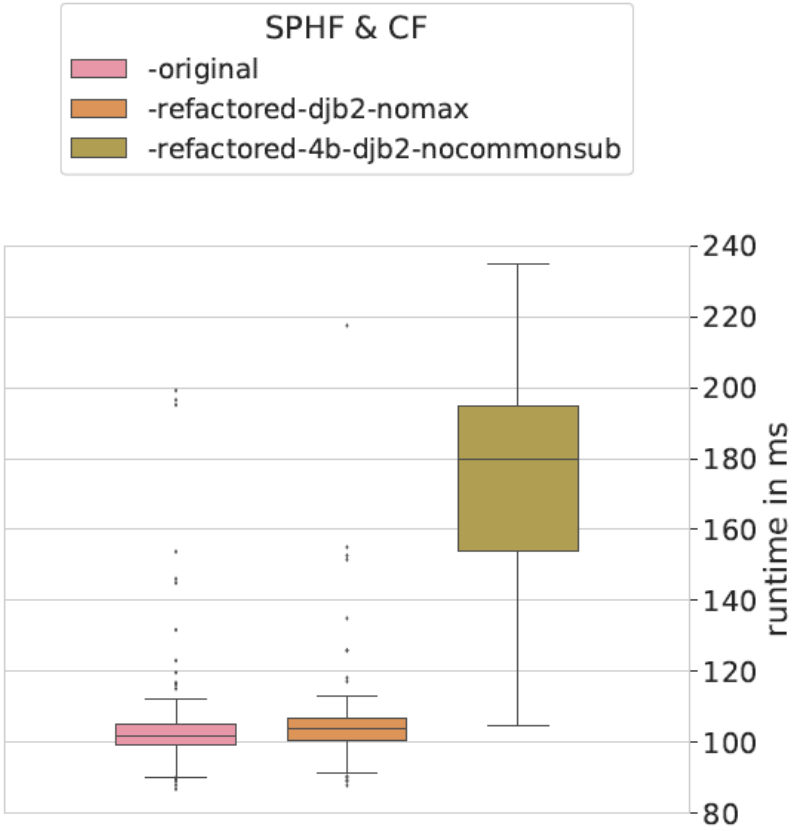
Evaluation – Runtime – CF



Fraunhofer FKIE

Evaluation – Runtime – SPHF & CF

SPHF & CF	Average runtime
-original	105.158
-refactored-djb2-nomax	**104.826**
-refactored-4b-djb2-nocommonsub	**175.919**



Fraunhofer FKIE

Evaluation – Compression

	Properties of the hash database					Average throughput (KB/ms)	
SPHF	Compression ratio	Avg(len(s1))	Avg(len(s2))	Signature ratio	Backward-compatibility	Real data	Synthetic data
-original	4548.82	50.51	22.65	3.01	n/a	48.629	69.619
-refactored-djb2	**4446.78**	50.56	25.49	2.33	no	57.608	89.068
-refactored-4b-djb2	**3824.28**	77.68	19.79	5.23	no	58.876	89.751

Fraunhofer FKIE

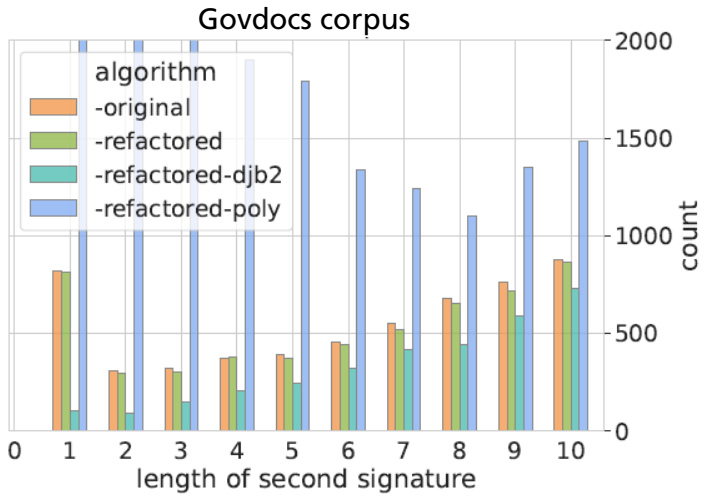
Evaluation – Compression

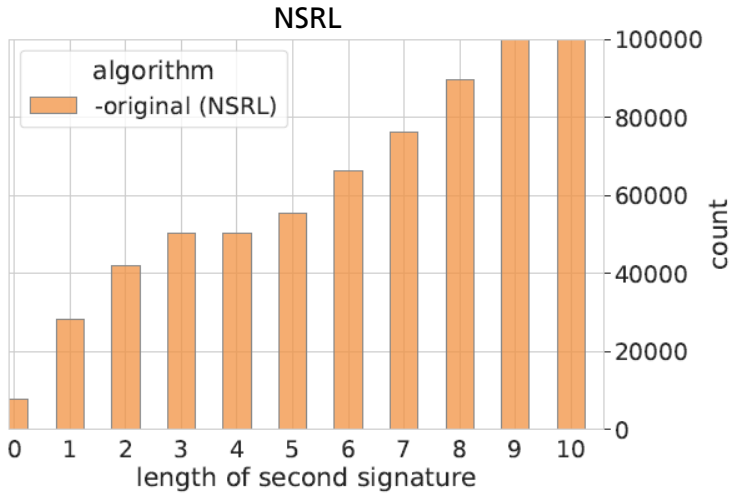
	Properties of the hash database					Average throughput (KB/ms)	
SPHF	Compression ratio	Avg(len(s1))	Avg(len(s2))	Signature ratio	Backward-compatibility	Real data	Synthetic data
`-original`	4548.82	50.51	22.65	3.01	n/a	48.629	69.619
`-refactored-djb2`	4446.78	**50.56**	**25.49**	**2.33**	no	57.608	89.068
`-refactored-4b-djb2`	3824.28	**77.68**	**19.79**	**5.23**	no	58.876	89.751

Fraunhofer FKIE

Evaluation – Compression

SPHF	Properties of the hash database				
	Compression ratio	Avg(len(s1))	Avg(len(s2))	Signature ratio	Backward-compatibility
`-original`	4548.82	50.51	22.65	3.01	n/a
`-refactored-djb2`	4446.78	**50.56**	**25.49**	**2.33**	no
`-refactored-4b-djb2`	3824.28	**77.68**	**19.79**	**5.23**	no





Fraunhofer FKIE

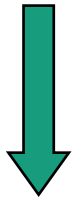
Evaluation – Accuracy

synthetic data corpus

- Randomness provided by input
  - Choice of rolling hash function
- Assessment of modification -4b

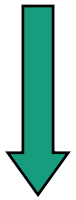
Evaluation – Accuracy

synthetic data corpus



- Randomness provided by input
  - Choice of rolling hash function
- Assessment of modification `-4b`

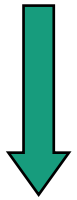
real data fragment corpus



- Worst case scenario
- Assessment of modification `-4b`, `-nocommonsub`, `-djb2` & `-nomax`

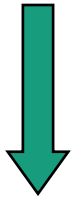
Evaluation – Accuracy

synthetic data corpus



- Randomness provided by input
  - Choice of rolling hash function
- Assessment of modification `-4b`

real data fragment corpus

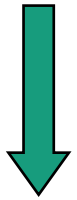


- Worst case scenario
- Assessment of modification `-4b`, `-nocommonsub`, `-djb2` & `-nomax`

- How to define accuracy and similarity?
  - The smallest error?
  - Here, variance & reliability

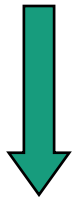
Evaluation – Accuracy

synthetic data corpus



- Randomness provided by input
  - Choice of rolling hash function
- Assessment of modification `-4b`

real data fragment corpus



- Worst case scenario
- Assessment of modification `-4b`, `-nocommonsub`, `-djb2` & `-nomax`

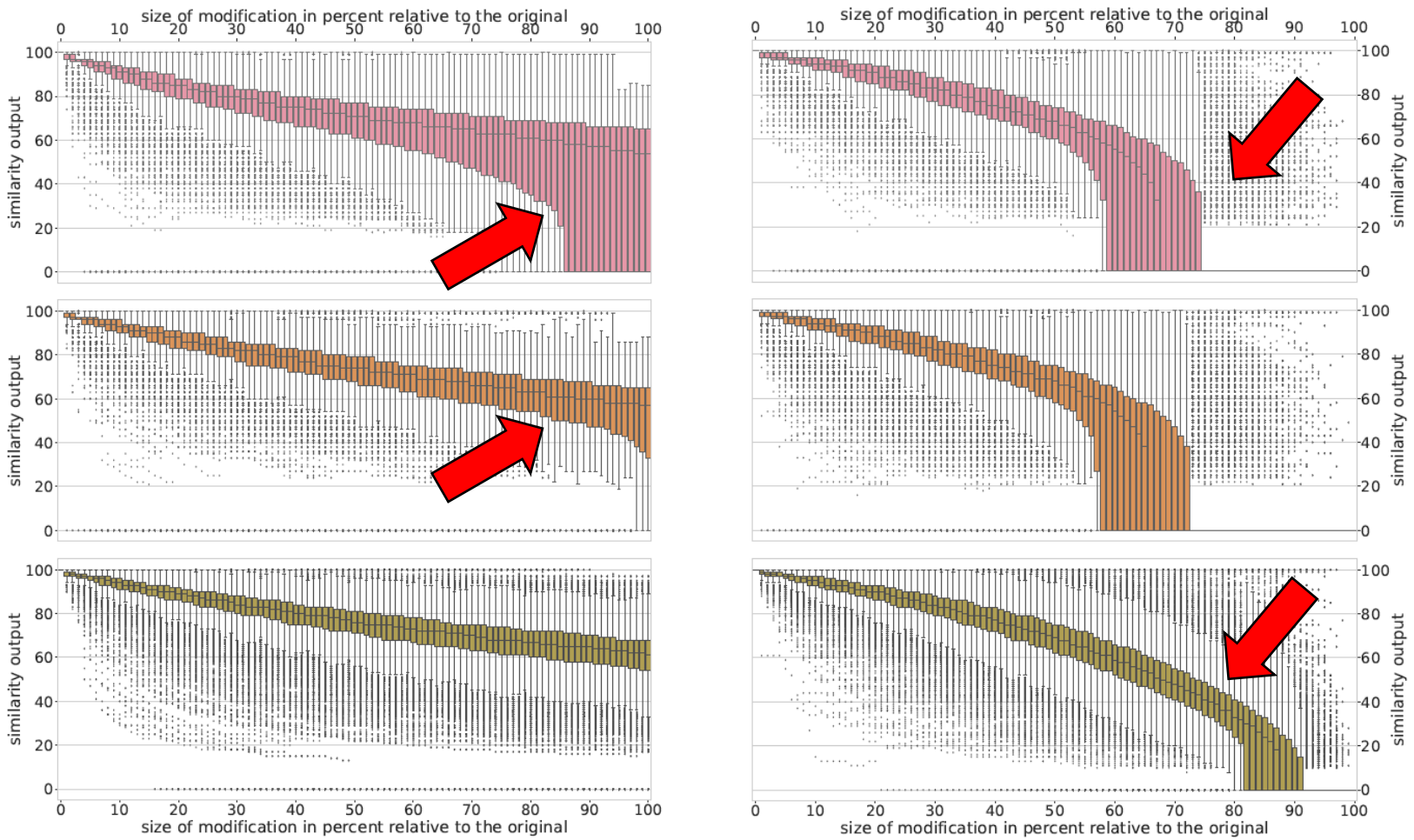
real data step-by-step corpus

- How to define accuracy and similarity?
  - The smallest error?
  - Here, variance & reliability

Evaluation – Accuracy – Real data

Insertion of bytes

Deletion of bytes



SPHF & CF

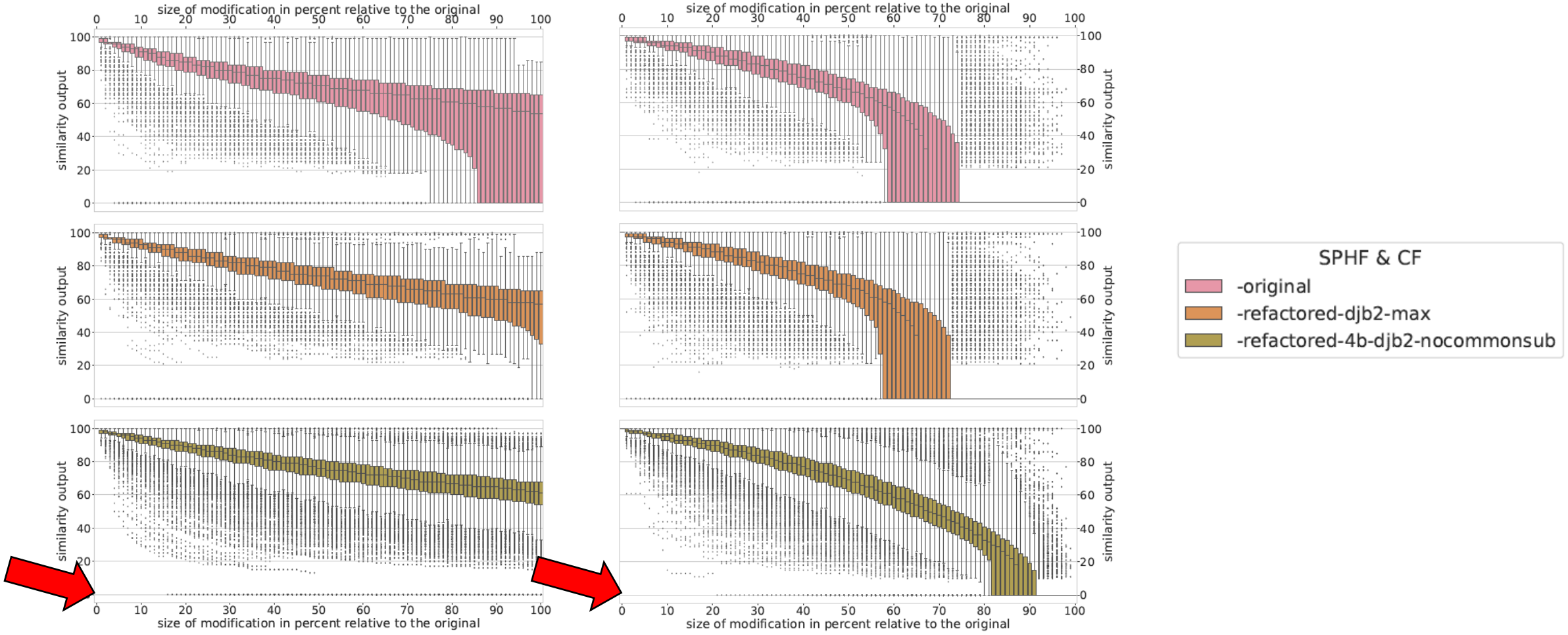
- -original
- -refactored-djb2-max
- -refactored-4b-djb2-nocommonsub

Fraunhofer FKIE

Evaluation – Accuracy – Real data

Insertion of bytes

Deletion of bytes



Fraunhofer FKIE

Evaluation – Security

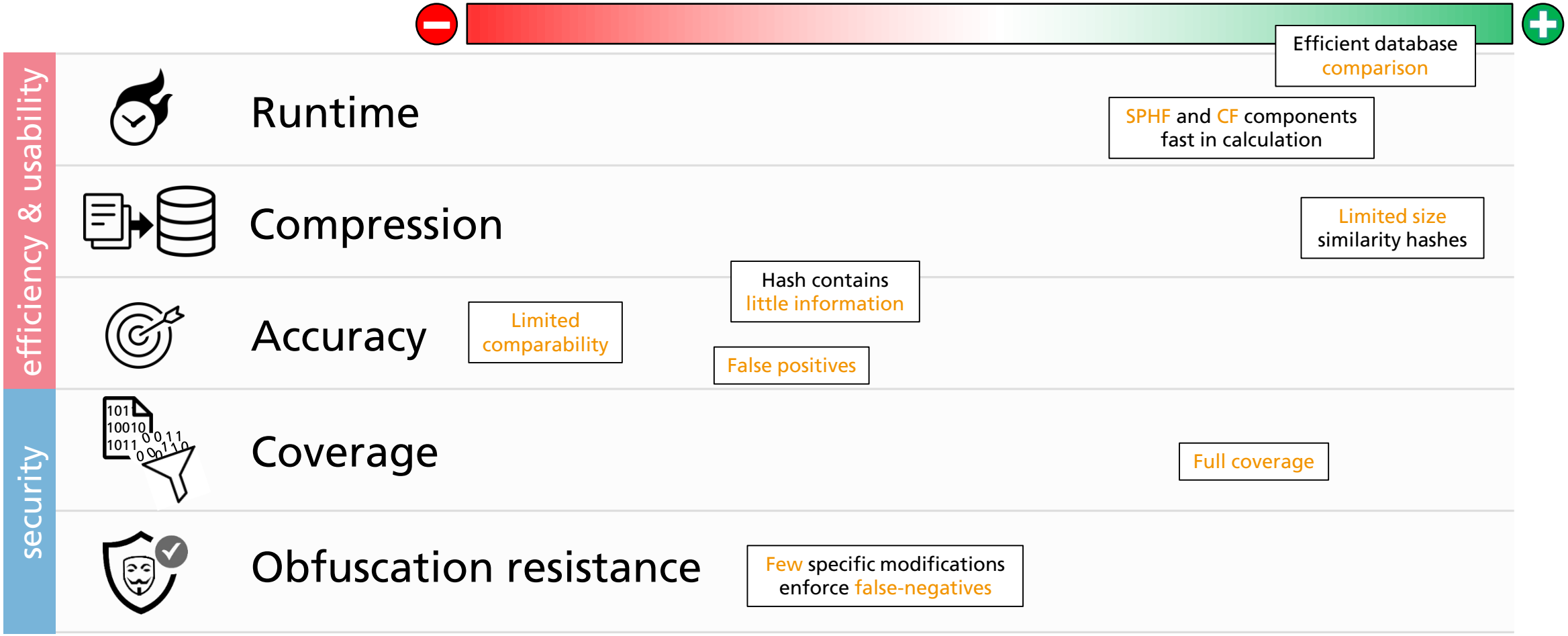
- Obfuscation resistance
  - Possibilities
    - Insertion of trigger points
    - Removal of common substrings
    - …
  - `-refactored-4b-djb2-nocommonsub`
    - Longer hash values, yet not really secure

Evaluation – Security

- Obfuscation resistance
  - Possibilities
    - Insertion of trigger points
    - Removal of common substrings
    - …
  - `-refactored-4b-djb2-nocommonsub`
    - Longer hash values, yet not really secure

- Coverage
  - `-bugfix` & `-refactored`

Evaluation – Summary – ssdeep



Fraunhofer FKIE

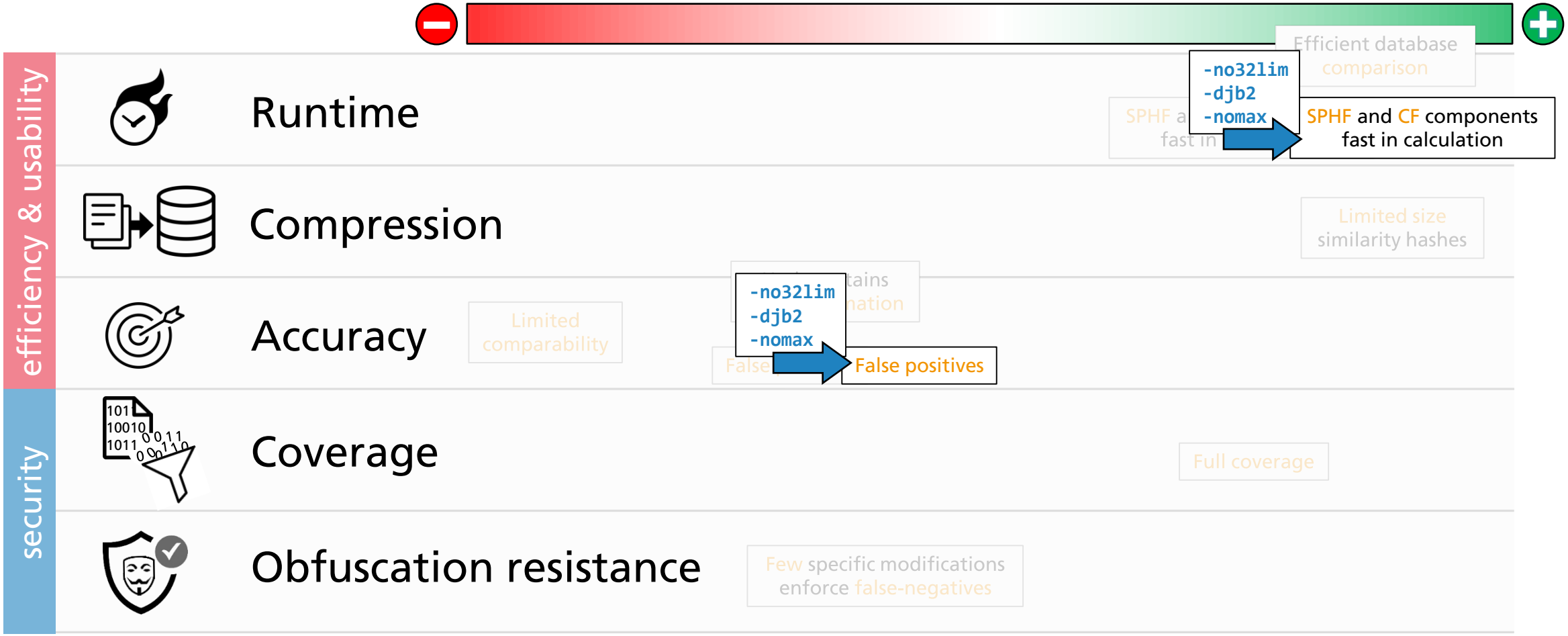
Evaluation – Summary – `ssdeep-refactored-djb2-nomax`

Fraunhofer FKIE

Evaluation – Summary – `ssdeep-refactored-djb2-nomax`

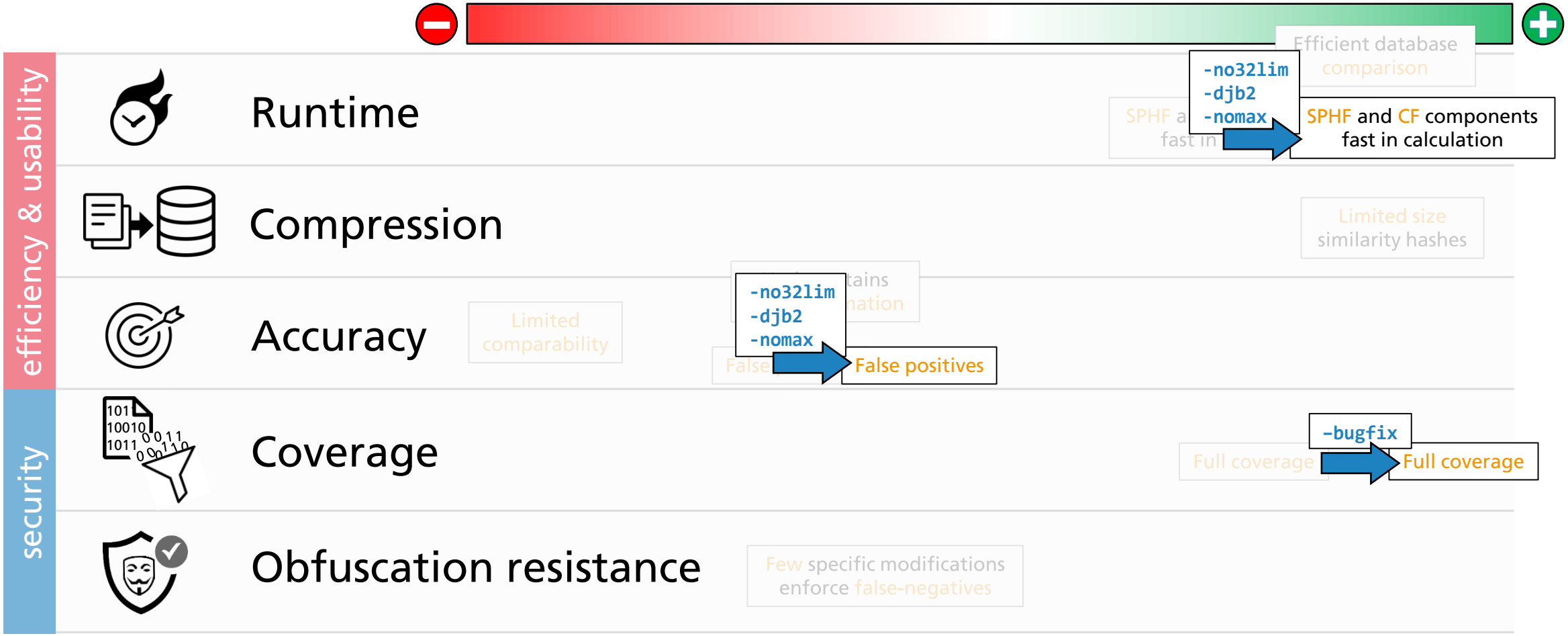
Fraunhofer FKIE

Evaluation – Summary –
ssdeep-refactored-djb2-nomax



Fraunhofer FKIE

Evaluation – Summary – `ssdeep-refactored-djb2-nomax`



Fraunhofer FKIE

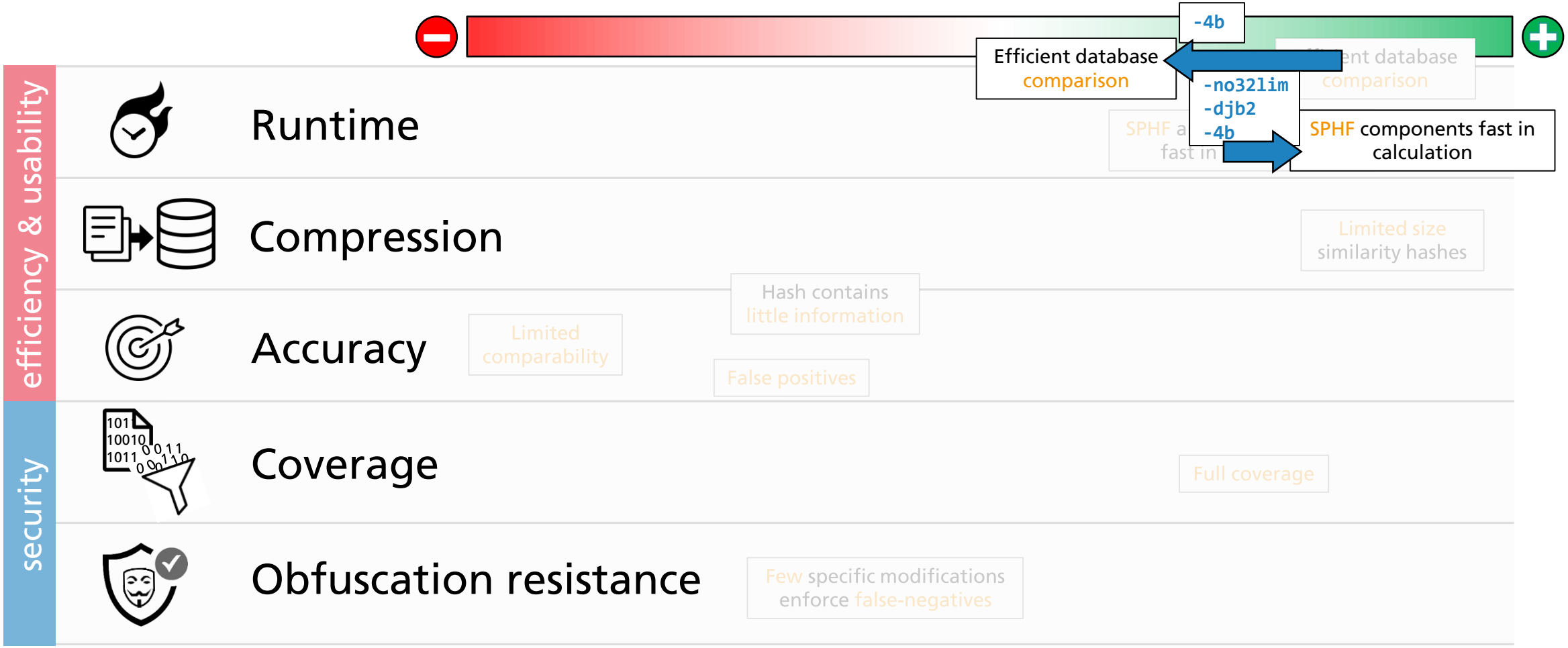
Evaluation – Summary –
ssdeep-refactored-4b-djb2-nocommonsub

Fraunhofer FKIE

Evaluation – Summary –
ssdeep-refactored-4b-djb2-nocommonsub

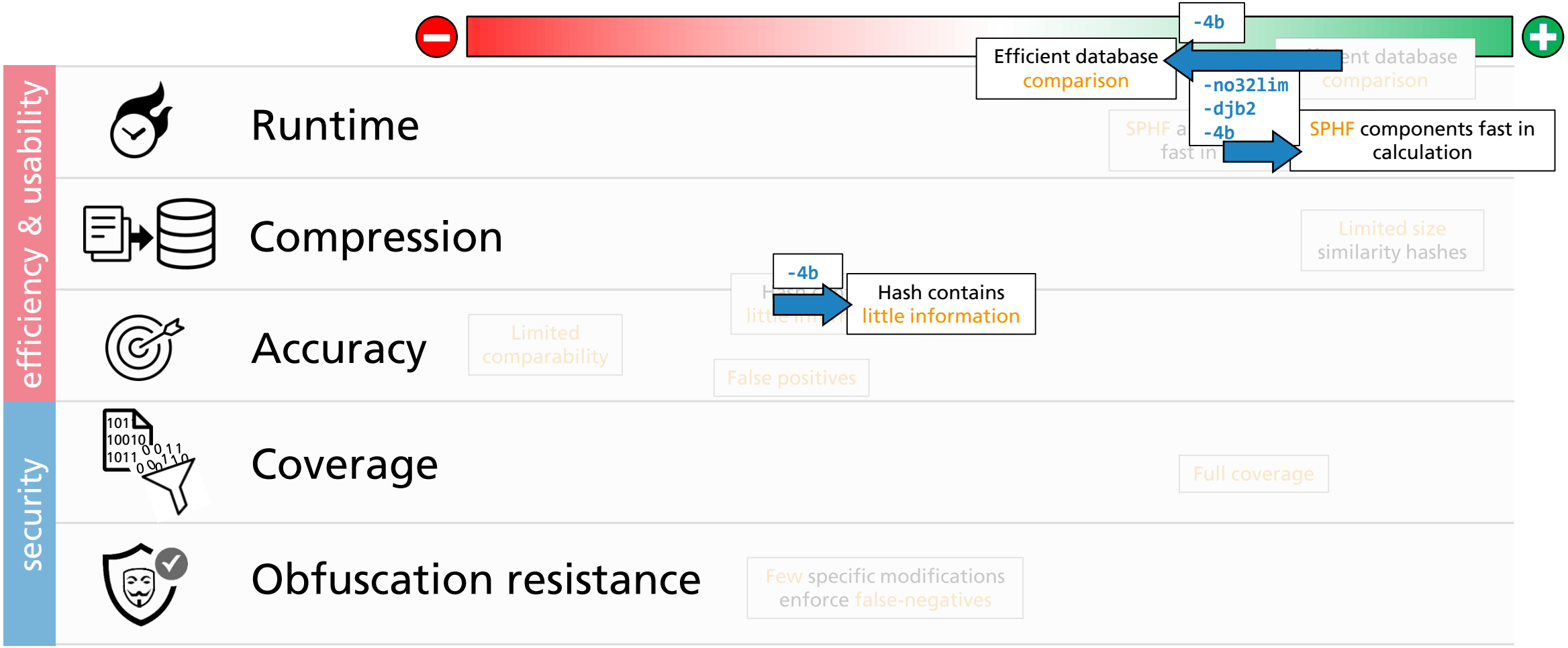
Fraunhofer FKIE

Evaluation – Summary –
ssdeep-refactored-4b-djb2-nocommonsub



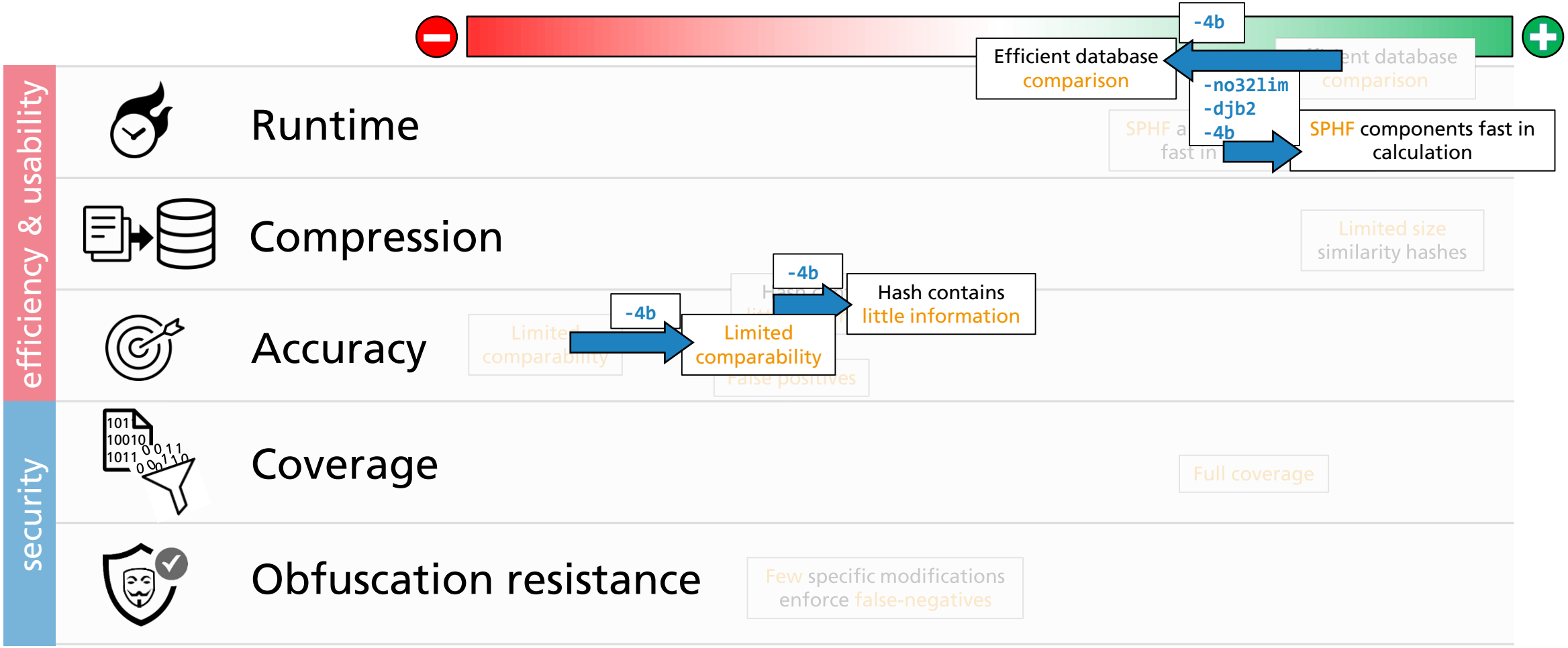
Fraunhofer FKIE

Evaluation – Summary –
ssdeep-refactored-4b-djb2-nocommonsub



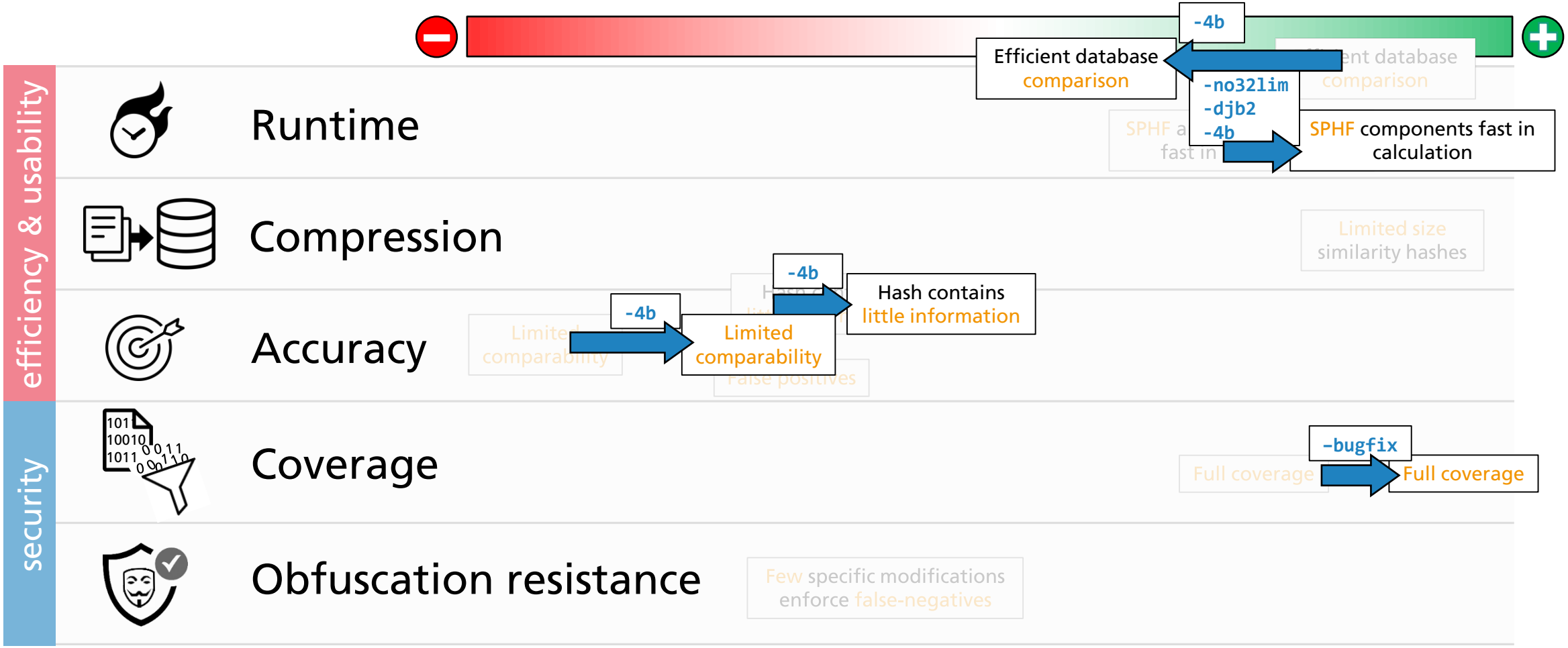
Fraunhofer FKIE

Evaluation – Summary –
ssdeep-refactored-4b-djb2-nocommonsub



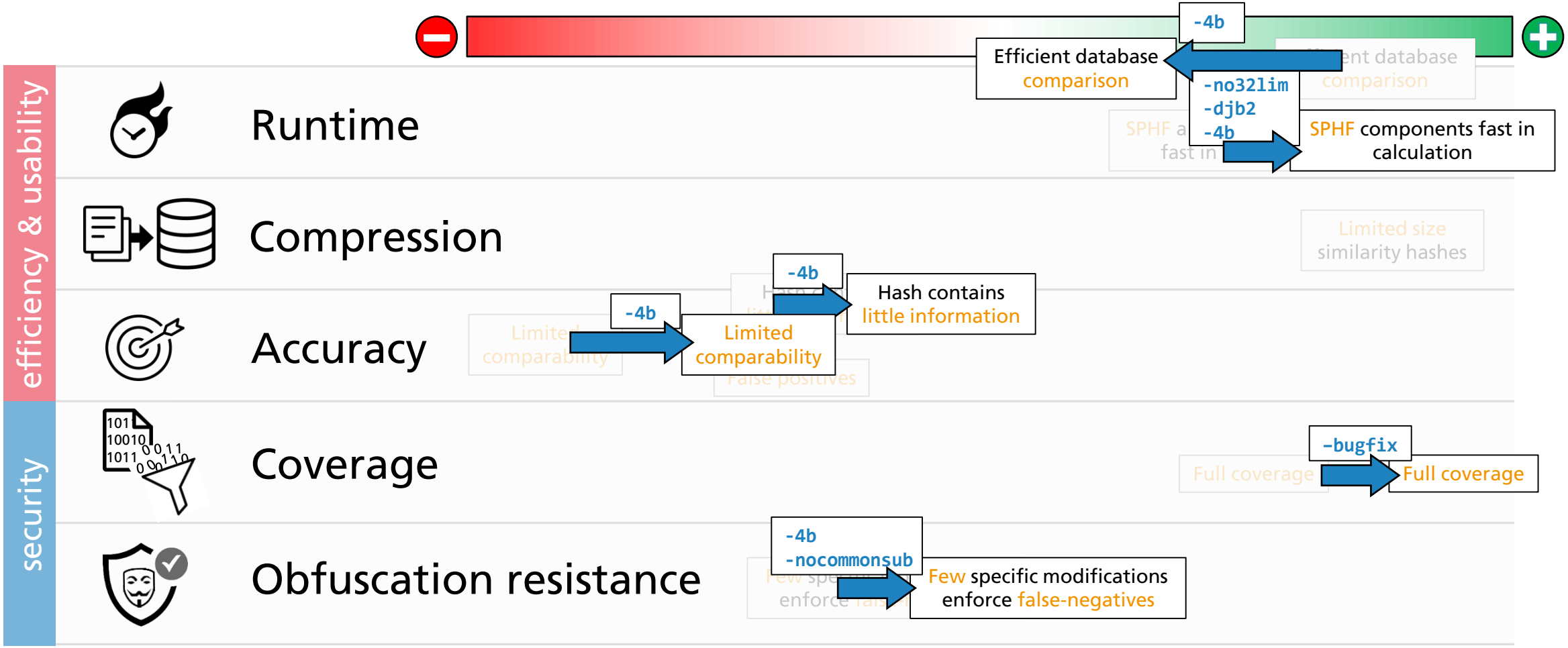
Fraunhofer FKIE

Evaluation – Summary –
ssdeep-refactored-4b-djb2-nocommonsub



Fraunhofer FKIE

Evaluation – Summary –
ssdeep-refactored-4b-djb2-nocommonsub



Fraunhofer FKIE

Conclusion

- Assessment and Evaluation of
  - ssdeep
  - proposed modifications
- Publicly available:
  - Modifications and Implementations
  - Data corpus
  - Evaluation results and scripts
  - Files highlighting issues

- `ssdeep-refactored-djb2-nomax`
  - More efficient SPHF and CF
  - Fewer false positives
  - Use case: Filtering and minimizing extraneous data

- `ssdeep-refactored-4b-djb2-nocommonsub`
  - Slower but larger range of comparisons
  - More efficient SPHF
  - Fewer false negatives
  - Use case: Detection of modified suspicious files

Thank you!

Any Questions?

Contact Information

Carlo Jakobs
carlo.jakobs@fkie.fraunhofer.de

Martin Lambertz
martin.lambertz@fkie.fraunhofer.de

Jan-Niclas Hilgert
jan-niclas.hilgert@fkie.fraunhofer.de

Code and data

https://github.com/fkie-cad/ssdeeper

Fraunhofer FKIE