Automation of the Reverse Engineering of Unknown Binary Network Protocols

Dissertation Defense

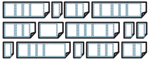
Stephan Kleber
Institute of Distributed Systems

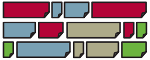
April 26, 2024

Definition of Protocol Reverse Engineering

PROTOCOL REVERSE ENGINEERING (PRE)
is the process of *inferring* the

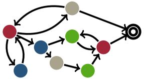
message formats,





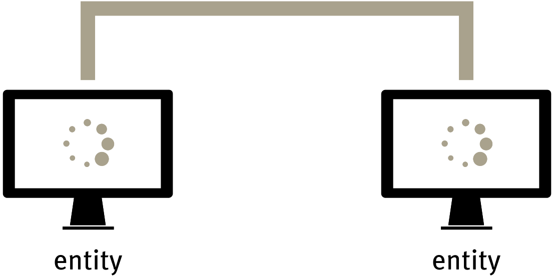
message types, and

grammar

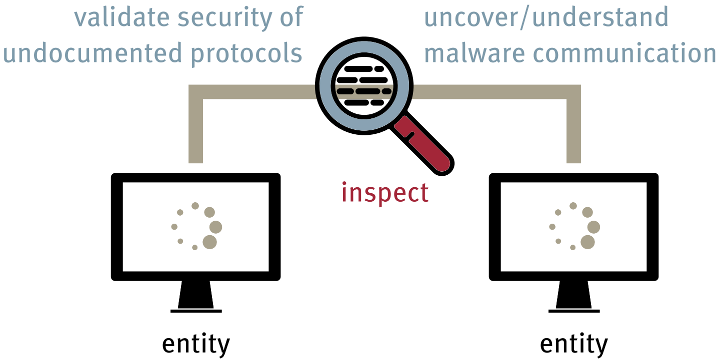


of a network protocol
for which a formal *specification is unknown*.

Motivation for Protocol Reverse Engineering



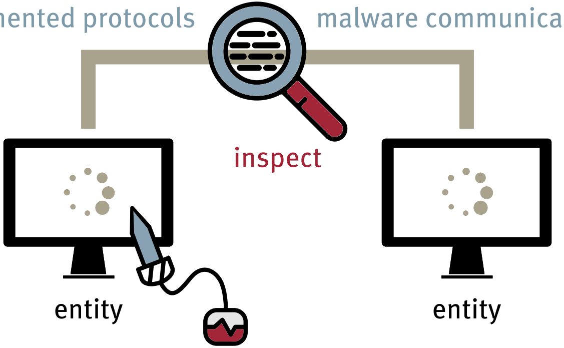
Motivation for Protocol Reverse Engineering



Motivation for Protocol Reverse Engineering

validate security of
undocumented protocols

uncover/understand
malware communication



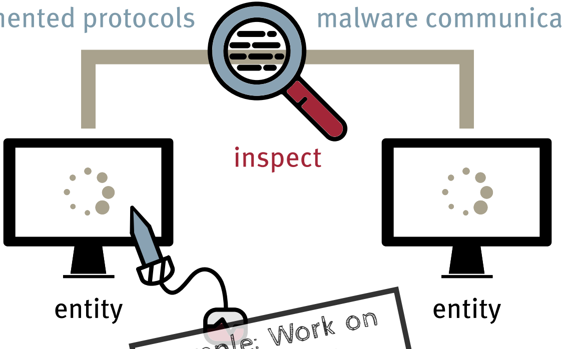
probe the implementation

Smart Fuzzing to discover vulnerabilities

Motivation for Protocol Reverse Engineering

validate security of undocumented protocols

uncover/understand malware communication

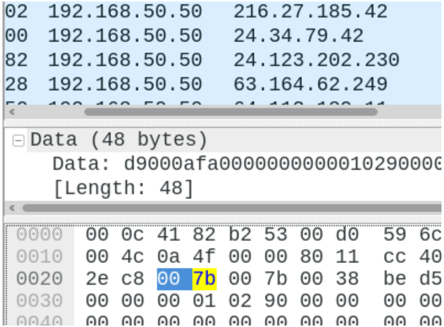


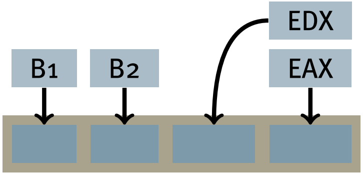
Example: Work on Apple protocols

probe the implementation

Smart Fuzzing to discover vulnerabilities

Methods of Protocol Reverse Engineering





Message TX/RX Buffer

```
MOVSB [0x1000], [0xff00]
MOVSB [0x1001], [0xff01]
MOVSB [0x1002], EDX
MOVSB [0x1003], EAX
```

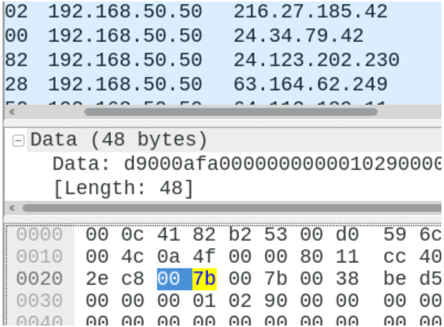
Static Traffic Analysis

Recording observable transmission of data

Dynamic Entity Analysis

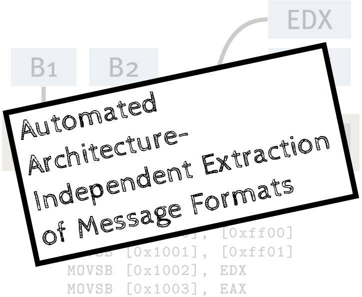
Source code or binary program of entities

Methods of Protocol Reverse Engineering



Static Traffic Analysis

Recording observable transmission of data



Dynamic Entity Analysis

Source code or binary program of entities

Types of Protocols

Field Boundaries of...

... textual protocols (e. g., SMTP):

RCPT | | TO: | | <twanda@blue6.ex> |

... binary protocols (e. g., DHCP):

63 82 53 63 | 35 01 | 05 36 04 ac 14 03 01 | 33 04 00 00 0e 10

Keyword

Separator

Value

Types of Protocols

Field Boundaries of...

... textual protocols (e. g., SMTP):

RCPT		TO:		<twanda@blue6.ex>	

... binary protocols (e. g., DHCP):

63 82 53 63	35 01	05 36 04 ac 14 03 01	33 04 00 00 0e 10

Keyword

Separator

Value

Types of Protocols

Field Boundaries of...

... textual protocols (e. g., SMTP):

RCPT		TO:		<twanda@blue6.ex>	

... binary protocols (e. g., DHCP):

63 82 53 63	35 01	05 36 04 ac 14 03 01	33 04 00 00 0e 10

Keyword

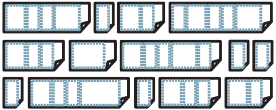
Separator

Value

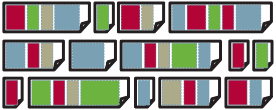
STATIC TRAFFIC ANALYSIS OF
UNKNOWN BINARY NETWORK PROTOCOLS

Targets of Protocol Reverse Engineering

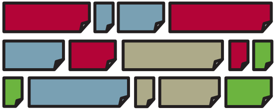
Protocol Specification



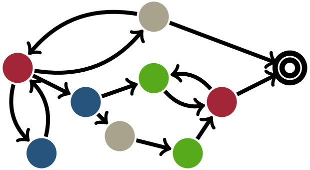
Message Formats | Fields



Field Data Types | Semantic



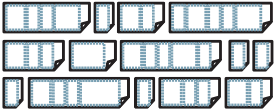
Message Types | Vocabulary



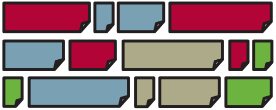
Behavior Model | Grammar

Targets of Protocol Reverse Engineering

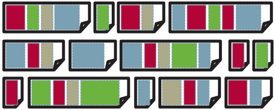
Protocol Specification



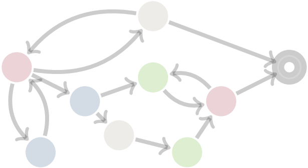
Message Formats | Fields



Message Types | Vocabulary



Field Data Types | Semantic



Behavior Model | Grammar

Static Traffic Analysis: Related Work Survey

Discoverer: Message types by segmentation of textual message parts.
Weidong Cui et al., USENIX Security 2007.

PRISMA: Message types and behavior using Markov models.
Tammo Krueger et al., AISec 2012.

Netzob: Message types and formats by aligning identical bytes in messages.
Georges Bossert et al., CCS 2014.

FieldHunter: Identify few specific field types within messages.
Ignacio Bermudez et al., COMCOM 84 (2016).

Contiguous Sequential Pattern: Recursive inference by frequency analysis.
Y.-H. Goo et al., IEEE Access, vol 7 (2019).

Static Traffic Analysis: Related Work Survey

Discoverer: Message types by segmentation of textual message parts.
Weidong Cui et al., USENIX Security 2007.

PRISMA: Message types and behavior using Markov models.
Tammo Krueger et al., AISec 2012.

Netzob: Message types and formats by aligning identical bytes in messages.
Georges Bossert et al., CCS 2014.

FieldHunter: Identify few specific field types within messages.
Ignacio Bermudez et al., COMCOM 84 (2016).

Contiguous Sequential Pattern: Recursive inference by frequency analysis.
Y.-H. Goo et al., IEEE Access, vol 7 (2019).

Static Traffic Analysis: Related Work Survey

Discoverer: Message types by segmentation of textual message parts.
Weidong Cui et al., USENIX Security 2007.

PRISMA: Message types and behavior using Markov models.
Tammo Krueger et al., AISec 2012.

Netzob: Message types and formats by aligning identical bytes in messages.
Georges Bossert et al., CCS 2014.

FieldHunter: Identify few specific field types within messages.
Ignacio Bermudez et al., COMCOM 84 (2016).

Contiguous Sequential Pattern: Recursive inference by frequency analysis.
Y.-H. Goo et al., IEEE Access, vol 7 (2019).

Static Traffic Analysis: Overcoming Limitations of Related Work[1]

Common limitations of related work: many specific assumptions about protocols

[1] Stephan Kleber et al. „Survey of Protocol Reverse Engineering Algorithms: Decomposition of Tools for Static Traffic Analysis“. In: *IEEE Communications Surveys and Tutorials* 21.1 (Feb. 2019). Firstquarter.

Static Traffic Analysis: Overcoming Limitations of Related Work[1]

Common limitations of related work: many specific assumptions about protocols

In contrast, make few assumptions: generically applicable approach

---

[1] Stephan Kleber et al. „Survey of Protocol Reverse Engineering Algorithms: Decomposition of Tools for Static Traffic Analysis“. In: *IEEE Communications Surveys and Tutorials* 21.1 (Feb. 2019). Firstquarter.

Static Traffic Analysis: Overcoming Limitations of Related Work[1]

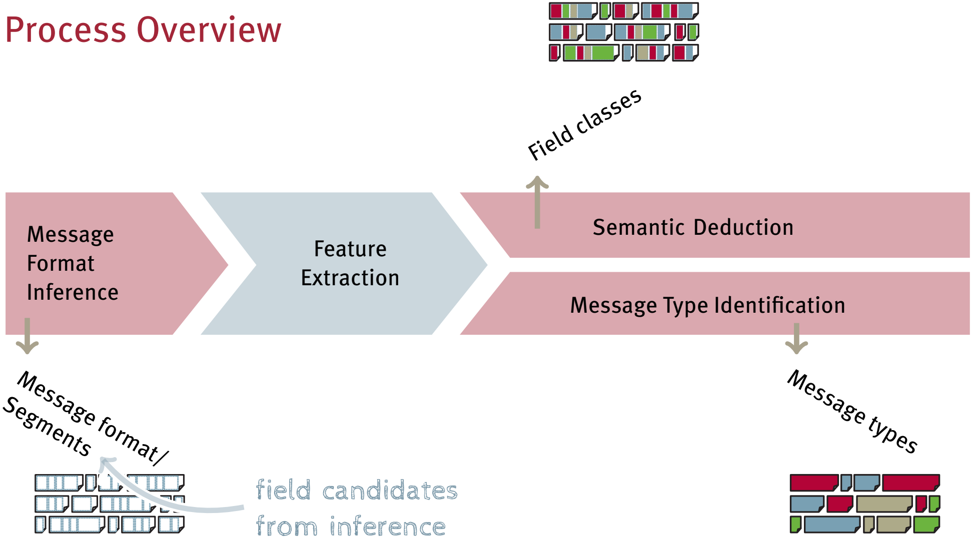
Common limitations of related work: many specific assumptions about protocols

In contrast, make few assumptions: generically applicable approach

- No specific message format or protocol structure
- No preceding classification of messages into types or flows
- No meta-data/encapsulation required

[1] Stephan Kleber et al. „Survey of Protocol Reverse Engineering Algorithms: Decomposition of Tools for Static Traffic Analysis". In: *IEEE Communications Surveys and Tutorials* 21.1 (Feb. 2019). Firstquarter.

Process Overview



Process Overview[1]

Preparation

Data Collection and Preprocessing

Message Format Inference

Feature Extraction

Semantic Deduction

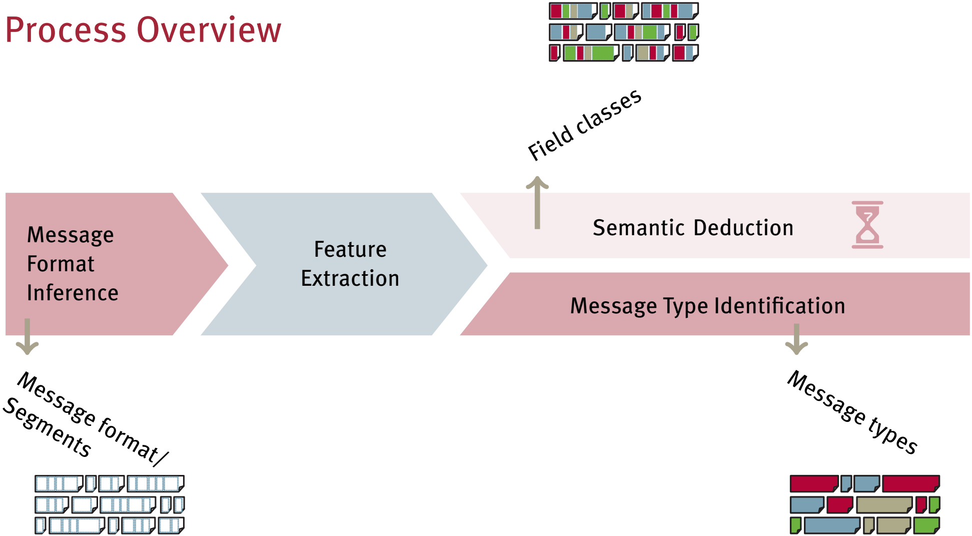
Message Type Identification

Behavior Model Reconstruction

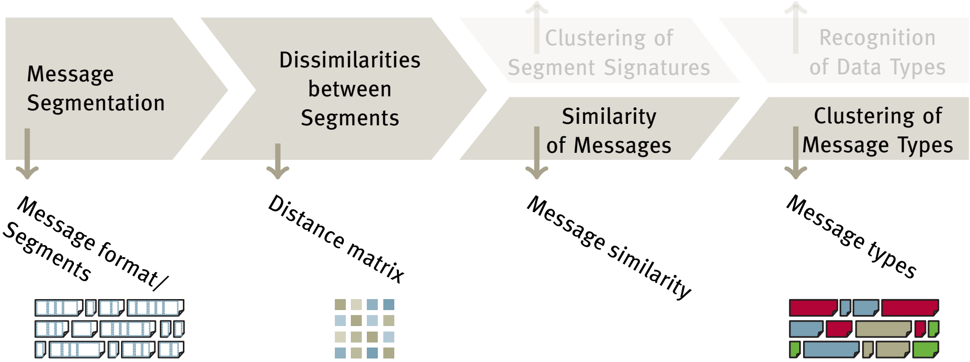
Processing of Results

[1] Stephan Kleber et al. „Survey of Protocol Reverse Engineering Algorithms: Decomposition of Tools for Static Traffic Analysis". In: *IEEE Communications Surveys and Tutorials* 21.1 (Feb. 2019). Firstquarter.

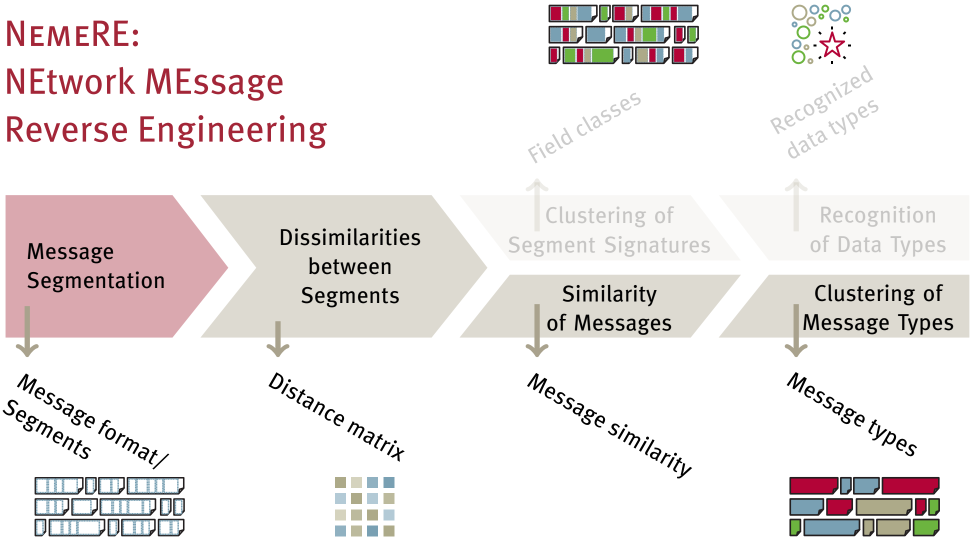
Process Overview



NemeRE:
NEtwork MEssage
Reverse Engineering



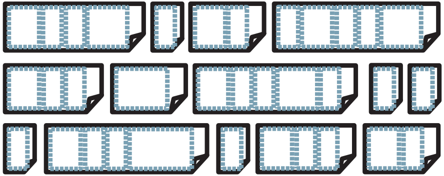
NemeRE:
NEtwork MEssage
Reverse Engineering



NEtwork MEssage SYntax analysiS

NEMESYS[1]: heuristic message segmentation

- Analyze each and every message individually
- Efficient heuristic for characteristics of substructures
- Intrinsic message structure
- Find probable field boundaries



[1] Stephan Kleber et al. „NEMESYS: Network Message Syntax Reverse Engineering by Analysis of the Intrinsic Structure of Individual Messages". In: *Proceedings of the 12th USENIX Workshop on Offensive Technologies*. WOOT. USENIX Association, 2018.

NEMESYS: Deltas of Bit Congruence

Bit Congruence: Comparison of bit-wise match of two subsequent bytes

Deltas of Bit Congruence:
Difference in the congruence of two pairs of subsequent byte values

NEMESYS: Deltas of Bit Congruence

Bit Congruence: Comparison of bit-wise match of two subsequent bytes

Deltas of Bit Congruence:
Difference in the congruence of two pairs of subsequent byte values

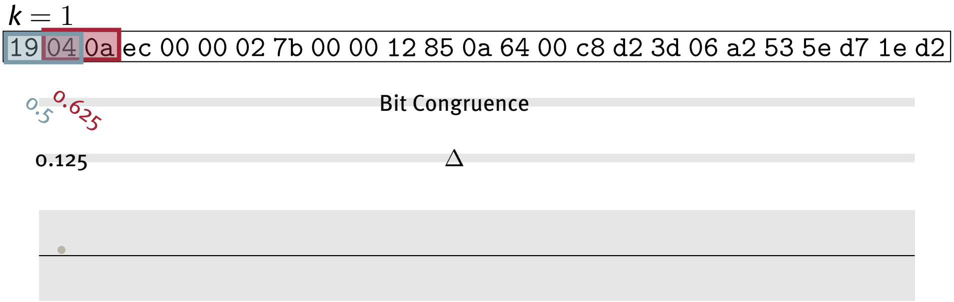
```
19 04 0a ec 00 00 02 7b 00 00 12 85 0a 64 00 c8 d2 3d 06 a2 53 5e d7 1e d2
```

Message of 25 bytes in hexadecimals

NEMESYS: Deltas of Bit Congruence

Bit Congruence: Comparison of bit-wise match of two subsequent bytes

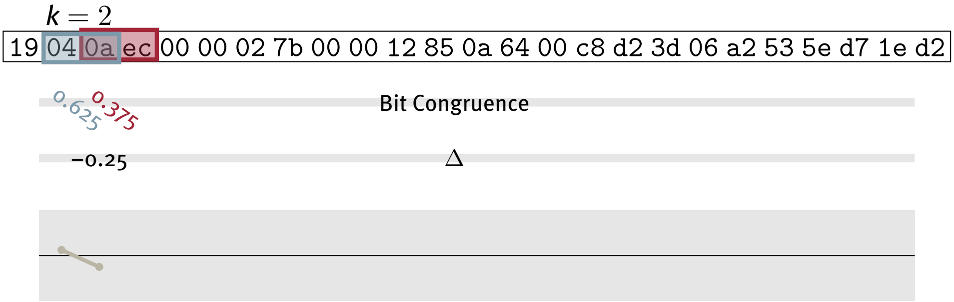
Deltas of Bit Congruence:
Difference in the congruence of two pairs of subsequent byte values



NEMESYS: Deltas of Bit Congruence

Bit Congruence: Comparison of bit-wise match of two subsequent bytes

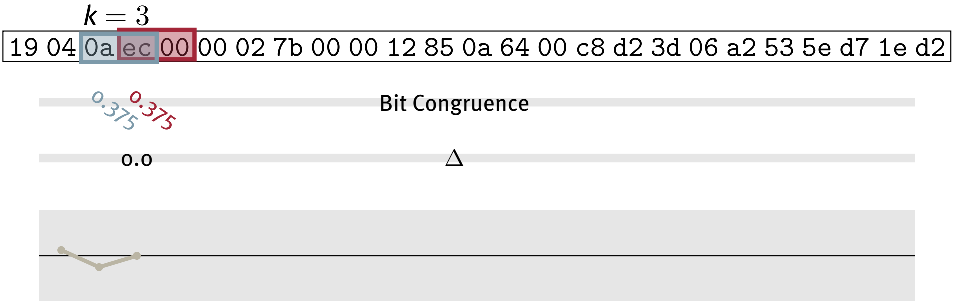
Deltas of Bit Congruence:
Difference in the congruence of two pairs of subsequent byte values



NEMESYS: Deltas of Bit Congruence

Bit Congruence: Comparison of bit-wise match of two subsequent bytes

Deltas of Bit Congruence:
Difference in the congruence of two pairs of subsequent byte values



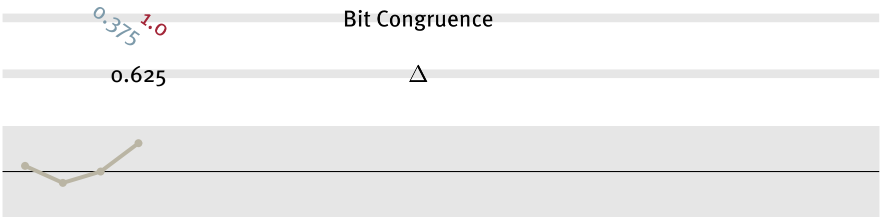
NEMESYS: Deltas of Bit Congruence

Bit Congruence: Comparison of bit-wise match of two subsequent bytes

Deltas of Bit Congruence:
Difference in the congruence of two pairs of subsequent byte values

$k = 4$

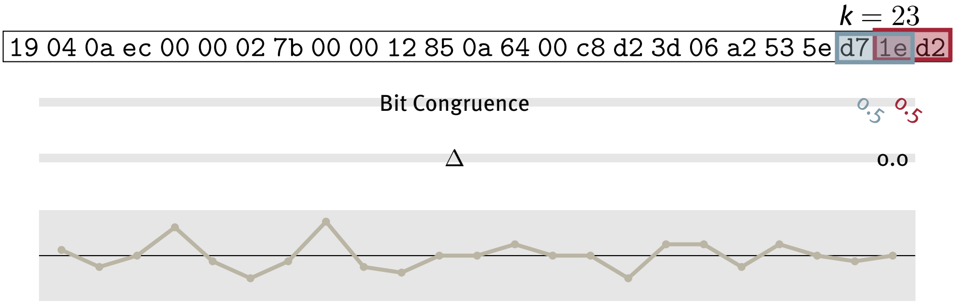
19 04 0a ec 00 00 02 7b 00 00 12 85 0a 64 00 c8 d2 3d 06 a2 53 5e d7 1e d2



NEMESYS: Deltas of Bit Congruence

Bit Congruence: Comparison of bit-wise match of two subsequent bytes

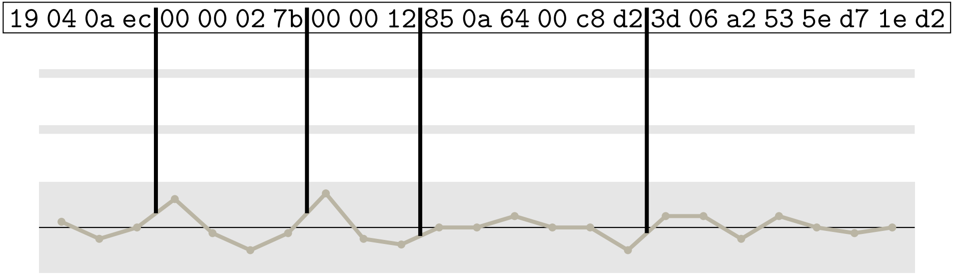
Deltas of Bit Congruence:
Difference in the congruence of two pairs of subsequent byte values



NEMESYS: Deltas of Bit Congruence

Bit Congruence: Comparison of bit-wise match of two subsequent bytes

Deltas of Bit Congruence:
Difference in the congruence of two pairs of subsequent byte values



Refinement by Principal Component Analysis (PCA)[1]

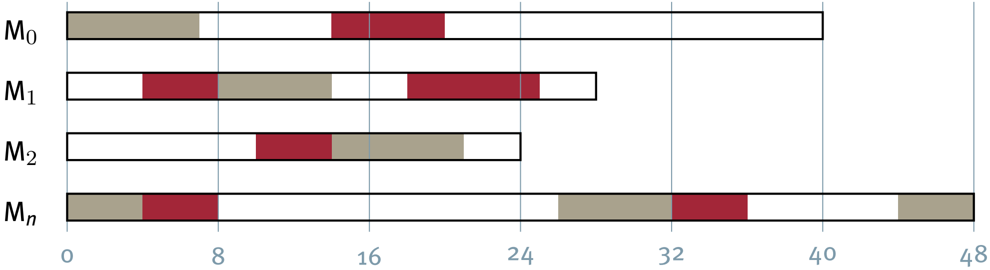
NEMESYS suffers from frequent off-by-one errors in field boundaries

Correct NEMESYS errors using Principal Component Analysis:

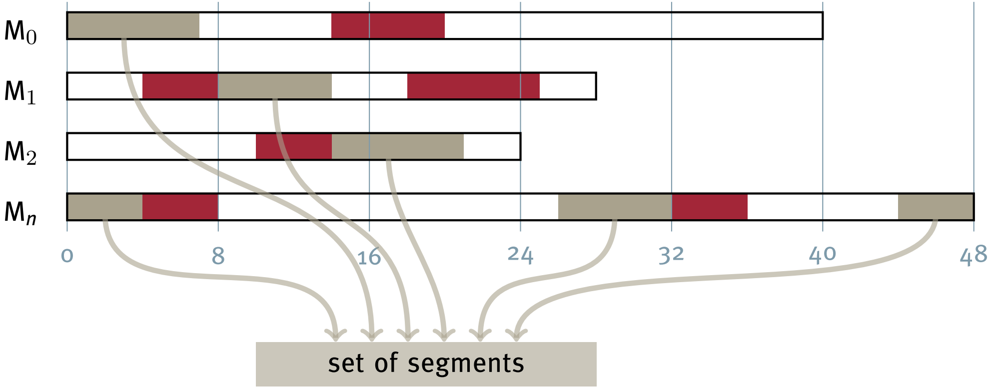
- Variance-locked bytes typically comprise one field
- Principal Component Analysis quantifies multivariate variance
- Basis: covariance matrix $\mathbf{C}$ of the data matrix $\mathbf{X}$

[1] Stephan Kleber and Frank Kargl. „Refining Network Message Segmentation with Principal Component Analysis“. In: *Proceedings of the tenth annual IEEE Conference on Communications and Network Security*. CNS. IEEE, 2022.

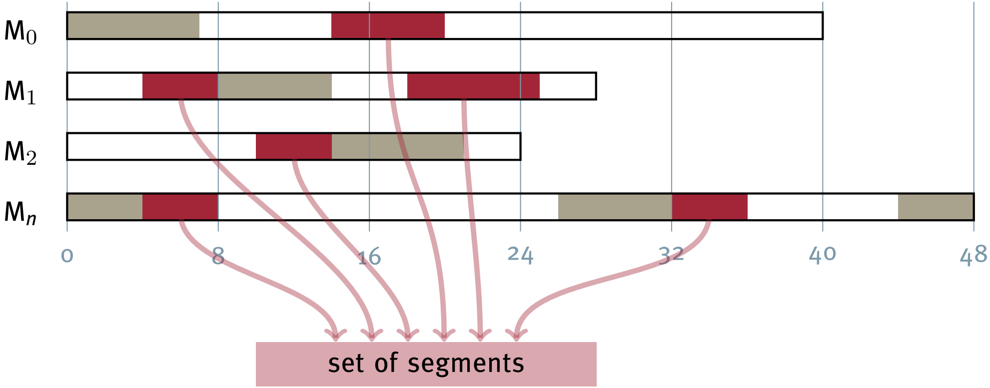
Recursive Clustering: Collecting Similar Segments



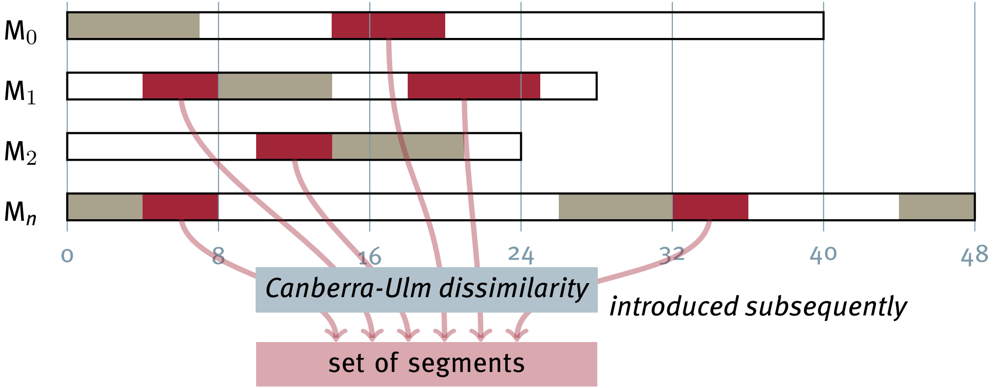
Recursive Clustering: Collecting Similar Segments



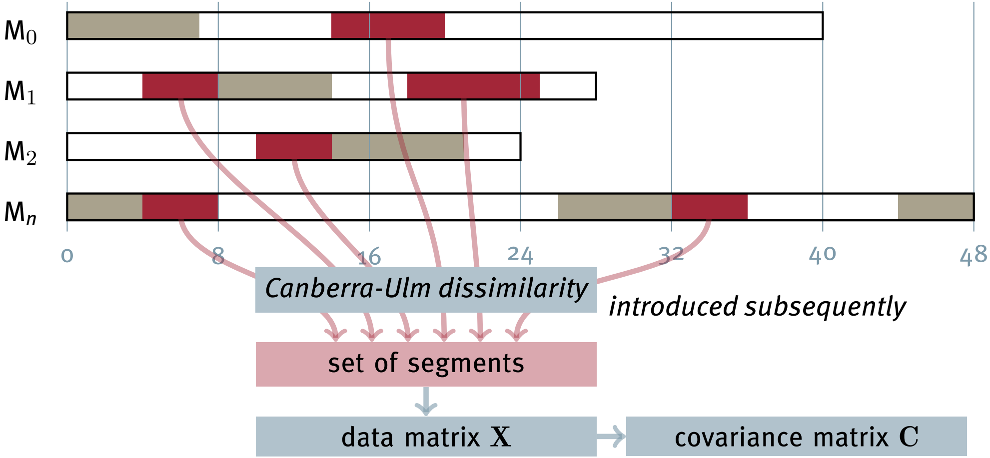
Recursive Clustering: Collecting Similar Segments



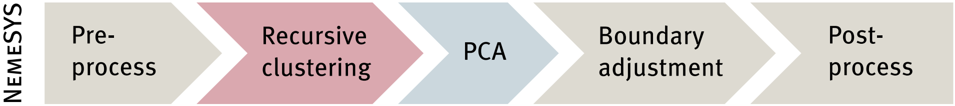
Recursive Clustering: Collecting Similar Segments



Recursive Clustering: Collecting Similar Segments



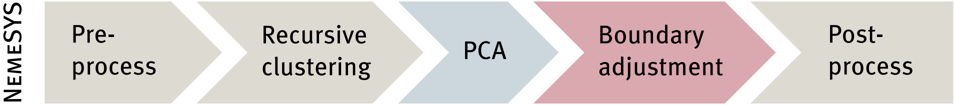
Refinement of NEMESYS: Byte-wise Segment Variance Analysis



Recursive clustering:

- Ensures application of PCA to a set of related segments

Refinement of NEMESYS: Byte-wise Segment Variance Analysis



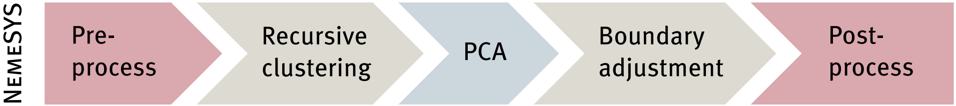
Recursive clustering:

- Ensures application of PCA to a set of related segments

Boundary adjustment:

- Heuristic rules for field boundary adjustments, e. g., sharp variance drops

Refinement of NEMESYS: Byte-wise Segment Variance Analysis



Recursive clustering:

- Ensures application of PCA to a set of related segments

Boundary adjustment:

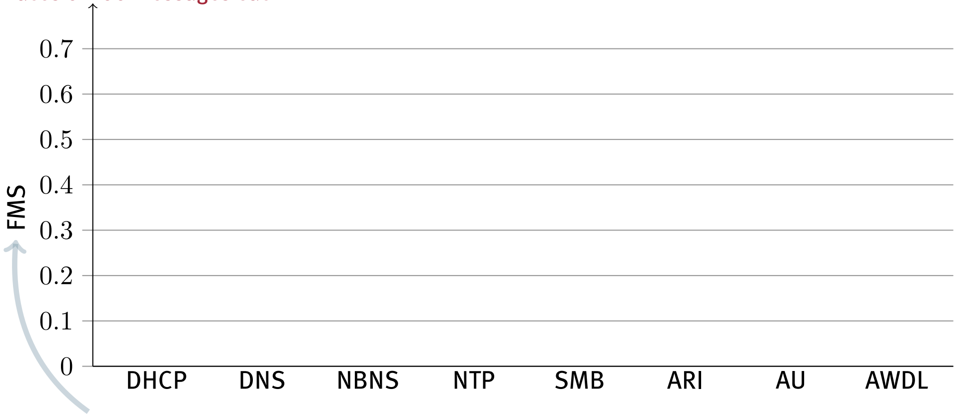
- Heuristic rules for field boundary adjustments, e. g., sharp variance drops

Static-rule pre- and post-processing:

- Merging of segments with similar local entropy
- Accommodate embedded text by character segment refinement

Evaluation of NEMESYS Segmentation

Traces of 100 messages each

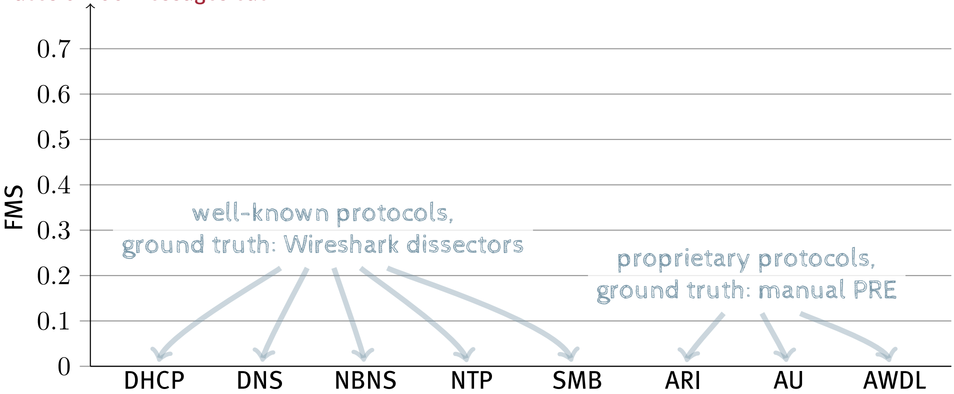


Format Match Score (FMS)[1]: Conformance of inference to message format specification

[1] Stephan Kleber et al. „NEMESYS: Network Message Syntax Reverse Engineering by Analysis of the Intrinsic Structure of Individual Messages“. In: *Proceedings of the 12th USENIX Workshop on Offensive Technologies*. WOOT. USENIX Association, 2018.

Evaluation of NEMESYS Segmentation

Traces of 100 messages each



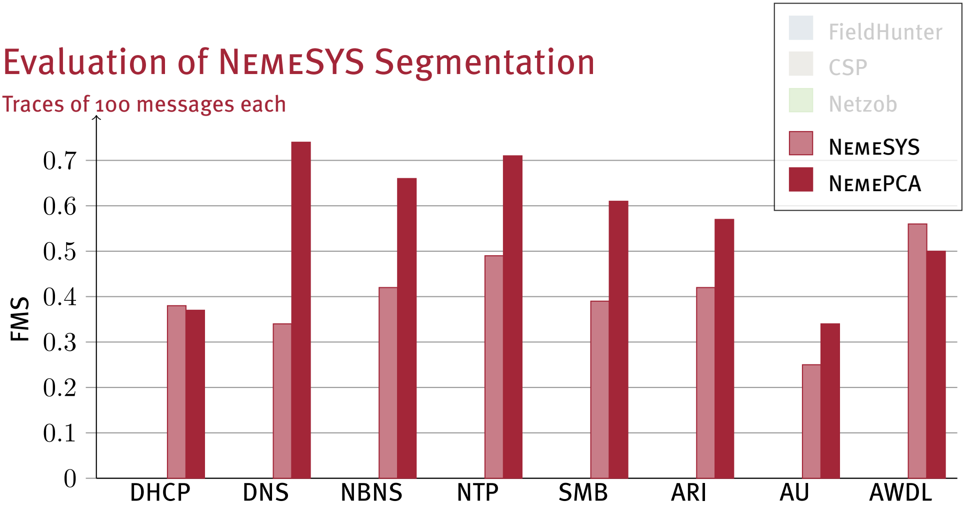
Format Match Score (FMS)[1]: Conformance of inference to message format specification

---

[1] Stephan Kleber et al. „NEMESYS: Network Message Syntax Reverse Engineering by Analysis of the Intrinsic Structure of Individual Messages“. In: *Proceedings of the 12th USENIX Workshop on Offensive Technologies*. WOOT. USENIX Association, 2018.

Evaluation of NEMESYS Segmentation

Traces of 100 messages each

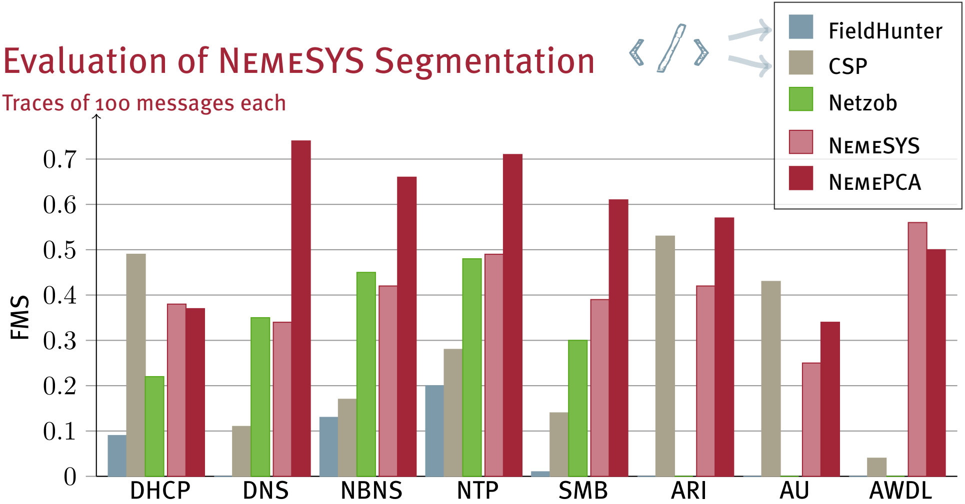


Format Match Score (FMS)[1]: Conformance of inference to message format specification

[1] Stephan Kleber et al. „NEMESYS: Network Message Syntax Reverse Engineering by Analysis of the Intrinsic Structure of Individual Messages“. In: *Proceedings of the 12th USENIX Workshop on Offensive Technologies*. WOOT. USENIX Association, 2018.

Evaluation of NEMESYS Segmentation

Traces of 100 messages each

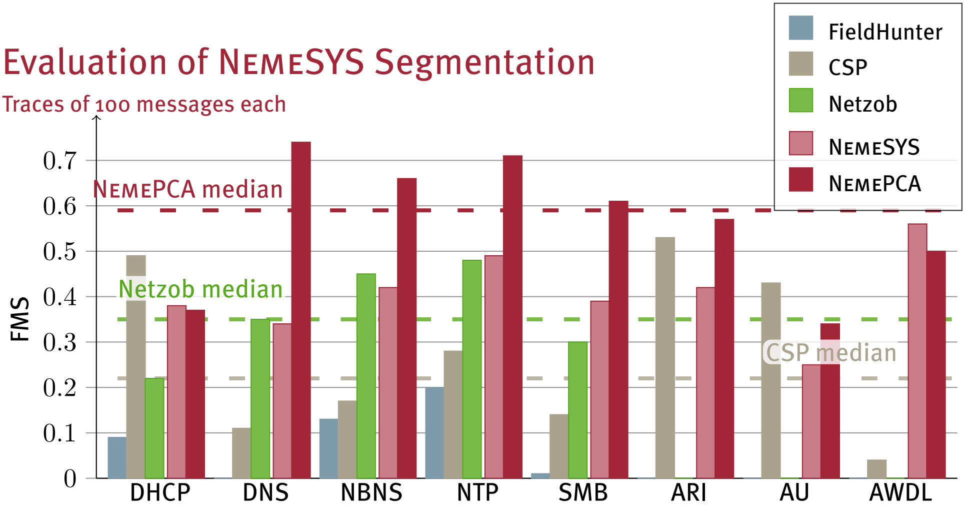


Format Match Score (FMS)[1]: Conformance of inference to message format specification

[1] Stephan Kleber et al. „NEMESYS: Network Message Syntax Reverse Engineering by Analysis of the Intrinsic Structure of Individual Messages“. In: *Proceedings of the 12th USENIX Workshop on Offensive Technologies*. WOOT. USENIX Association, 2018.

Evaluation of NEMESYS Segmentation

Traces of 100 messages each

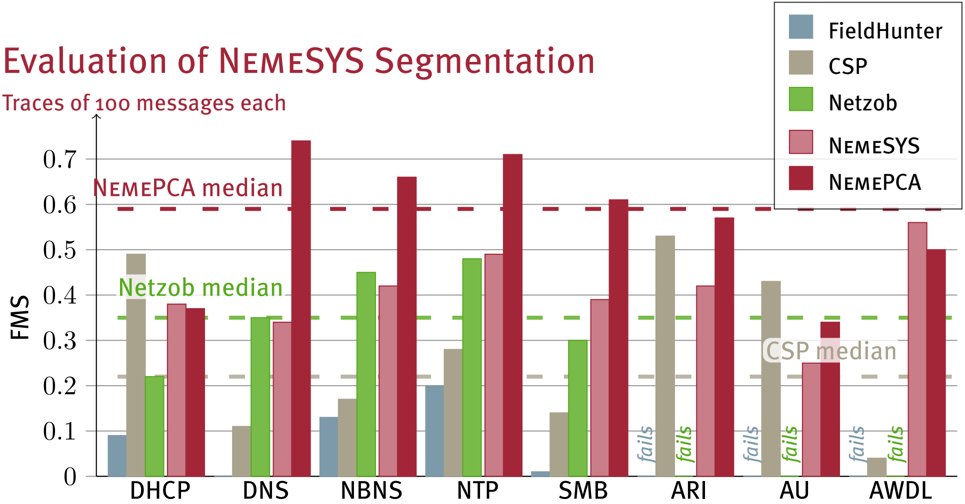


Format Match Score (FMS)[1]: Conformance of inference to message format specification

[1] Stephan Kleber et al. „NEMESYS: Network Message Syntax Reverse Engineering by Analysis of the Intrinsic Structure of Individual Messages“. In: *Proceedings of the 12th USENIX Workshop on Offensive Technologies*. WOOT. USENIX Association, 2018.

Evaluation of NEMESYS Segmentation

Traces of 100 messages each

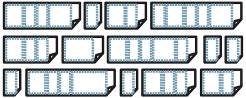


Format Match Score (FMS)[1]: Conformance of inference to message format specification

[1] Stephan Kleber et al. „NEMESYS: Network Message Syntax Reverse Engineering by Analysis of the Intrinsic Structure of Individual Messages". In: *Proceedings of the 12th USENIX Workshop on Offensive Technologies*. WOOT. USENIX Association, 2018.

Result of Message Format Inference

Segmentation into Field Candidates



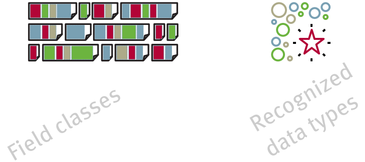
NEMESYS: NEtwork MEssage SYntax analysIS (WOOT2018)[1]

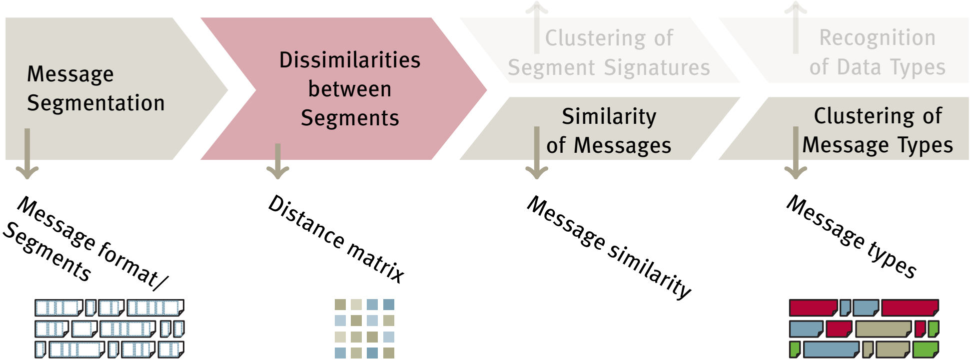
NEMEPCA: NEMESYS with PCA Refinement (CNS2022)[2]

[1] Stephan Kleber et al. „NEMESYS: Network Message Syntax Reverse Engineering by Analysis of the Intrinsic Structure of Individual Messages“. In: *Proceedings of the 12th USENIX Workshop on Offensive Technologies*. WOOT. USENIX Association, 2018.

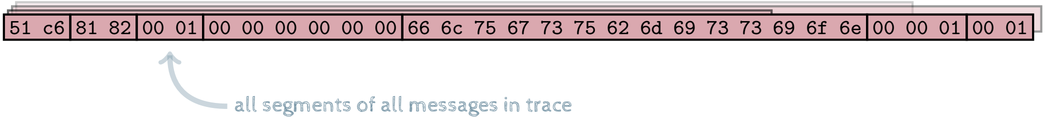
[2] Stephan Kleber and Frank Kargl. „Refining Network Message Segmentation with Principal Component Analysis“. In: *Proceedings of the tenth annual IEEE Conference on Communications and Network Security*. CNS. IEEE, 2022.

NEMERE:
NEtwork MEssage
Reverse Engineering

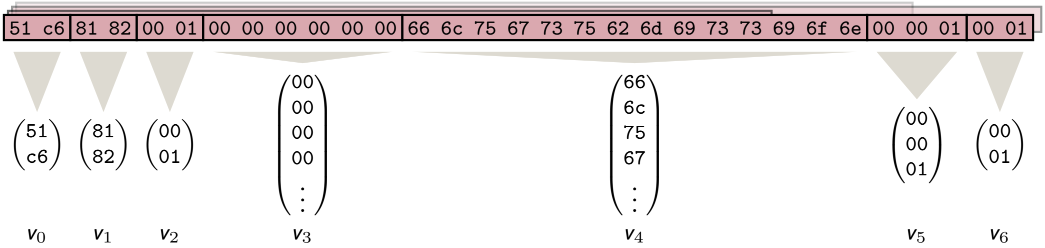




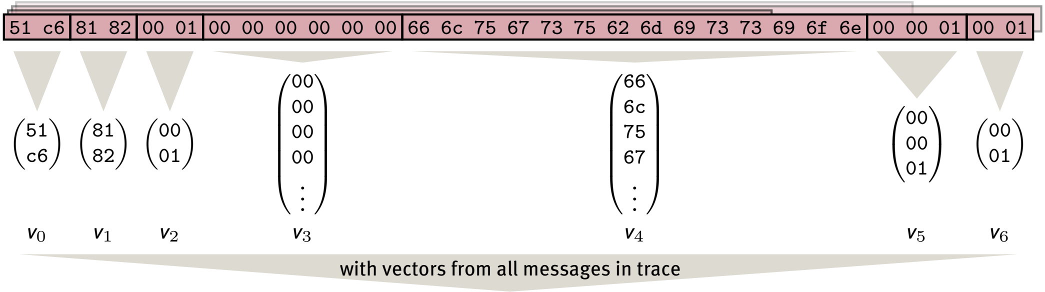
Calculate Segment Dissimilarities



Calculate Segment Dissimilarities



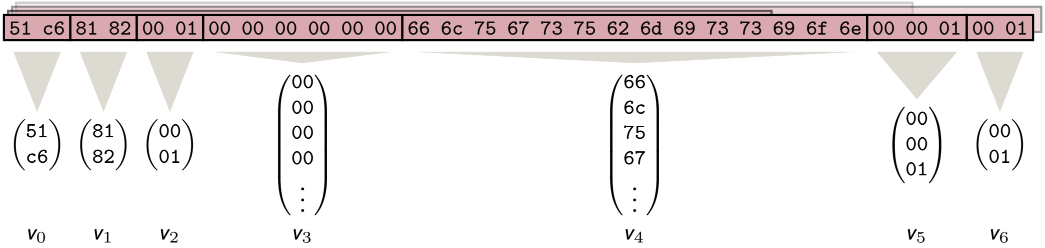
Calculate Segment Dissimilarities



Dissimilarity matrix **D**

	v_0	v_1	v_2	v_3	v_5	$\cdots$	v_n
v_0	0.0	0.2	1.0	1.0	0.7		1.0
v_1	0.2	0.0	1.0	1.0	0.6		1.0
v_2	1.0	1.0	0.0	0.7	1.0		0.0
v_3	1.0	1.0	0.7	0.0	1.0		0.7
v_5	0.7	0.6	1.0	1.0	0.0		1.0
$\vdots$						$\ddots$	0.3
v_n	1.0	1.0	0.0	0.7	1.0	0.3	0.0

Calculate Segment Dissimilarities



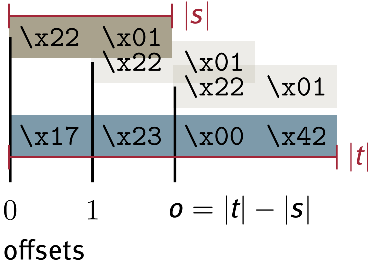
with vectors from all messages in trace

Dissimilarity matrix **D**

	v_0	v_1	v_2	v_3	v_5	$\cdots$	v_n
v_0	0.0	0.2	1.0	1.0	0.7		1.0
v_1	0.2	0.0	1.0	1.0	0.6		1.0
v_2	1.0	1.0	0.0	0.7	1.0		0.0
v_3	1.0	1.0	0.7	0.0	1.0		0.7
v_5	0.7	0.6	1.0	1.0	0.0		1.0
$\vdots$						$\ddots$	0.3
v_n	1.0	1.0	0.0	0.7	1.0	0.3	0.0

pairwise,
gradual dissimilarity
instead of boolean match

Linear Subspaces from Sliding Window



$$d_C\left(\begin{pmatrix}0x17\\0x23\end{pmatrix},\begin{pmatrix}0x22\\0x01\end{pmatrix}\right)$$

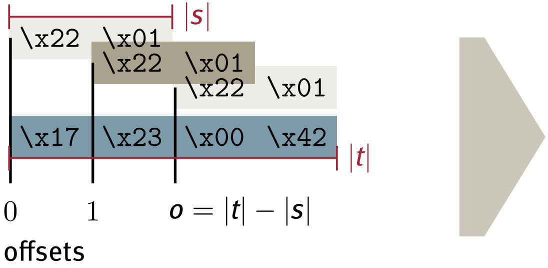
$$d_C\left(\begin{pmatrix}0x23\\0x00\end{pmatrix},\begin{pmatrix}0x22\\0x01\end{pmatrix}\right)$$

$$d_C\left(\begin{pmatrix}0x00\\0x42\end{pmatrix},\begin{pmatrix}0x22\\0x01\end{pmatrix}\right)$$

Canberra distance d_C

weighted L_1 or Manhattan distance

Linear Subspaces from Sliding Window



$$d_C\left(\begin{pmatrix}0x17\\0x23\end{pmatrix},\begin{pmatrix}0x22\\0x01\end{pmatrix}\right)$$

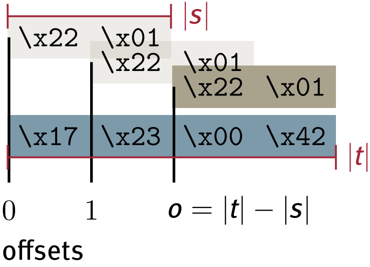
$$d_C\left(\begin{pmatrix}0x23\\0x00\end{pmatrix},\begin{pmatrix}0x22\\0x01\end{pmatrix}\right)$$

$$d_C\left(\begin{pmatrix}0x00\\0x42\end{pmatrix},\begin{pmatrix}0x22\\0x01\end{pmatrix}\right)$$

Canberra distance d_C

weighted L_1 or Manhattan distance

Linear Subspaces from Sliding Window



$$d_C\left(\begin{pmatrix}0x17\\0x23\end{pmatrix}, \begin{pmatrix}0x22\\0x01\end{pmatrix}\right)$$

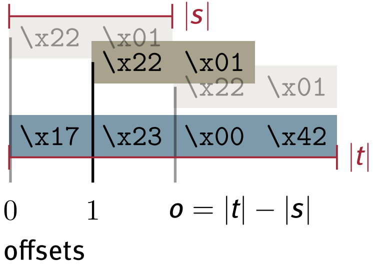
$$d_C\left(\begin{pmatrix}0x23\\0x00\end{pmatrix}, \begin{pmatrix}0x22\\0x01\end{pmatrix}\right)$$

$$d_C\left(\begin{pmatrix}0x00\\0x42\end{pmatrix}, \begin{pmatrix}0x22\\0x01\end{pmatrix}\right)$$

Canberra distance d_C

weighted L_1 or Manhattan distance

Linear Subspaces from Sliding Window



$$d_C\left(\begin{pmatrix}0x17\\0x23\end{pmatrix},\begin{pmatrix}0x22\\0x01\end{pmatrix}\right)$$

$$d_C\left(\begin{pmatrix}0x23\\0x00\end{pmatrix},\begin{pmatrix}0x22\\0x01\end{pmatrix}\right)$$

$$d_C\left(\begin{pmatrix}0x00\\0x42\end{pmatrix},\begin{pmatrix}0x22\\0x01\end{pmatrix}\right)$$

Minimum Canberra distance

$$d_\beta(T, \mathrm{s}) = \frac{\min_T(\{d_C(\mathbf{t}_{[o,o+|s|]})\})}{|s|} \tag{1}$$

Canberra-Ulm Dissimilarity[1]

$$d_m(\mathrm{s}, \mathrm{t}) = \qquad + \qquad + \tag{2}$$

[1] Stephan Kleber et al. „Message Type Identification of Binary Network Protocols using Continuous Segment Similarity“. In: *Proceedings of the Conference on Computer Communications*. INFOCOM. IEEE, 2020.

Canberra-Ulm Dissimilarity[1]

$$d_m(\mathrm{s}, \mathrm{t}) = \underbrace{\frac{|s|}{|t|} d_\beta(\mathrm{s}, \mathrm{t})}_{\text{subterm 1}} + \qquad + \tag{2}$$

normalize d_β

[1] Stephan Kleber et al. „Message Type Identification of Binary Network Protocols using Continuous Segment Similarity“. In: *Proceedings of the Conference on Computer Communications*. INFOCOM. IEEE, 2020.

Canberra-Ulm Dissimilarity[1]

$$d_m(\mathrm{s},\mathrm{t}) = \underbrace{\frac{|s|}{|t|} d_\beta(\mathrm{s},\mathrm{t})}_{\text{subterm 1}} + \underbrace{r}_{\text{subterm 2}} + \quad (2)$$

with the relative segment length difference

$$r = \frac{|t| - |s|}{|t|}$$

[1] Stephan Kleber et al. „Message Type Identification of Binary Network Protocols using Continuous Segment Similarity“. In: *Proceedings of the Conference on Computer Communications*. INFOCOM. IEEE, 2020.

Canberra-Ulm Dissimilarity[1]

$$d_m(\mathrm{s},\mathrm{t}) = \underbrace{\frac{|s|}{|t|} d_\beta(\mathrm{s},\mathrm{t})}_{\text{subterm 1}} + \underbrace{r}_{\text{subterm 2}} + \underbrace{(1 - d_\beta(\mathrm{s},\mathrm{t}))r(\frac{|s|}{|t|^2} - p_f)}_{\text{subterm 3}} \quad (2)$$

penalize absolute dimensionality differences

with the relative segment length difference

$$r = \frac{|t| - |s|}{|t|}$$

„2 out of 4 bytes is less information than 4 out of 8 bytes“ despite both $r = \frac{1}{2}$

[1] Stephan Kleber et al. „Message Type Identification of Binary Network Protocols using Continuous Segment Similarity“. In: *Proceedings of the Conference on Computer Communications*. INFOCOM. IEEE, 2020.

Canberra-Ulm Dissimilarity[1]

$$d_m(\mathrm{s},\mathrm{t}) = \underbrace{\frac{|s|}{|t|} d_\beta(\mathrm{s},\mathrm{t})}_{\text{subterm 1}} + \underbrace{r}_{\text{subterm 2}} + \underbrace{(1 - d_\beta(\mathrm{s},\mathrm{t})) r (\frac{|s|}{|t|^2} - p_f)}_{\text{subterm 3}} \tag{2}$$

with the relative segment length difference

$$r = \frac{|t| - |s|}{|t|}$$

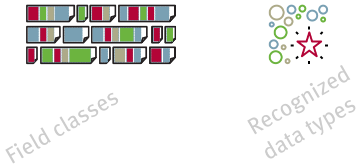
[1] Stephan Kleber et al. „Message Type Identification of Binary Network Protocols using Continuous Segment Similarity". In: *Proceedings of the Conference on Computer Communications*. INFOCOM. IEEE, 2020.

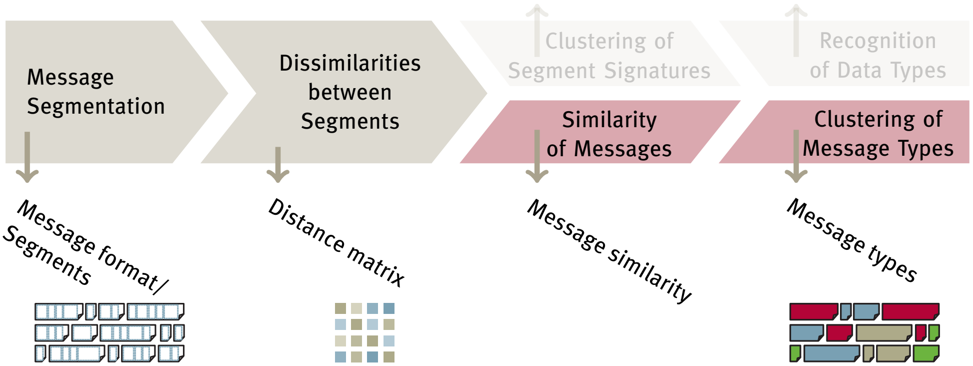
Result of Feature Extraction

Canberra-Ulm Dissimilarity of Segments

Basis for Field Data Type Classification
and Message Type Identification

NEMERE:
NEtwork MEssage
Reverse Engineering





Similarity of Messages

$m_0 =$	0208	0008	07	*GAP*
$m_1 =$	07	2700	0008	2317
$\vdots$				
$m_n =$	$\cdots$			

Segment dissimilarity
in conjunction with
Needleman-Wunsch (NW)
Sequence Alignment

Similarity of Messages

$m_0 =$	0208	0008	07	*GAP*
$m_1 =$	07	2700	0008	2317
$\vdots$				
$m_n =$	$\cdots$			

Segment dissimilarity in conjunction with Needleman-Wunsch (NW) Sequence Alignment

NW-scores message similarity:

	m_0	m_1	$\cdots$	m_n
m_0	4	0.76		
m_1	0.76	3		
$\vdots$			$\ddots$	
m_n				

DBSCAN Clustering

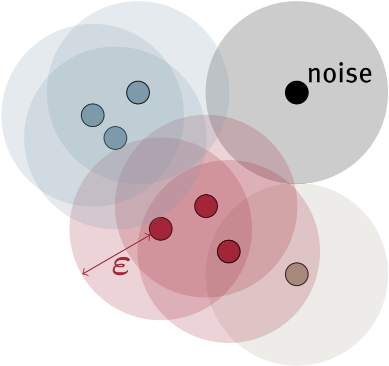
Density-Based Spatial Clustering of Applications with Noise

Main Parameter ε

Range around a density core of samples that should constitute a cluster

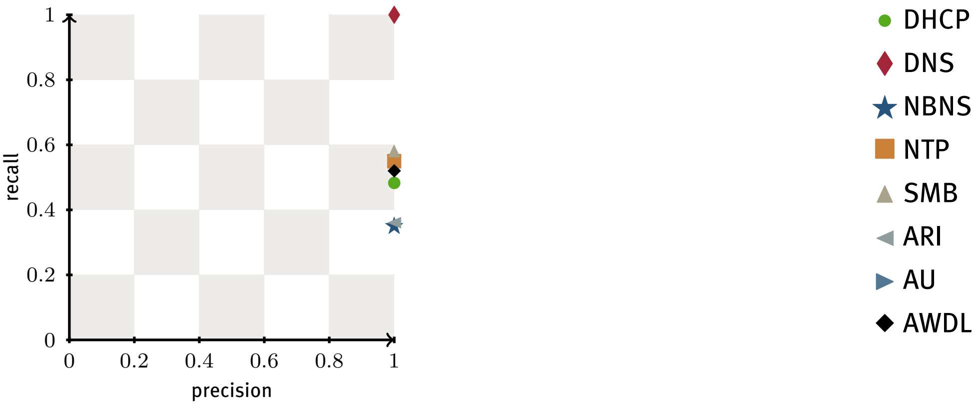
Auto-Configuration of ε

Greatest change in message similarity distribution



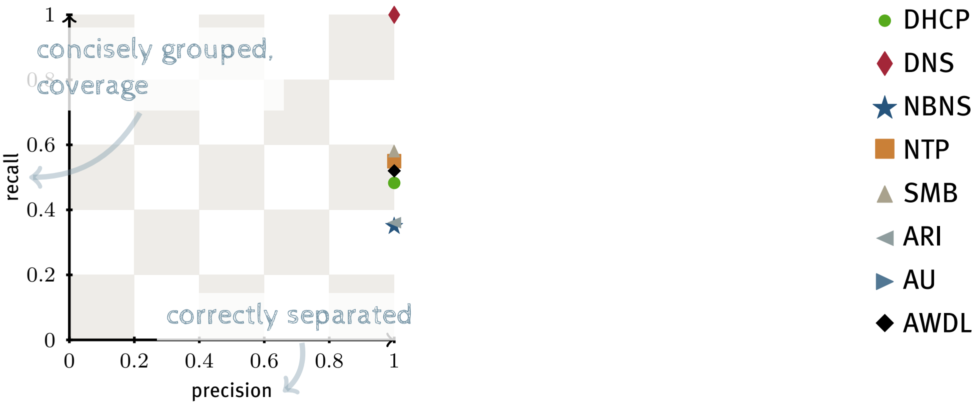
Evaluation Results: Message Type Quality...

... with segments from **Wireshark**



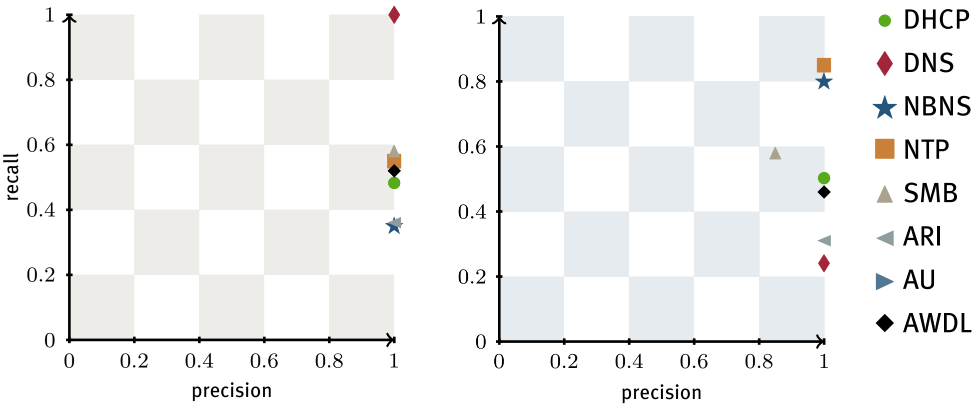
Evaluation Results: Message Type Quality...

... with segments from **Wireshark**



Evaluation Results: Message Type Quality...

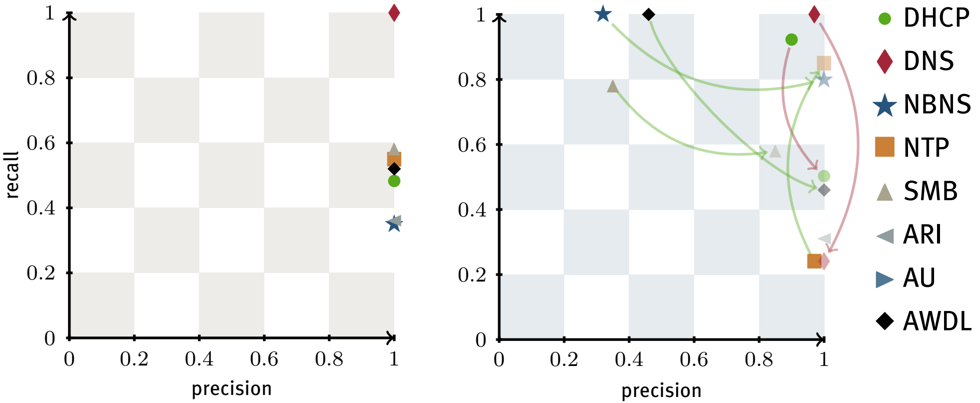
... with segments from **Wireshark** ... with segments from **NEMEPCA**



Evaluation Results: Message Type Quality...

... with segments from **Wireshark**

... with segments from **NEMEPCA**
... when clustering with **Netzob**



Evaluation Interpretation

Prioritize Precision over Recall

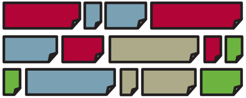
Wireshark Canberra-Ulm Dissimilarity works as expected, differences in message structure reveal message types.

Netzob Netzob's recall outperforms NEMESYS in few cases, Netzob's precision is unreliable.

NEMEPCA Close-to-perfect precision with heuristic segments, Segmentation quality has tremendous effect.

Result of Message Type Identification

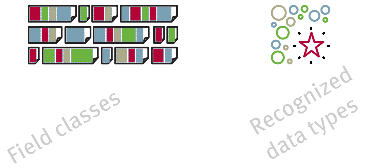
Clusters of Messages resembling Message Types

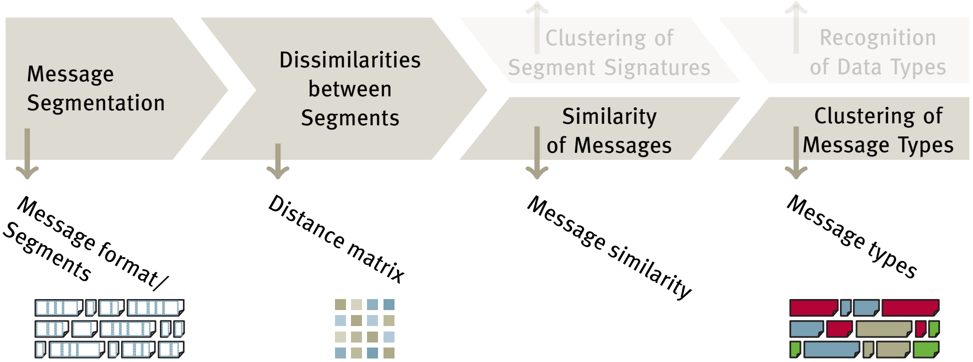


NemeTYL: NEtwork MEssage TYpe identification by aLignment (INFOCOM2020)[1]

[1] Stephan Kleber et al. „Message Type Identification of Binary Network Protocols using Continuous Segment Similarity“. In: *Proceedings of the Conference on Computer Communications*. INFOCOM. IEEE, 2020.

NEMERE:
NEtwork MEssage
Reverse Engineering





Limitations and Future Work

Limitations

- Encryption, compression, and obfuscation
- Empirical parameters. Robustness thoroughly tested but not provable

Limitations and Future Work

Limitations

- Encryption, compression, and obfuscation
- Empirical parameters. Robustness thoroughly tested but not provable

Future Work

- Alternatives to sequence alignment, e.g., LDA, LSTM
- Supervised learning of cluster properties for recognition by a ML model

Foundational Advances for Static Traffic Analysis

Related Work

Message Format Inference

Message Type Identification

Semantic Deduction

Foundational Advances for Static Traffic Analysis

Related Work

Contributions: Static Traffic Analysis process formalization

Message Format Inference

Message Type Identification

Semantic Deduction

Foundational Advances for Static Traffic Analysis

Related Work

Contributions: Static Traffic Analysis process formalization

Message Format Inference

Contributions: Deltas of Bit Congruence + PCA-based heuristic refinements

Message Type Identification

Semantic Deduction

Foundational Advances for Static Traffic Analysis

Related Work

Contributions: Static Traffic Analysis process formalization

Message Format Inference

Contributions: Deltas of Bit Congruence + PCA-based heuristic refinements

Message Type Identification

Contributions: Canberra-Ulm Dissimilarity of segments + DBSCAN clustering autoconf.

Semantic Deduction

Foundational Advances for Static Traffic Analysis

Related Work

Contributions: Static Traffic Analysis process formalization

Message Format Inference

Contributions: Deltas of Bit Congruence + PCA-based heuristic refinements

Message Type Identification

Contributions: Canberra-Ulm Dissimilarity of segments + DBSCAN clustering autoconf.

Semantic Deduction

Contribution: First generic semantic interpretation of field data types from traces

Foundational Advances for Static Traffic Analysis

Related Work

Contributions: Static Traffic Analysis process formalization

Message Format Inference

Contributions: Deltas of Bit Congruence + PCA-based heuristic refinements

Message Type Identification

Contributions: Canberra-Ulm Dissimilarity of segments + DBSCAN clustering autoconf.

Semantic Deduction

Contribution: First generic semantic interpretation of field data types from traces

Peer-Reviewed Publications

dissertation-related

Stephan Kleber, Henning Kopp and Frank Kargl. „NEMESYS: Network Message Syntax Reverse Engineering by Analysis of the Intrinsic Structure of Individual Messages“. In: *Proceedings of the 12th USENIX Workshop on Offensive Technologies*. WOOT. USENIX Association, 2018.

Stephan Kleber and Frank Kargl. „Poster: Network Message Field Type Recognition“. In: *Proceedings of the 26th Conference on Computer and Communications Security*. CCS. 2019.

Stephan Kleber, Lisa Maile and Frank Kargl. „Survey of Protocol Reverse Engineering Algorithms: Decomposition of Tools for Static Traffic Analysis“. In: *IEEE Communications Surveys and Tutorials* 21.1 (Feb. 2019). Firstquarter.

Stephan Kleber, Rens Wouter van der Heijden and Frank Kargl. „Message Type Identification of Binary Network Protocols using Continuous Segment Similarity“. In: *Proceedings of the Conference on Computer Communications*. INFOCOM. IEEE, 2020.

Stephan Kleber and Frank Kargl. „Refining Network Message Segmentation with Principal Component Analysis“. In: *Proceedings of the tenth annual IEEE Conference on Communications and Network Security*. CNS. IEEE, 2022.

Stephan Kleber, Milan Stute, Matthias Hollick and Frank Kargl. „Network Message Field Type Classification and Recognition for Unknown Binary Protocols“. In: *Proceedings of the DSN Workshop on Data-Centric Dependability and Security*. DCDS. IEEE/IFIP, 2022.

further

Stephan Kleber, Rens W. van der Heijden, Henning Kopp and Frank Kargl. „Terrorist Fraud Resistance of Distance Bounding Protocols Employing Physical Unclonable Functions“. In: *Proceedings of the International Conference and Workshops on Networked Systems*. NetSys. IEEE, 2015.

Florian Unterstein, Stephan Kleber, Matthias Matousek, Frank Kargl, Frank Slomka and Matthias Hiller. „Design of the Secure Execution PUF-based Processor (SEPP)“. In: *Proceedings of the Workshop on Trustworthy Manufacturing and Utilization of Secure Devices, TRUDEVICE 2015*. Universität Ulm, 2015.

Stephan Kleber, Henrik Ferdinand Nölscher and Frank Kargl. „Automated PCB Reverse Engineering“. In: *Proceedings of the 11th USENIX Workshop on Offensive Technologies*. WOOT. USENIX Association, 2017.

Stephan Kleber, Florian Unterstein, Matthias Hiller, Frank Slomka, Matthias Matousek, Frank Kargl and Christoph Boesch. „Secure Code Execution: A Generic PUF-Driven System Architecture“. In: *Proceedings of the 21st Information Security (ISC 2018)*. Universität Ulm, 2018.

Thomas Lukaseder, Kevin Stölzle, Stephan Kleber, Benjamin Erb and Frank Kargl. „An SDN-based Approach For Defending Against Reflective DDoS Attacks“. In: *Proceedings of the Conference on Local Computer Networks (LCN)*. IEEE, 2018.

Tobias Kröll, Stephan Kleber, Frank Kargl, Matthias Hollick and Jiska Classen. „ARIstoteles - Dissecting Apple's Baseband Interface“. In: *Proceedings of the European Symposium on Research in Computer Security*. ESORICS. 2021.

Patrick Wachter and Stephan Kleber. „Analysis of the DoIP Protocol for Security Vulnerabilities“. In: *Proceedings of the Computer Science in Cars Symposium*. CSCS. ACM, 2022.

Stephan Kleber and Patrick Wachter. „A Strategy to Evaluate Test Time Evasion Attack Feasibility“. In: *Datenschutz und Datensicherheit - DuD* 47.8 (Aug. 2023), S. 478–482.

Overview of Contributions

Preparation

Data Collection and Preprocessing

Message Format Inference

Feature Extraction

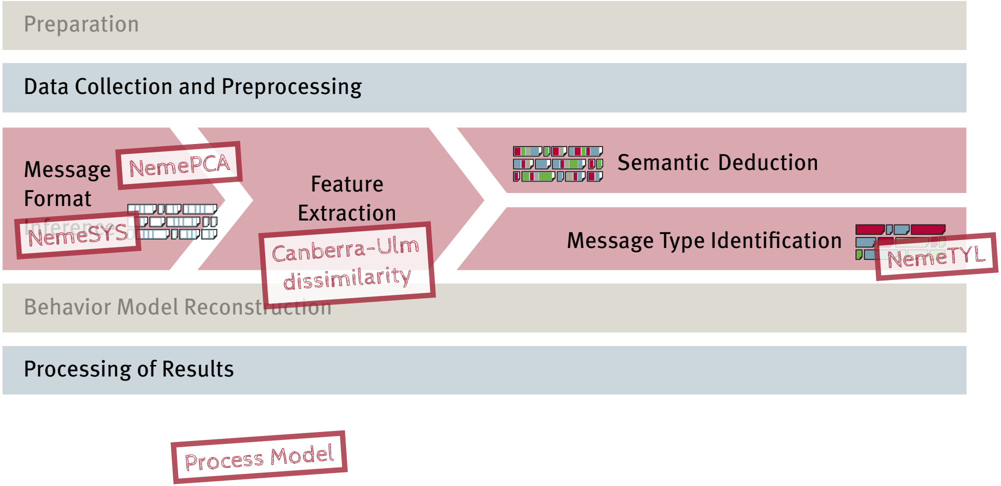
Semantic Deduction

Message Type Identification

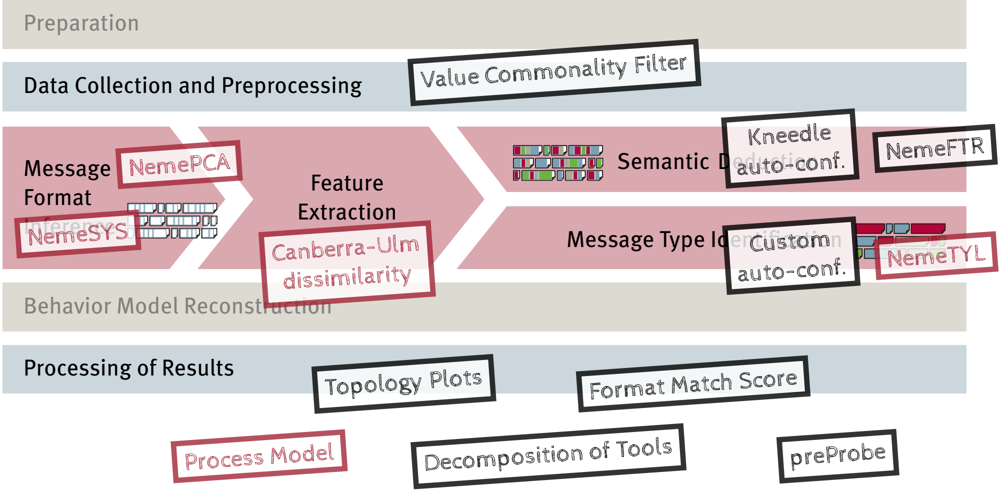
Behavior Model Reconstruction

Processing of Results

Overview of Contributions



Overview of Contributions



THANK YOU!

Questions?

web kleber.space/en/research
mail stephan.kleber@uni-ulm.de
LinkedIn stephan-kleber

Institute of Distributed Systems, Ulm University

web uulm.de/in/vs
mail frank.kargl@uni-ulm.de
github github.com/vs-uulm

intentionally left blank

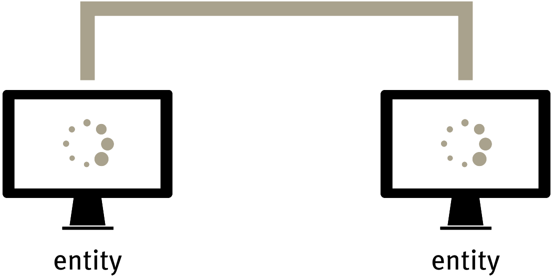
Icon Sources

Pictograms from Stephan Kleber, based on icon set by Lisa Maile

From the Noun Project, modified by Stephan Kleber:

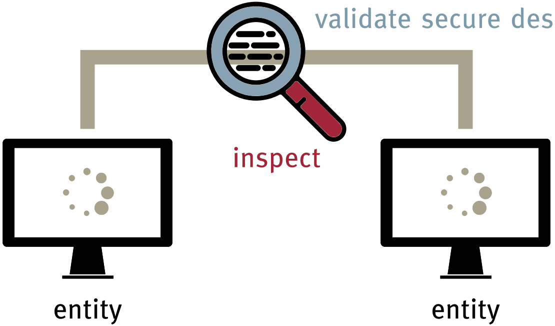
- *Searching* Created by Ziyad Al junaidi
- *Communication* Created by SlideGenius
- *Sensor* Created by Adnen Kadri
- *Specification* Created by ProSymbols
- *Paper* Created by Ilham Fitrotul Hayat
- *Hourglass* Created by Aswell Studio

Motivation for Protocol Reverse Engineering



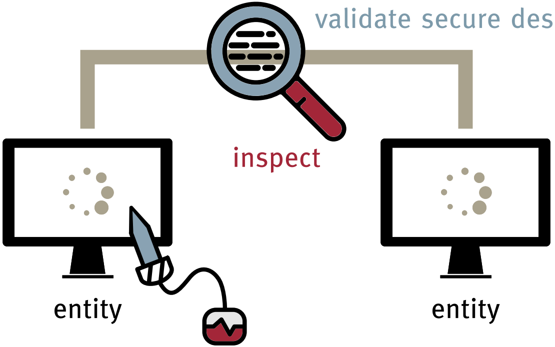
Motivation for Protocol Reverse Engineering

uncover/understand malware communication,
validate secure design



Motivation for Protocol Reverse Engineering

uncover/understand malware communication,
validate secure design

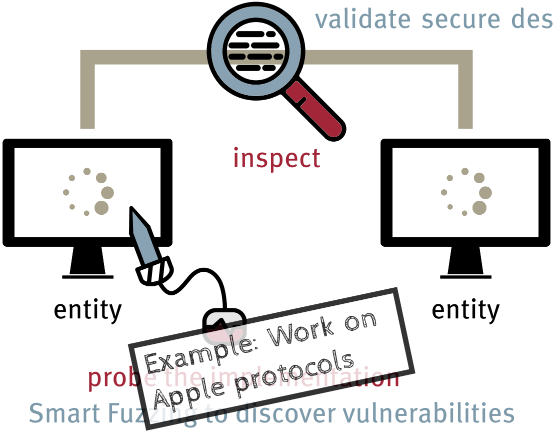


probe the implementation

Smart Fuzzing to discover vulnerabilities

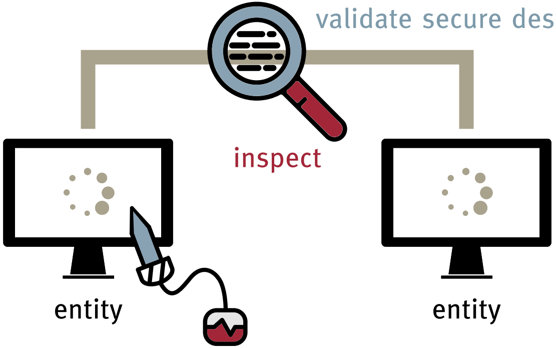
Motivation for Protocol Reverse Engineering

uncover/understand malware communication,
validate secure design



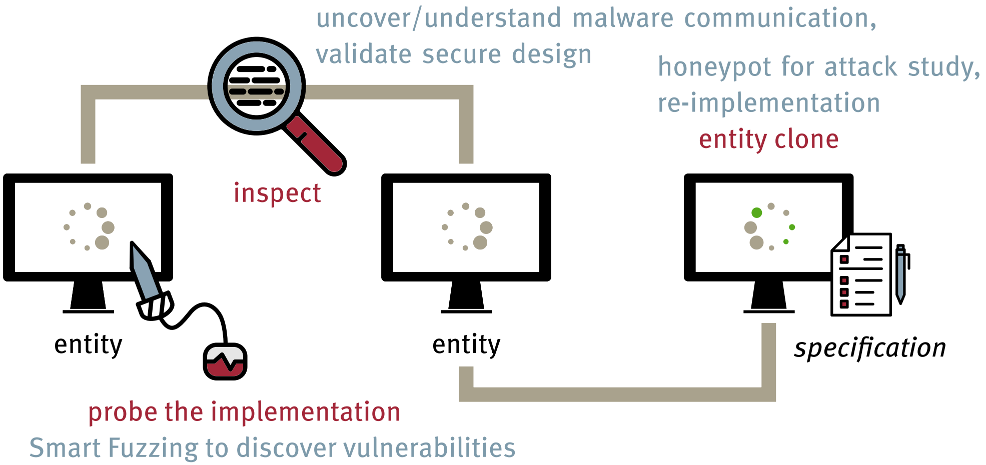
Motivation for Protocol Reverse Engineering

uncover/understand malware communication,
validate secure design

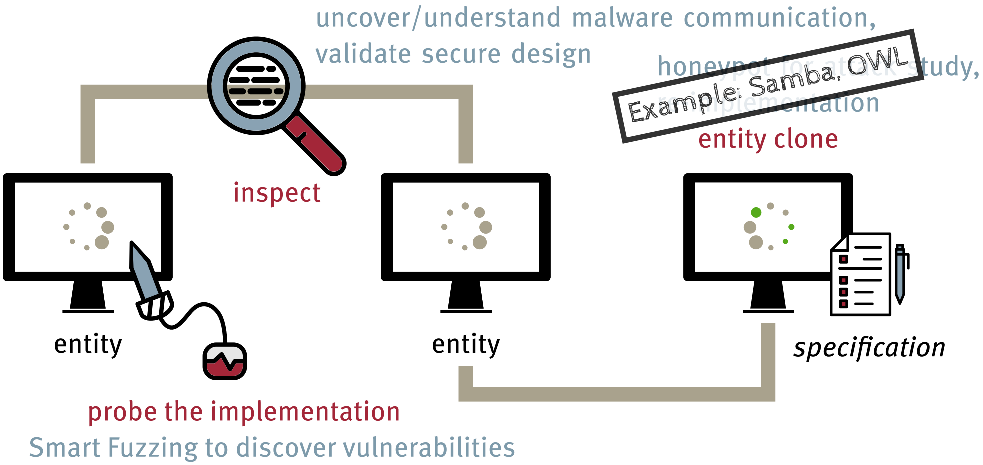


probe the implementation
Smart Fuzzing to discover vulnerabilities

Motivation for Protocol Reverse Engineering



Motivation for Protocol Reverse Engineering



References for Use Cases

- Validating the correct and secure implementation of network services

  - Rouf, Ishtiaq, et al. „Security and Privacy Vulnerabilities of In-Car Wireless Networks: A Tire Pressure Monitoring System Case Study“. In Proceedings of the 19th USENIX Security Symposium, 323–38. USENIX Association, 2010.
  - Halperin, Daniel, et al. „Pacemakers and Implantable Cardiac Defibrillators: Software Radio Attacks and Zero-Power Defenses“. In IEEE Symposium on Security and Privacy. SP. Washington, DC, USA: IEEE, 2008.
  - Fereidooni, Hossein, et al. „Breaking Fitness Records Without Moving: Reverse Engineering and Spoofing Fitbit“. In 20th International Symposium Research in Attacks, Intrusions, and Defenses. RAID. Atlanta, GA, USA: Springer, 2017.
  - Ji, Ran, et al. „Automatic Reverse Engineering of Private Flight Control Protocols of UAVs“. Security and Communication Networks 2017.
  - Wen, Shameng, et al. „Protocol Vulnerability Detection Based on **Network Traffic Analysis** and Binary Reverse Engineering“. PLOS ONE 12, Nr. 10 (19. Oktober 2017).
  - Rios, Billy, and Jonathan Butts. „Understanding and Exploiting Implanted Medical Devices“. Black Hat USA, Las Vegas, 9. August 2018.
  - Stute, Milan, David Kreitschmann, and Matthias Hollick. „One Billion Apples’ Secret Sauce: Recipe for the Apple Wireless Direct Link Ad Hoc Protocol“. Proceedings of the 24th Annual International Conference on Mobile Computing and Networking - MobiCom ’18, 2018.
  - Stute, Milan, et al. „A Billion Open Interfaces for Eve and Mallory: MitM, DoS, and Tracking Attacks on IOS and MacOS Through Apple Wireless Direct Link“, 2019.

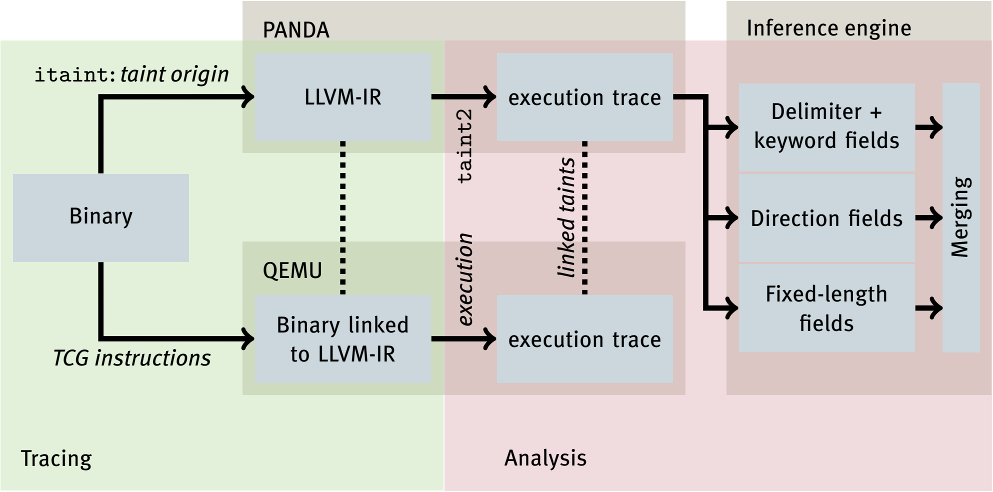
- Define input formats for Smart Fuzzing

  - Gascon, Hugo, et al. „PULSAR: Stateful **Black-Box Fuzzing** of Proprietary Network Protocols“. In 11th International Conference of Security and Privacy in Communication Networks, Revised Selected Papers. SecureComm. Dallas, TX, USA: Springer, 2015.
  - Blaze Information Security - Wildfire Labs. „Fuzzing proprietary protocols with Scapy, radamsa and a handful of PCAPs“, 10. Juni 2017.
  - Fiterau-Brostean, Paul, et al. „Analysis of DTLS Implementations Using Protocol State Fuzzing“, 29th USENIX Security Symposium. USENIX Security, 2020.

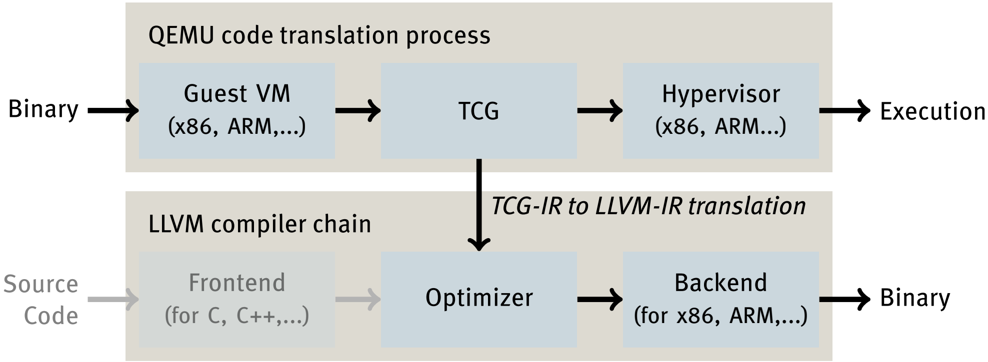
References for Use Cases

- Malware and Botnet analysis: Understand Command-and-Control-Server communication
  - Cui, Weidong. „Automating malware detection by **inferring intent**“. University of California, Berkeley, 2006.
  - Cho, Chia Y., et al. „Inference and **Analysis of Formal Models** of Botnet Command and Control Protocols“. In Proceedings of the 17th ACM Conference on Computer and Communications Security. CCS. New York, NY, USA: ACM, 2010.
- Network modeling for anomaly detection
  - Bieniasz, Jędrzej, et al. „Towards **Model-Based Anomaly Detection** in Network Communication Protocols“. In International Conference on Frontiers of Signal Processing, 126–30. ICFSP. IEEE, 2016.
  - Wressnegger, Christian, Ansgar Kellner, and Konrad Rieck. „ZOE: Content-Based Anomaly Detection for Industrial Control Systems“. In Proceedings of the 48th Conference on Dependable Systems and Networks, 2018.
- Honeypot setup
  - Leita, Corrado, Ken Mermoud, and Marc Dacier. „ScriptGen: An Automated Script Generation Tool for Honeyd“. In Proceedings of the 21st Annual Computer Security Applications Conference, 203–14. ACSAC. Tucson, AZ, USA: IEEE, 2005.
  - Krueger, Tammo, Hugo Gascon, Nicole Krämer, und Konrad Rieck. „Learning Stateful Models for Network Honeypots“. In Proceedings of the 5th ACM Workshop on Security and Artificial Intelligence, 37–48. AISec. New York, NY, USA: ACM, 2012.
- Re-implementation
  - Tridgell, Andrew. „How Samba Was Written“. www.samba.org, August 2003.
  - Instant Messaging protocols like OSCAR, Yahoo!, and QQ

Automated Architecture-Ind. Extraction of Message Formats



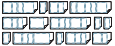
Overview of PANDA's translation process



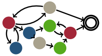
Static Traffic Analysis Process: Survey[1]

D. Message Format Inference

E. Message Type Identification

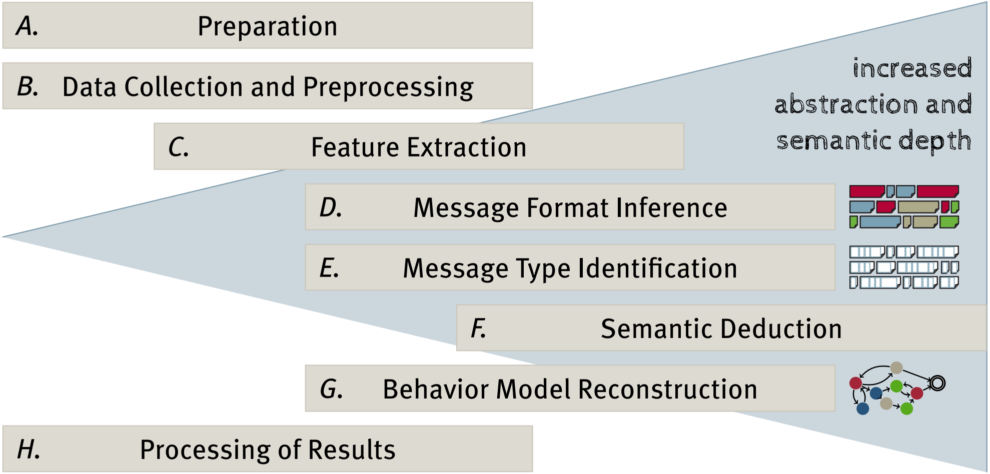


G. Behavior Model Reconstruction



[1] Stephan Kleber et al. „Survey of Protocol Reverse Engineering Algorithms: Decomposition of Tools for Static Traffic Analysis". In: *IEEE Communications Surveys and Tutorials* 21.1 (Feb. 2019). Firstquarter.

Static Traffic Analysis Process: Survey[1]



[1] Stephan Kleber et al. „Survey of Protocol Reverse Engineering Algorithms: Decomposition of Tools for Static Traffic Analysis“. In: *IEEE Communications Surveys and Tutorials* 21.1 (Feb. 2019). Firstquarter.

Static Traffic Analysis Process: Survey[1]

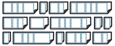
A. Preparation

B. Data Collection and Preprocessing

C. Feature Extraction

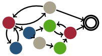
D. Message Format Inference

E. Message Type Identification



F. Semantic Deduction

G. Behavior Model Reconstruction



H. Processing of Results

---

[1] Stephan Kleber et al. „Survey of Protocol Reverse Engineering Algorithms: Decomposition of Tools for Static Traffic Analysis“. In: *IEEE Communications Surveys and Tutorials* 21.1 (Feb. 2019). Firstquarter.

Static Traffic Analysis: Related Work Survey

Discoverer[1]: Message Types by Segmentation of textual message parts.

PRISMA[2]: Message Types and Behavior using Markov Models.

Netzob[3]: Message Types and Formats by aligning identical bytes in messages.

FieldHunter[4]: Identify few specific field types within messages.

Contiguous Sequential Pattern[5]: Recursive inference by frequency analysis.

---

[1] Weidong Cui et al., „Discoverer: Automatic Protocol Reverse Engineering from Network Traces“, USENIX Security 2007.

[2] Tammo Krueger et al., „Learning Stateful Models for Network Honeypots“, AISec 2012.

[3] Georges Bossert et al., „Towards Automated Protocol Reverse Engineering Using Semantic Information“, CCS 2014.

[4] Ignacio Bermudez et al., „Towards Automatic Protocol Field Inference“, COMCOM 84 (2016).

[5] Y.-H. Goo et al., „Protocol Specification Extraction Based on Contiguous Sequential Pattern Algorithm“, IEEE Access, vol 7 (2019).

Static Traffic Analysis: Limitations of Related Work

- Fixed message length and similar syntaxes — Discoverer, PRISMA, FieldHunter
- Few, specific heuristics with low coverage — Discoverer, FieldHunter
- Inefficient application of sequence alignment — Netzob
- Insufficient coarse-grained similarity measures for binary data — Netzob
- Requires environment/context information like flow associations — FieldHunter

Research Questions

1. Which methods are currently used for application in PRE, and which of these are candidates to improve automation?
2. What is the generic process for STA, and which steps offer room for improved automation?
3. Which methods and algorithms are suitable for improving automation and result quality, and how must they be applied to reliably infer arbitrary communication?
4. How can the correctness of the specification inference be measured?
5. Has traffic analysis with active probing the potential to surpass STA’s correctness and to automatically discover insights not contained in the traces?

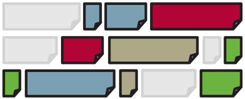
Research Questions - Sub-Questions of RQ3

3 Which methods and algorithms are suitable for improving automation and result quality, and how must they be applied to reliably infer arbitrary communication?

- 3.A How can messages be efficiently split into segments that approximate fields?
- 3.B How can segments be related to generically characterize the message contents and deduce field properties?
- 3.C How can the format and content of messages reliably and correctly be inferred, as well as, message types and field data of an arbitrary communication robustly be classified?

Preprocess

Filtering trace for messages of target protocol



Sub-sample message number:
reduce memory complexity/limit runtime

Preprocess: Input Trace Optimization

- Reduce redundancy and increase value variance
- Value Commonality Filter:
  - Determine value frequency of NEMESYS segments
  - Calculate the median of the value frequencies throughout the message
  - Select unique messages with the least medians
- Truncate message number for comparing the evaluations of multiple traces

NEMESYS: Deltas of Bit Congruence

Bit Congruence:

based on similarity measure for bit strings
by Sokal and Michener (1958)

NEMESYS: Deltas of Bit Congruence

Bit Congruence:

$$\mathrm{BC}(b, \overline{b}) = \frac{c_{\mathrm{agree}}(b, \overline{b})}{8}$$

$c_{\mathrm{agree}}(b, \overline{b})$: number of congruent bits for bytes b and $\overline{b}$

NEMESYS: Deltas of Bit Congruence

$$\Delta\mathrm{BC} = \big(\mathrm{BC}(m_k, m_{k+1}) - \mathrm{BC}(m_{k-1}, m_k)\big)_{0<k<n}$$

with

$$\mathrm{BC}(b, \overline{b}) = \frac{c_{\mathrm{agree}}(b, \overline{b})}{8}$$

m_k: Message m's byte at position k, m has length $n+1$

$c_{\mathrm{agree}}(b, \overline{b})$: number of congruent bits for bytes b and $\overline{b}$

NemeSYS: Deltas of Bit Congruence

Illustration of a message   byte values color-coded: `0x00` = black to `0xff` = white

$$\Delta \mathrm{BC} = \big(\mathrm{BC}(m_k, m_{k+1}) - \mathrm{BC}(m_{k-1}, m_k)\big)_{0<k<n}$$

with

$$\mathrm{BC}(b, \overline{b}) = \frac{c_{\mathrm{agree}}(b, \overline{b})}{8}$$

m_k: Message m's byte at position k, m has length $n+1$

$c_{\mathrm{agree}}(b, \overline{b})$: number of congruent bits for bytes b and $\overline{b}$

NemeSYS: Deltas of Bit Congruence

$k = 1$

Illustration of a message   byte values color-coded: 0x00 = black to 0xff = white

$$\Delta \mathrm{BC} = \left(\mathrm{BC}(m_k, m_{k+1}) - \mathrm{BC}(m_{k-1}, m_k)\right)_{0<k<n}$$

with

$$\mathrm{BC}(b, \overline{b}) = \frac{c_{\mathrm{agree}}(b, \overline{b})}{8}$$

m_k: Message m's byte at position k, m has length $n+1$

$c_{\mathrm{agree}}(b, \overline{b})$: number of congruent bits for bytes b and $\overline{b}$

NEMESYS: Deltas of Bit Congruence

Illustration of a message   byte values color-coded: `0x00` = black to `0xff` = white

$$\Delta\mathrm{BC} = \left(\mathrm{BC}(m_k, m_{k+1}) - \mathrm{BC}(m_{k-1}, m_k)\right)_{0<k<n}$$

with

$$\mathrm{BC}(b, \overline{b}) = \frac{c_{\text{agree}}(b, \overline{b})}{8}$$

m_k: Message m's byte at position k, m has length $n+1$

$c_{\text{agree}}(b, \overline{b})$: number of congruent bits for bytes b and $\overline{b}$

NEMESYS: Deltas of Bit Congruence

$k = 3$

Illustration of a message   byte values color-coded: `0x00` = black to `0xff` = white

$$\Delta \mathrm{BC} = \left(\mathrm{BC}(m_k, m_{k+1}) - \mathrm{BC}(m_{k-1}, m_k)\right)_{0<k<n}$$

with

$$\mathrm{BC}(b, \overline{b}) = \frac{c_{\text{agree}}(b, \overline{b})}{8}$$

m_k: Message m's byte at position k, m has length $n+1$

$c_{\text{agree}}(b, \overline{b})$: number of congruent bits for bytes b and $\overline{b}$

NEMESYS: Deltas of Bit Congruence

Illustration of a message byte values color-coded: 0x00 = black to 0xff = white

$$\Delta \mathrm{BC} = \left(\mathrm{BC}(m_k, m_{k+1}) - \mathrm{BC}(m_{k-1}, m_k)\right)_{0<k<n}$$

with

$$\mathrm{BC}(b, \overline{b}) = \frac{c_{\mathrm{agree}}(b, \overline{b})}{8}$$

m_k: Message m's byte at position k, m has length $n+1$

$c_{\mathrm{agree}}(b, \overline{b})$: number of congruent bits for bytes b and $\overline{b}$

NEMESYS: Deltas of Bit Congruence

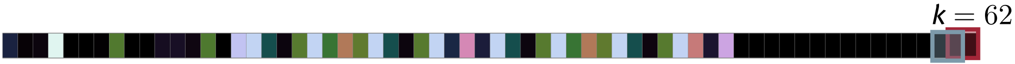


Illustration of a message   byte values color-coded: `0x00` = black to `0xff` = white

$$\Delta \mathrm{BC} = \big(\mathrm{BC}(m_k, m_{k+1}) - \mathrm{BC}(m_{k-1}, m_k)\big)_{0<k<n}$$

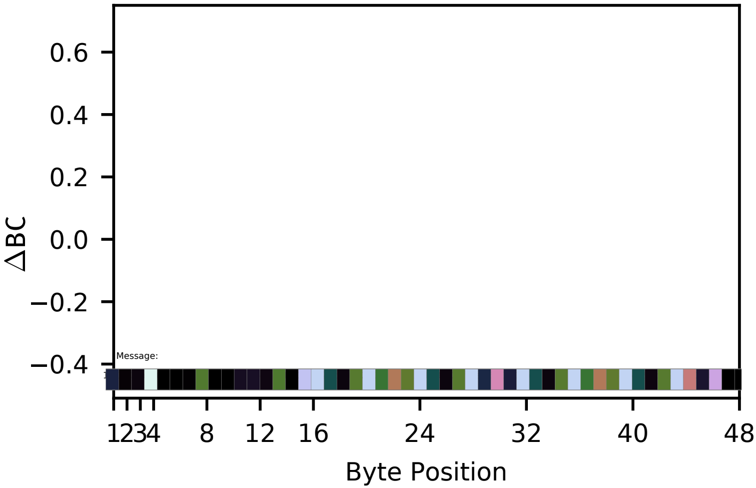
with

$$\mathrm{BC}(b, \overline{b}) = \frac{c_{\text{agree}}(b, \overline{b})}{8}$$

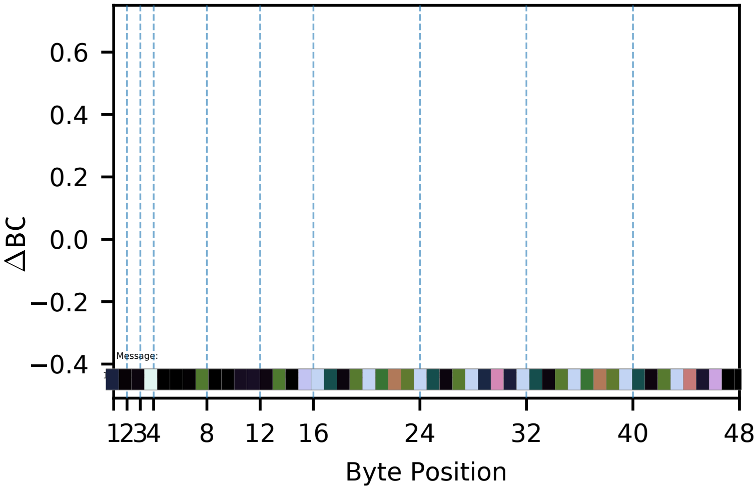
m_k: Message m's byte at position k, m has length $n+1$

$c_{\text{agree}}(b, \overline{b})$: number of congruent bits for bytes b and $\overline{b}$

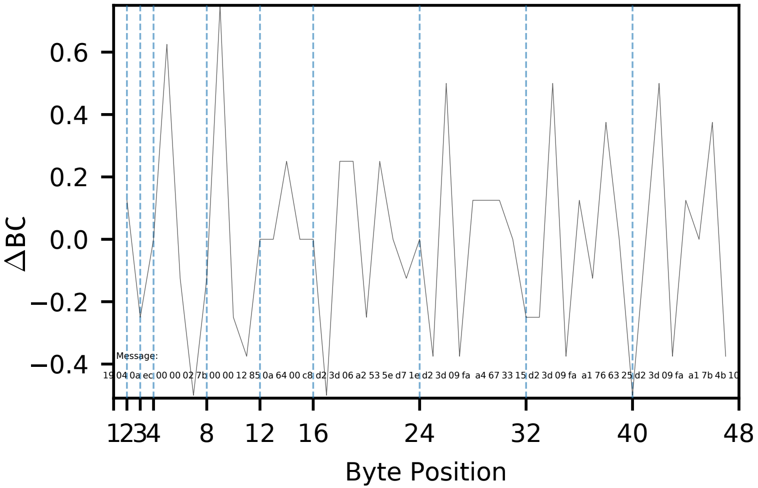
NEMESYS: Value Pattern Example NTP



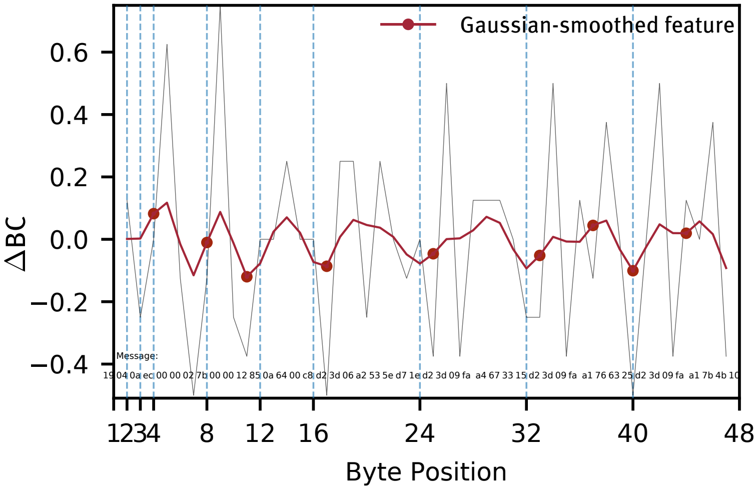
NEMESYS: Value Pattern Example NTP



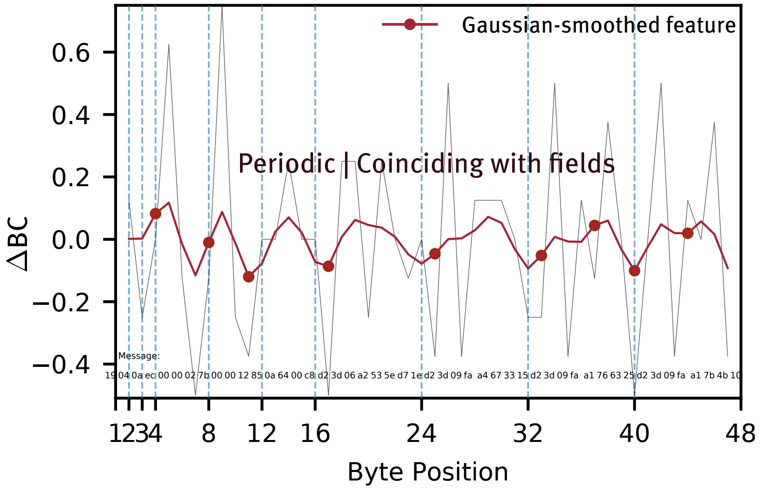
NEMESYS: Value Pattern Example NTP



NEMESYS: Value Pattern Example NTP



NEMESYS: Value Pattern Example NTP



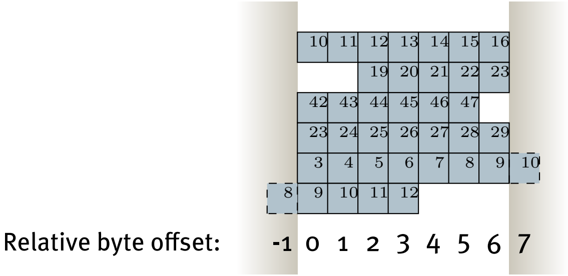
NemeSYS: Heuristic Position of Field Boundaries

Feature $\triangle$BC:
distinctive distribution for binary numbers:

- At field transition: low $\triangle$BC
- Towards field end: high $\triangle$BC
- Gaussian filter $g_\sigma(\cdot)$ to reduce noise

Inflection points of rising edges of $g_\sigma(\triangle\mathrm{BC})$

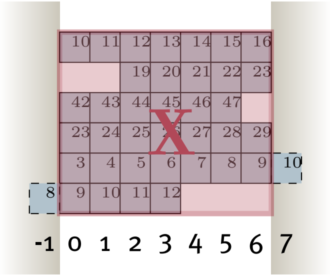
Overlaying Segment Vectors



- Superimpose segments at most useful offsets: meaningfully comparable
- Quantified by Canberra dissimilarity: *Kleber et al., INFOCOM 2020*
  extension of Canberra distance to vectors of differing dimensions
- Relative byte offsets of all segments at lowest dissimilarity

Overlaying Segment Vectors

Relative byte offset:



- Superimpose segments at most useful offsets: meaningfully comparable
- Quantified by Canberra dissimilarity: *Kleber et al., INFOCOM 2020*
  extension of Canberra distance to vectors of differing dimensions
- Relative byte offsets of all segments at lowest dissimilarity

Covariance

Byte values of similar segments:

	Relative byte offset 0	1	2	3	4
Segment 1	00	08	50	00	02
Segment 2	01	08	90	00	04
Segment 3	01	08	90	00	07
Segment 4	01	08	b0	00	02
Segment 5	02	90	40	01	02
Segment 6	02	90	40	01	02
Segment 7	01	08	80	00	04
Segment 8	01	08	80	00	04

Covariance

Byte values of similar segments:

$$\mathbf{X} = \begin{pmatrix} 00 & 08 & 50 & 00 & 02 \\ 01 & 08 & 90 & 00 & 04 \\ 01 & 08 & 90 & 00 & 07 \\ 01 & 08 & \text{b0} & 00 & 02 \\ 02 & 90 & 40 & 01 & 02 \\ 02 & 90 & 40 & 01 & 02 \\ 01 & 08 & 80 & 00 & 04 \\ 01 & 08 & 80 & 00 & 04 \end{pmatrix}$$

Covariance

Byte values of similar segments:

$$\mathbf{C} = \begin{pmatrix} 0.41 & 34 & -9.71 & 0.25 & -0.19 \\ 34 & 3963 & -2020 & 29.14 & -53.42 \\ -9.71 & -2020 & 1737 & -14.85 & 34.85 \\ 0.25 & 29.14 & -14.85 & 0.21 & -0.39 \\ -0.19 & -53.42 & 34.85 & -0.39 & 3.12 \end{pmatrix}$$

$$\mathbf{X} = \begin{pmatrix} \texttt{00} & \texttt{08} & \texttt{50} & \texttt{00} & \texttt{02} \\ \texttt{01} & \texttt{08} & \texttt{90} & \texttt{00} & \texttt{04} \\ \texttt{01} & \texttt{08} & \texttt{90} & \texttt{00} & \texttt{07} \\ \texttt{01} & \texttt{08} & \texttt{b0} & \texttt{00} & \texttt{02} \\ \texttt{02} & \texttt{90} & \texttt{40} & \texttt{01} & \texttt{02} \\ \texttt{02} & \texttt{90} & \texttt{40} & \texttt{01} & \texttt{02} \\ \texttt{01} & \texttt{08} & \texttt{80} & \texttt{00} & \texttt{04} \\ \texttt{01} & \texttt{08} & \texttt{80} & \texttt{00} & \texttt{04} \end{pmatrix}$$

Covariance

Byte values of similar segments:

$$\mathbf{X} = \begin{pmatrix} \texttt{00} & \texttt{08} & \texttt{50} & \texttt{00} & \texttt{02} \\ \texttt{01} & \texttt{08} & \texttt{90} & \texttt{00} & \texttt{04} \\ \texttt{01} & \texttt{08} & \texttt{90} & \texttt{00} & \texttt{07} \\ \texttt{01} & \texttt{08} & \texttt{b0} & \texttt{00} & \texttt{02} \\ \texttt{02} & \texttt{90} & \texttt{40} & \texttt{01} & \texttt{02} \\ \texttt{02} & \texttt{90} & \texttt{40} & \texttt{01} & \texttt{02} \\ \texttt{01} & \texttt{08} & \texttt{80} & \texttt{00} & \texttt{04} \\ \texttt{01} & \texttt{08} & \texttt{80} & \texttt{00} & \texttt{04} \end{pmatrix}$$

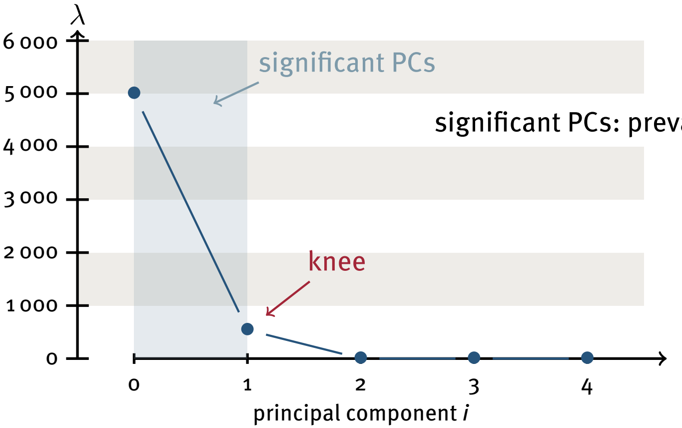
$$\mathbf{C} = \begin{pmatrix} 0.41 & 34 & -9.71 & 0.25 & -0.19 \\ 34 & 3963 & -2020 & 29.14 & -53.42 \\ -9.71 & -2020 & 1737 & -14.85 & 34.85 \\ 0.25 & 29.14 & -14.85 & 0.21 & -0.39 \\ -0.19 & -53.42 & 34.85 & -0.39 & 3.12 \end{pmatrix}$$

$\mathbf{C}$'s eigenvalues λ: scores, factors

$$\begin{aligned} \lambda_0 &= 5158 \\ \lambda_1 &= 543 \\ \lambda_2 &= 2.3 \\ \lambda_3 &= 0.023 \\ \lambda_4 &= -4.5 \cdot 10^{-16} \end{aligned}$$

Determining Significant Variance

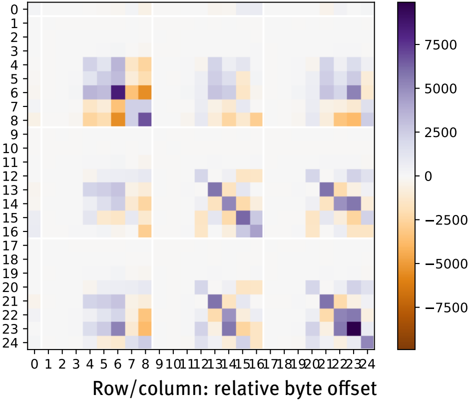
Scree graph of principal components (PCs) sorted by their scores λ_i



significant PCs: prevalent variance in data

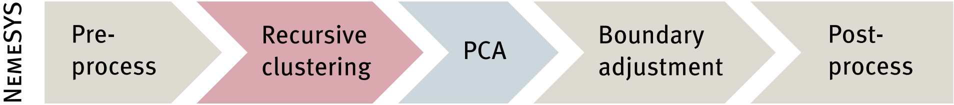
Covariance Matrix for Principal Component Analysis

Covariance matrix $\mathbf{C}$ as heat map



PCA: strengths of linearly dependent variance at all byte offsets in a set

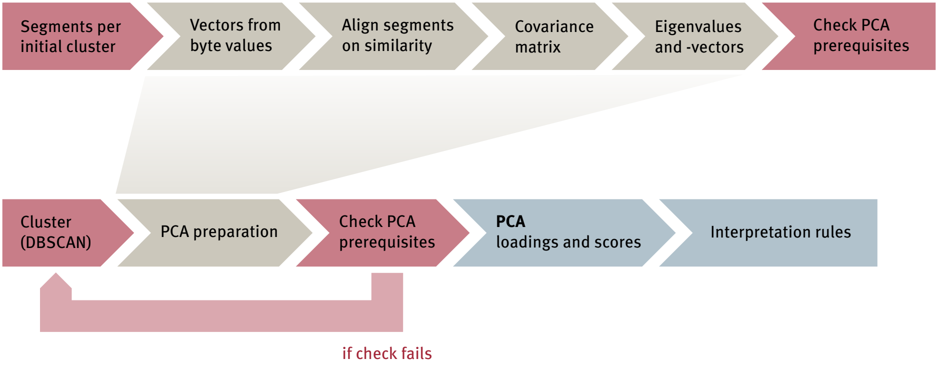
Refinement of NEMESYS: Byte-wise Segment Variance Analysis



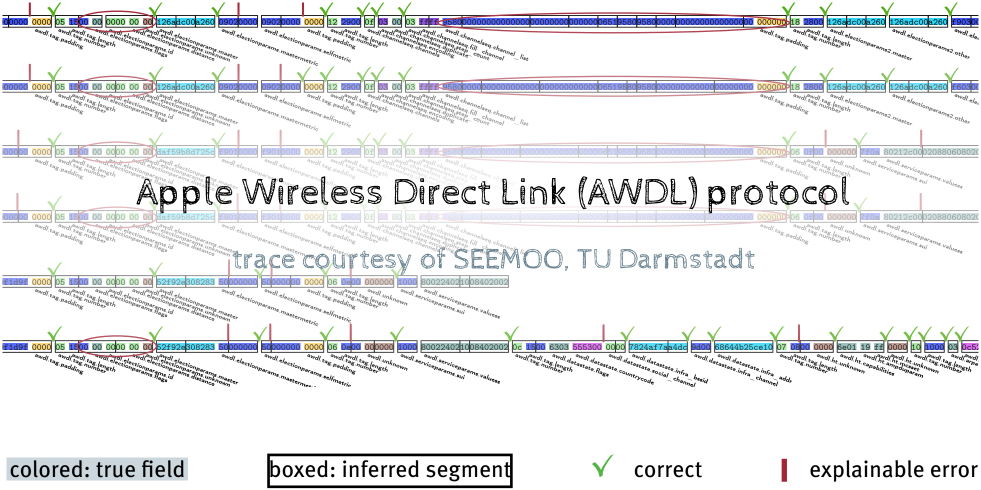
Recursive clustering:

- Ensures application of PCA to a set of related segments

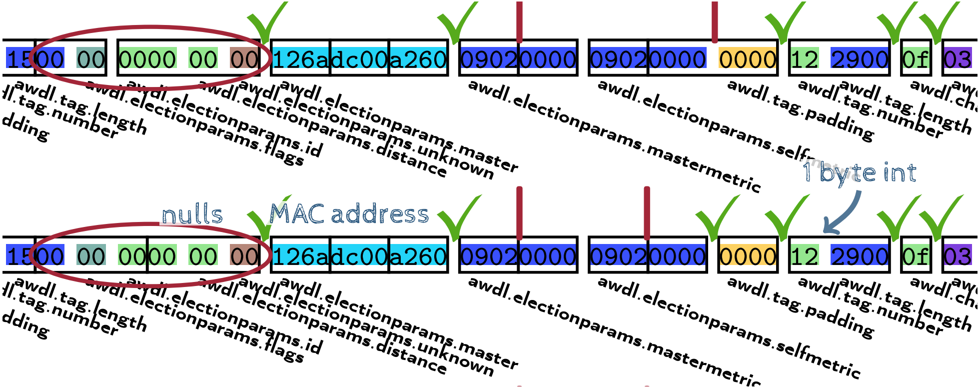
Recursive Clustering



True Fields and NEMESYS-Inferred Segments Interleaved



True Fields and NEMESYS-Inferred Segments Interleaved



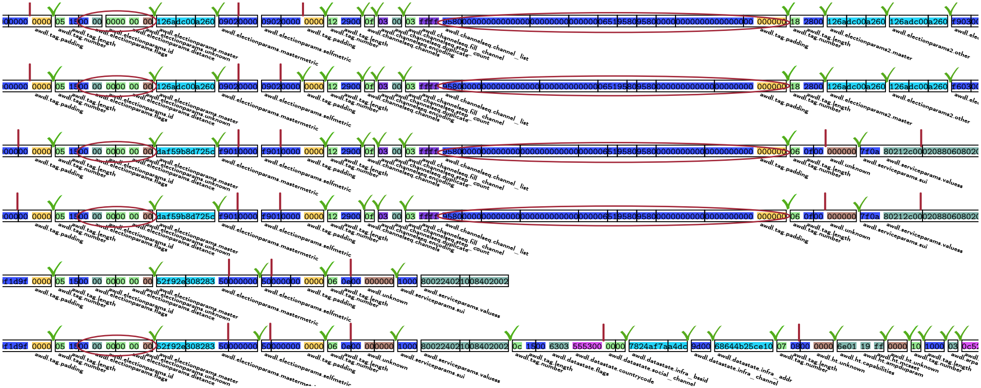
colored: true field

boxed: inferred segment

✓ correct

| explainable error

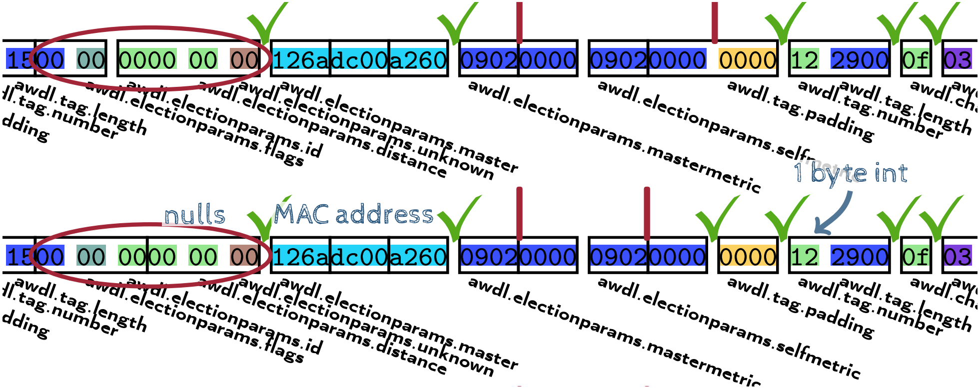
AWDL: Interleave True Fields and Inferred Segments



✓ correct

| explicable error

AWDL: Interleave True Fields and Inferred Segments

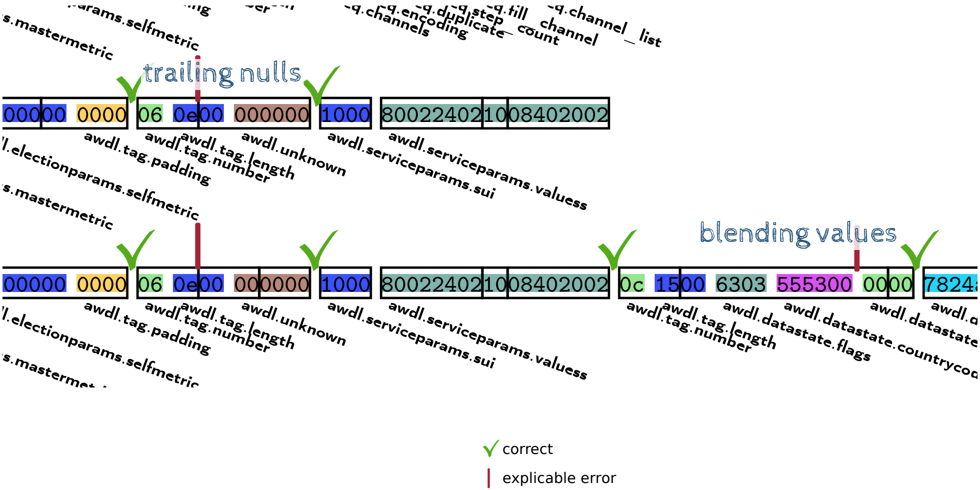


✓ correct

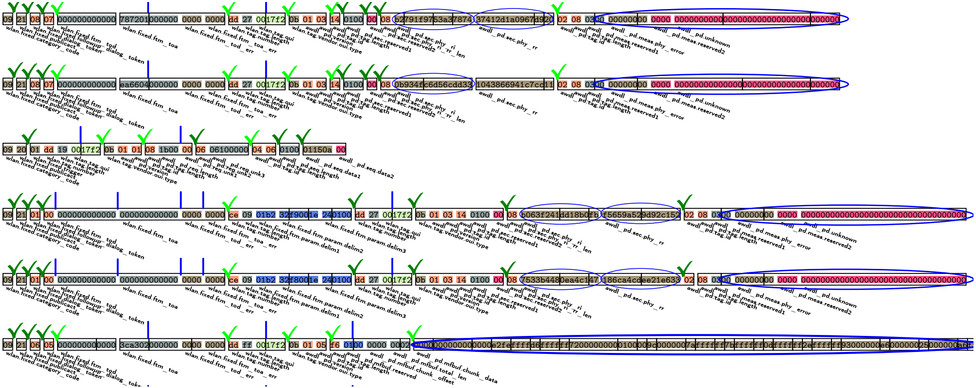
| explicable error

AWDL: Interleave True Fields and Inferred Segments

AWDL: Interleave True Fields and Inferred Segments



AU-WiFi: Interleave True Fields and Inferred Segments



✓ larger structures correct ✓ small structures correct

| difficult/impossible to know otherwise

AU-WiFi: Interleave True Fields and Inferred Segments

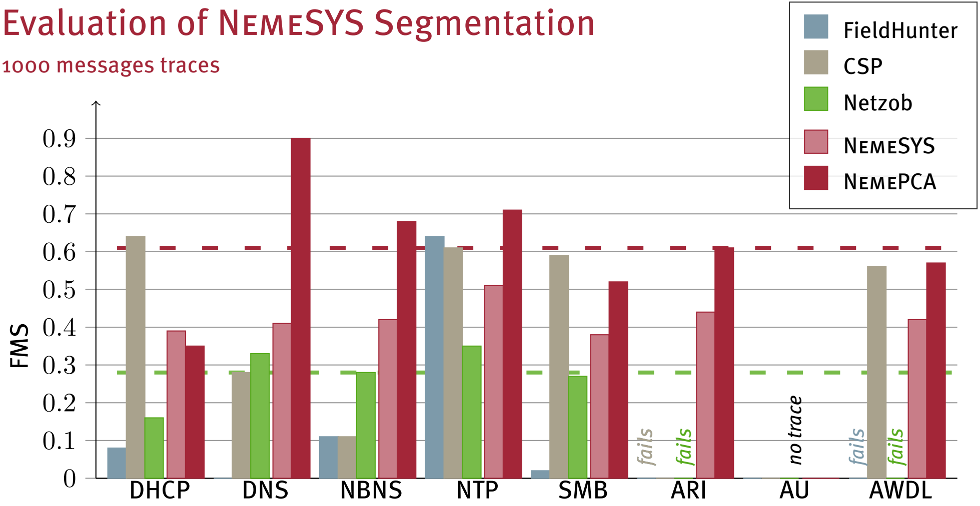
larger structures correct

small structures correct

| difficult/impossible to know otherwise

Evaluation of NEMESYS Segmentation

1000 messages traces



Minimum Canberra Distance d_β vs. Canberra Dissimilarity d_m

$\mathbf{t}$	0208	0008	0208	5706906e
$\mathbf{t}_{[o,\lvert s \rvert]}$	0208	00	08	5706
$\mathbf{s}$	0008	00	07	2700
d_β	0.5	0.000	0.067	0.690

Minimum Canberra Distance d_β vs. Canberra Dissimilarity d_m

$\mathbf{t}$	0208	0008	0208	5706906e
$\mathbf{t}_{[o,\lvert s \rvert]}$	0208	00	08	5706
$\mathbf{s}$	0008	00	07	2700
d_β	0.5	0.000	0.067	0.690
d_m	0.5	0.460	0.496	0.814

Previous Approaches for Comparing Binary Protocol Messages

…	Field 02	Field 03	Field 04	Field 05	Field 06	Field 07	Field 08	Field 09	Field 10	…
	0000000a	0000	80	00	00	00	0000	c0a801	65	
	4f214e45	0000	80	00	00	00	0000	c0a801	66	
	8940fa36	0000	80	00	00	00	0000	c0a801	67	
	a55cb819	0000		00		00	c0a80166	c0a801	66	
	0a4da00f	0000		00		00	c0a80169	c0a801	69	
	8940fa36	0000		00		00		c0a801	6700000000	

- Align on **values** in messages (Netzob, Discoverer)
- Search for tokens to correlate message **values** (PRISMA)

Unsupervised Clustering Algorithm Criteria

1. **Number** of message types/clusters is unknown
2. Stable **auto-**configuration: no **parameter**/threshold to specify by the analyst
3. Performance **efficient** enough to deal with large traces

	1	2	3
Hierarchical Agglomerative, Affinity Propagation		⊖	⊖
Spectral	⊖	⊖	
Single Linkage, Support Vector Machine (SVM)		⊖	
k-means, Partitioning around Medoid (PAM)	⊖		
Density-Based Spatial Clustering of Applications with Noise (DBSCAN)		○	
Hierarchical DBSCAN (HDBSCAN), OPTICS		⊖	

Message Type Discriminators

Cluster refinement by discriminator fields:

- Split underspecific clusters
- Merge overspecific clusters

Examples

AWDL:

MIF and PSF + `awdl.datastate.extflags`

AU-WiFi:

series of `awdl_pd.tag.id` (values: `0x01` to `0x05`) and
`wlan.fixed.ftm.param.status_indication` (values: `0x0`, `0x1`)
which is subfield of `wlan.fixed.ftm.param.delim1`

Introducing Topology Plots[1]

Goal Visualize distances of the clustered segments

Problem Mixed dimensionality (> 3) of feature vectors,
only pairwise pseudo-distances in dissimilarity matrix

[1] Stephan Kleber and Frank Kargl. „Poster: Network Message Field Type Recognition“. In: *Proceedings of the 26th Conference on Computer and Communications Security*. CCS. 2019.

Introducing Topology Plots[1]

Goal Visualize distances of the clustered segments

Problem Mixed dimensionality (> 3) of feature vectors,
only pairwise pseudo-distances in dissimilarity matrix

Multidimensional scaling (MDS)

Place segments as points in an n-dimensional space
according to their relative distances.

Here: $n = 2$ to plot a diagram („Topology of Distances“)

---

[1] Stephan Kleber and Frank Kargl. „Poster: Network Message Field Type Recognition“. In: *Proceedings of the 26th Conference on Computer and Communications Security*. CCS. 2019.

Introducing Topology Plots[1]

no coherent feature vector space

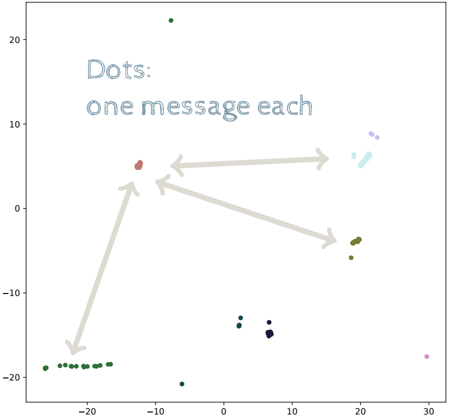
no message coordinates for plot

pairwise dissimilarities ≈ relative distances

absolute positions meaningless

[1] Stephan Kleber and Frank Kargl. „Poster: Network Message Field Type Recognition“. In: *Proceedings of the 26th Conference on Computer and Communications Security*. CCS. 2019.

Introducing Topology Plots[1]

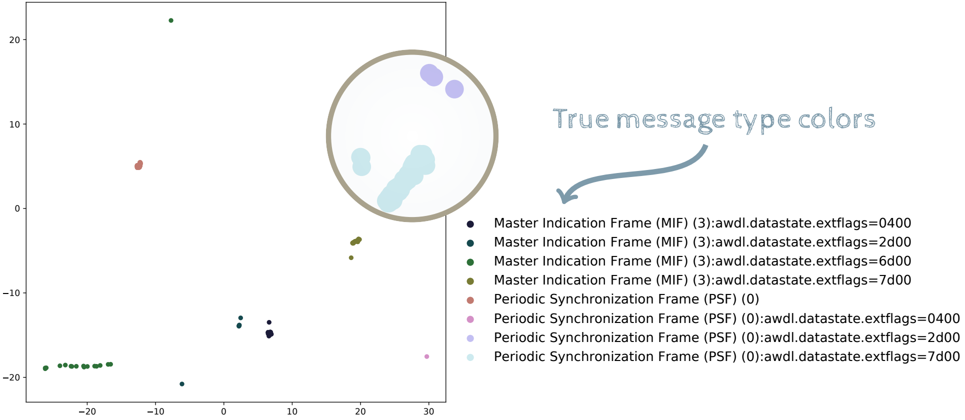


pairwise dissimilarities ≈ relative distances

absolute positions meaningless

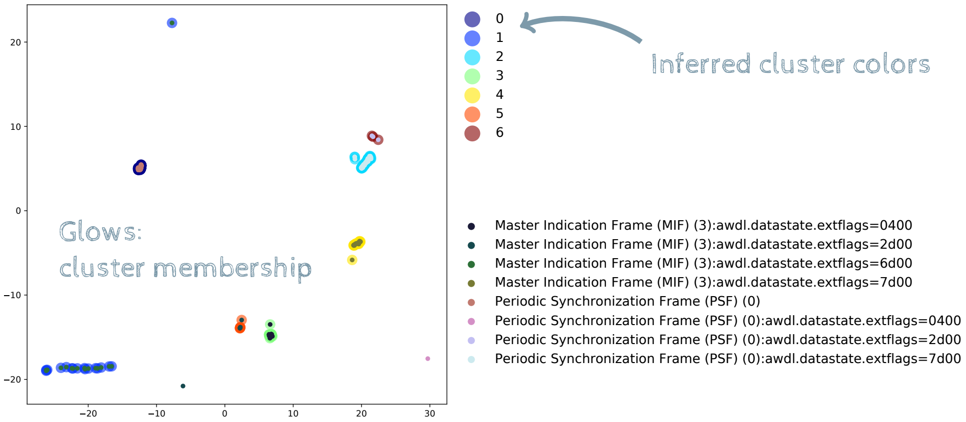
[1] Stephan Kleber and Frank Kargl. „Poster: Network Message Field Type Recognition“. In: *Proceedings of the 26th Conference on Computer and Communications Security*. CCS. 2019.

Introducing Topology Plots[1]



[1] Stephan Kleber and Frank Kargl. „Poster: Network Message Field Type Recognition“. In: *Proceedings of the 26th Conference on Computer and Communications Security*. CCS. 2019.

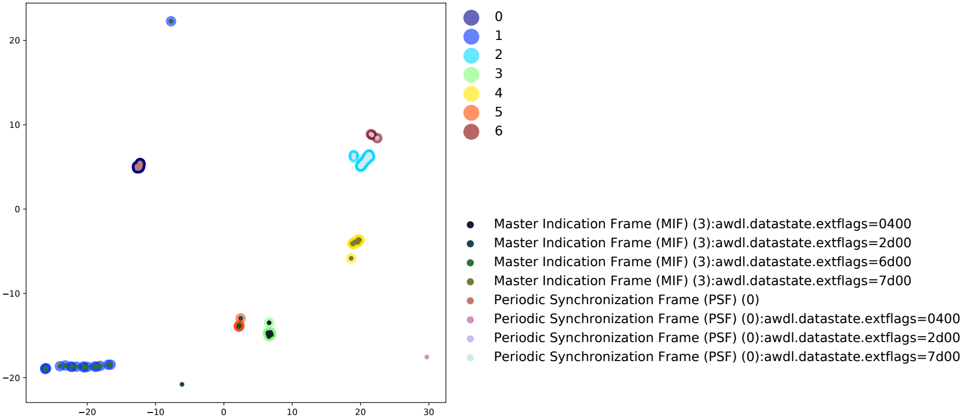
Introducing Topology Plots[1]



[1] Stephan Kleber and Frank Kargl. „Poster: Network Message Field Type Recognition“. In: *Proceedings of the 26th Conference on Computer and Communications Security*. CCS. 2019.

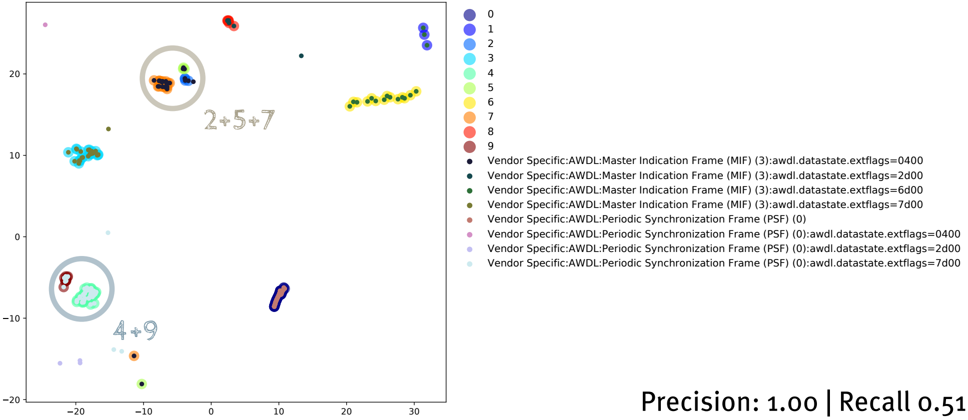
AWDL: Topology Plot of Messages using Groundtruth from Wireshark

Apple Wireless Direct Link protocol



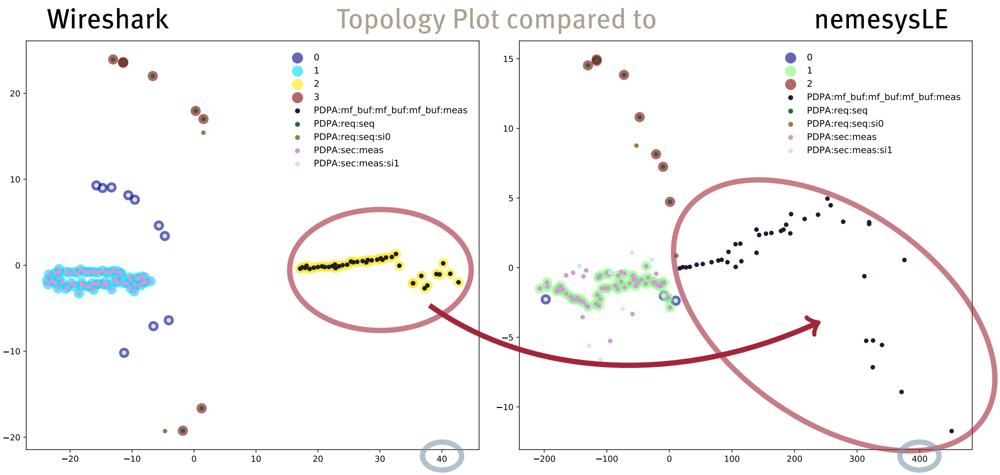
AWDL: Topology Plot of Messages using Segmenter NEMESYS

Apple Wireless Direct Link protocol

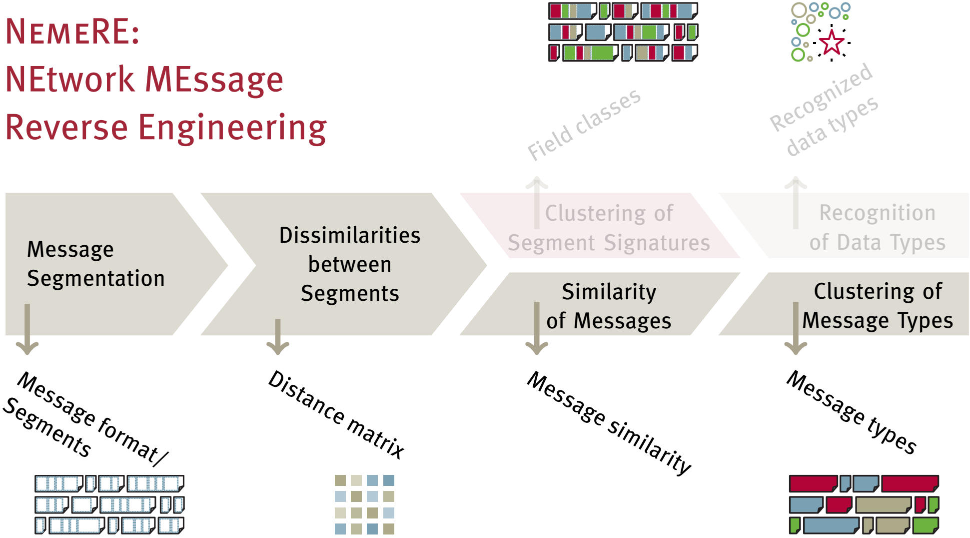


Precision: 1.00 | Recall 0.51

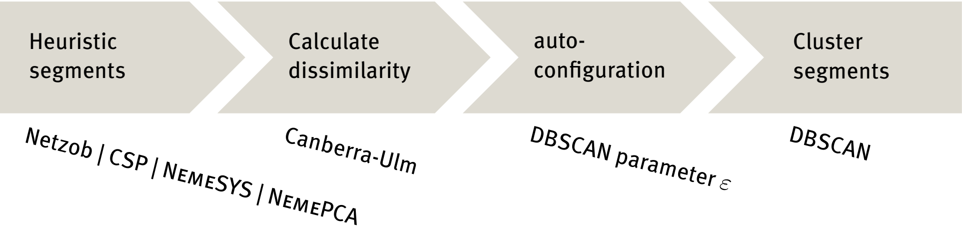
AU-WiFi: NEMETYL- Segmenter: NEMESYS



NemeRE:
NEtwork MEssage
Reverse Engineering



Field Type Classification[1,2]

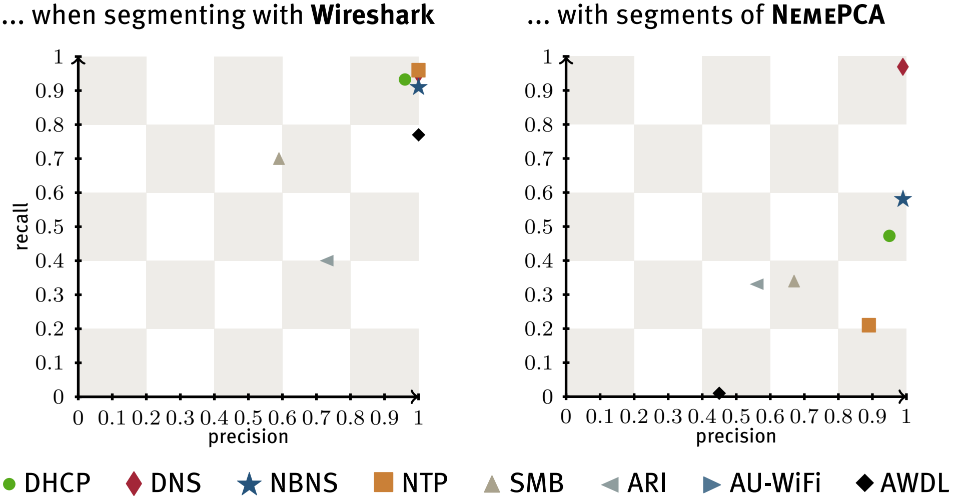


Ground truth: Field data type (e. g., int, timestamp, address) from Wireshark

[1] Stephan Kleber and Frank Kargl. „Poster: Network Message Field Type Recognition“. In: *Proceedings of the 26th Conference on Computer and Communications Security*. CCS. 2019.

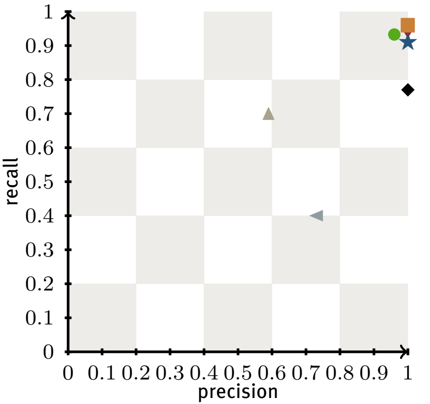
[2] Stephan Kleber et al. „Network Message Field Type Classification and Recognition for Unknown Binary Protocols“. In: *Proceedings of the DSN Workshop on Data-Centric Dependability and Security*. DCDS. IEEE/IFIP, 2022.

Field Type Classification - Clustering Results...

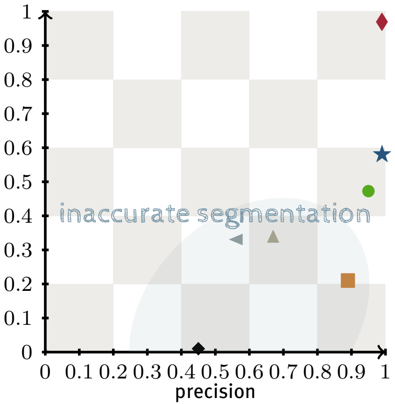


Field Type Classification - Clustering Results...

... when segmenting with **Wireshark**



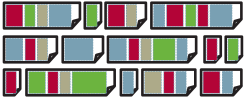
... with segments of **NemePCA**



● DHCP ♦ DNS ★ NBNS ■ NTP ▲ SMB ◀ ARI ▶ AU-WiFi ◆ AWDL

Result of Field Type Classification

Clusters of Segments resembling Field Data Types



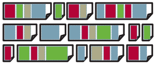
NemeFTR: NEtwork MEssage Field Type Classification

(poster at CCS2019[1]; paper at DCDS2022[2])

[1] Stephan Kleber and Frank Kargl. „Poster: Network Message Field Type Recognition“. In: *Proceedings of the 26th Conference on Computer and Communications Security*. CCS. 2019.

[2] Stephan Kleber et al. „Network Message Field Type Classification and Recognition for Unknown Binary Protocols“. In: *Proceedings of the DSN Workshop on Data-Centric Dependability and Security*. DCDS. IEEE/IFIP, 2022.

NEMERE:
NEtwork MEssage Reverse Engineering

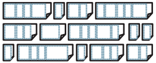


Field classes

Recognized data types

Message Segmentation → Dissimilarities between Segments → Similarity of Messages / Clustering of Segment Signatures → Clustering of Message Types / Recognition of Data Types

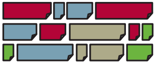
Message format/ Segments



Distance matrix

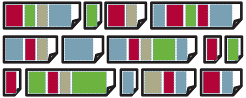
Message similarity

Message types



Result of Field Type Recognition

Recognition of Learned Data Types

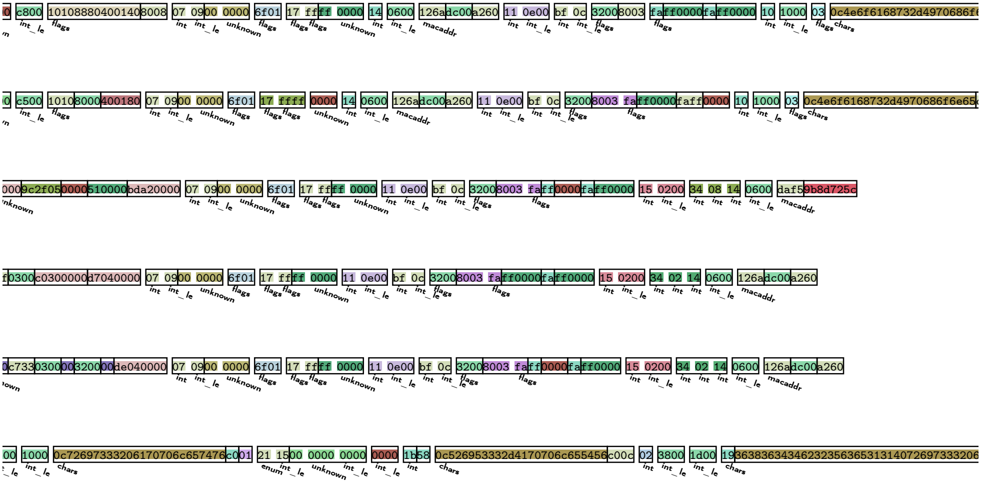


NemeFTR: NEtwork MEssage Field Type Recognition
(poster at CCS2019[1]; paper at DCDS2022[2])

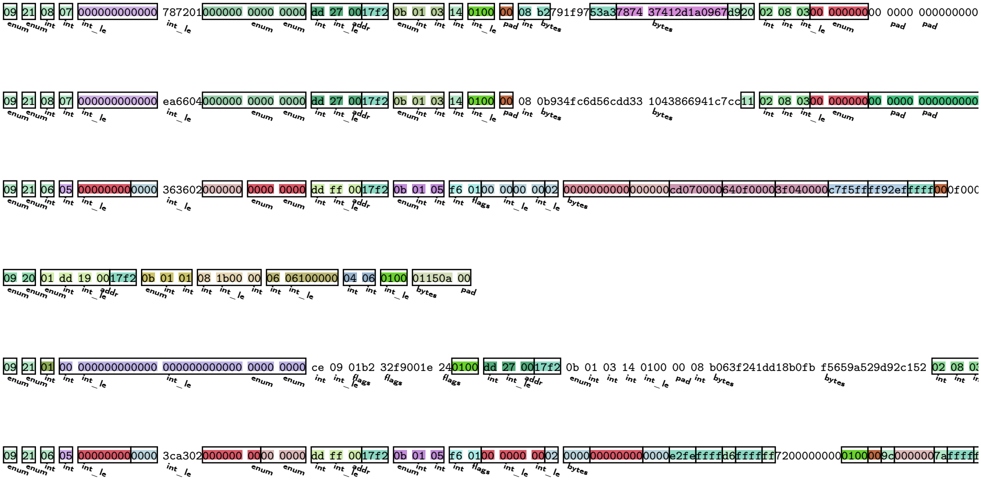
[1] Stephan Kleber and Frank Kargl. „Poster: Network Message Field Type Recognition“. In: *Proceedings of the 26th Conference on Computer and Communications Security*. CCS. 2019.

[2] Stephan Kleber et al. „Network Message Field Type Classification and Recognition for Unknown Binary Protocols“. In: *Proceedings of the DSN Workshop on Data-Centric Dependability and Security*. DCDS. IEEE/IFIP, 2022.

AWDL: Clustered Data Types Marked in Messages

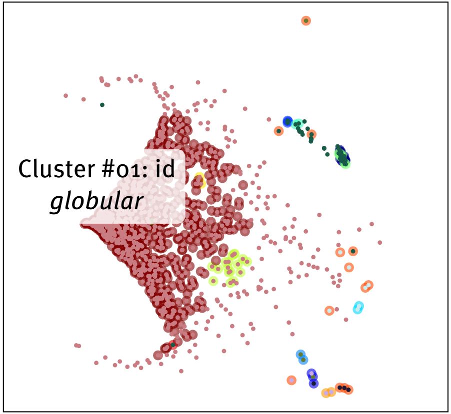


AU-WiFi: Clustered Data Types Marked in Messages



Field Type Classification - Groundtruth: Wireshark

DNS



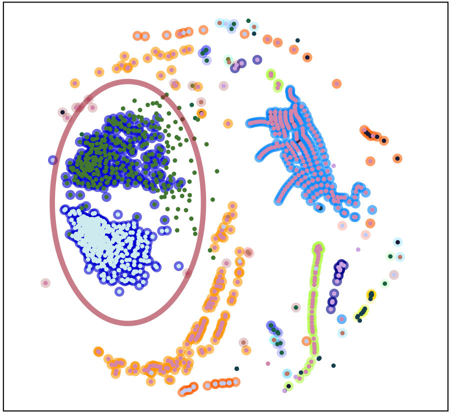
DNS

true data types

- chars
- flags
- id
- int
- ipv4

Field Type Classification - Groundtruth: Wireshark

SMB



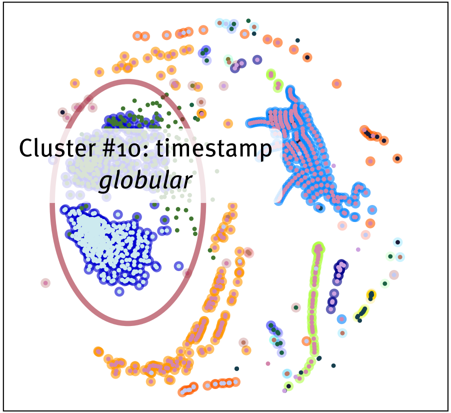
SMB

true data types

- bytes
- chars
- crypto
- enum
- flags
- id
- int
- int-le
- timestamp

Field Type Classification - Groundtruth: Wireshark

SMB



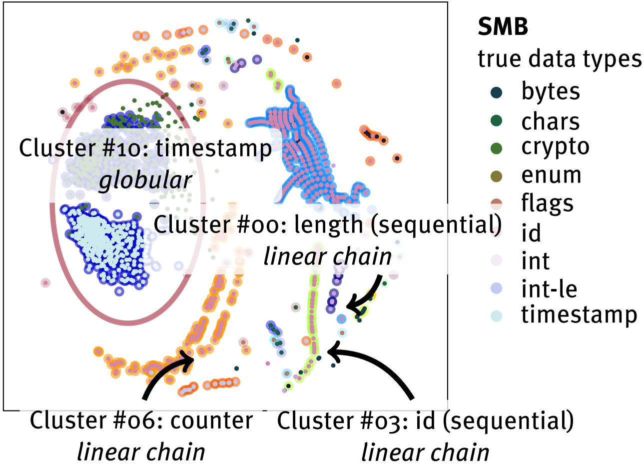
SMB

true data types

- bytes
- chars
- crypto
- enum
- flags
- id
- int
- int-le
- timestamp

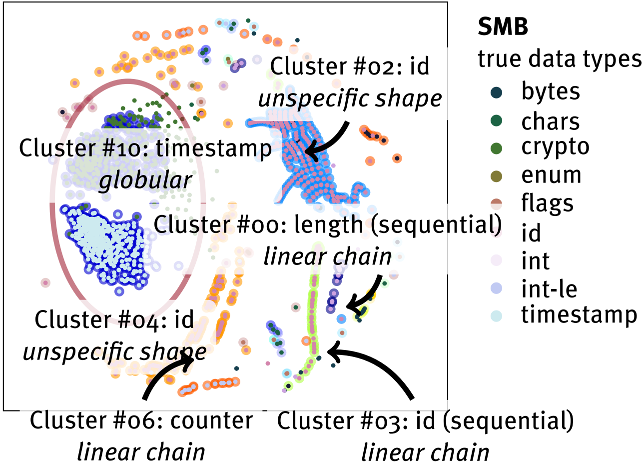
Field Type Classification - Groundtruth: Wireshark

SMB



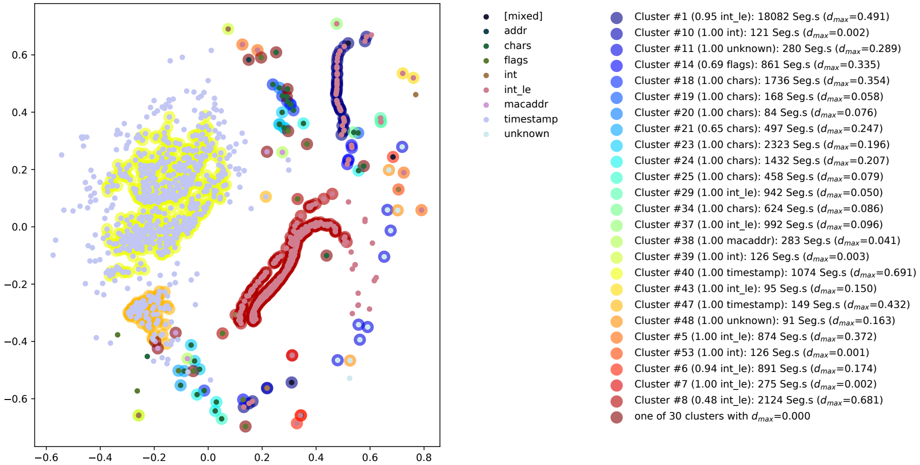
Field Type Classification - Groundtruth: Wireshark

SMB



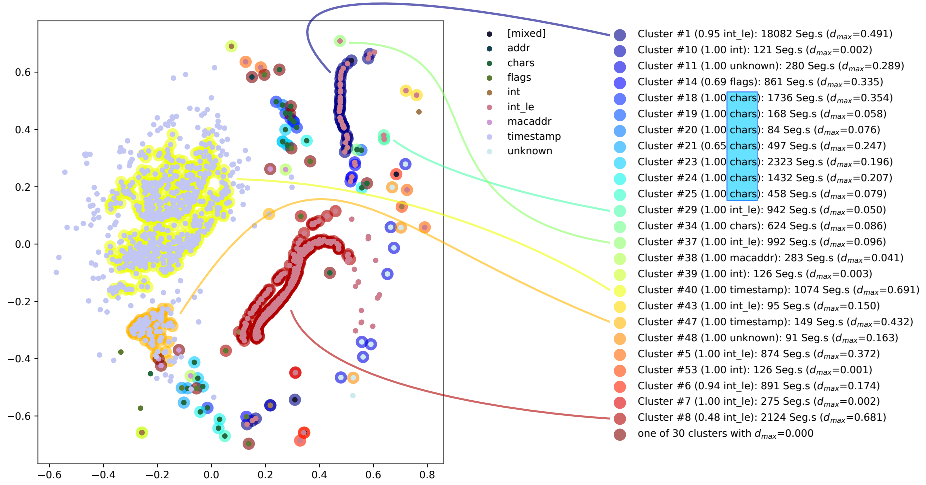
Field Type Classification - Groundtruth: Wireshark

Apple Wireless Direct Link protocol

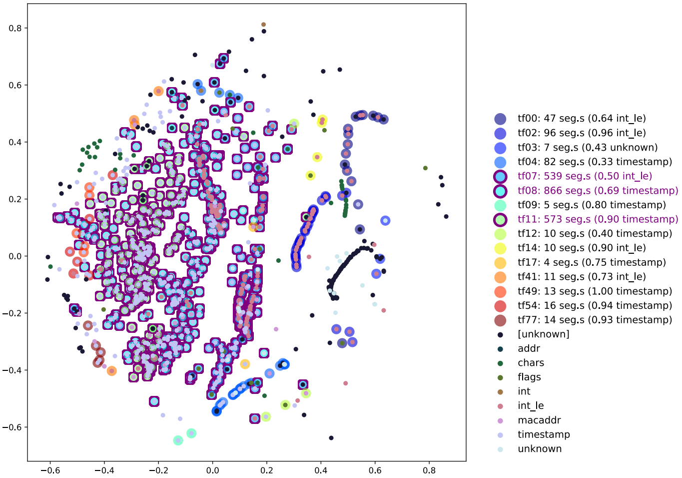


Field Type Classification - Groundtruth: Wireshark

Apple Wireless Direct Link protocol



NEMEFTR with NEMESYS Segments

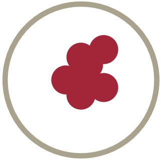


Data-Type-Specific Patterns

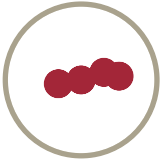
Signatures of Variance: Characteristic feature patterns for data types

- int (BE/LE) — 00 00 15 7b — variance increases towards most significant byte
- chars — 69 44 53 00 — typical ASCII value domain, null-terminated
- id/flags — frequent distinct values
- enumerations

- timestamps

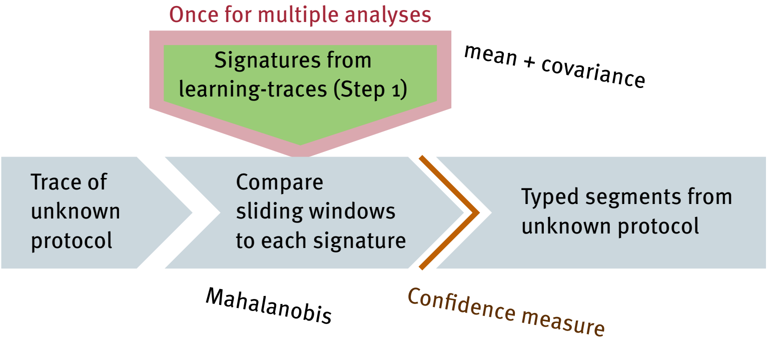


- addresses



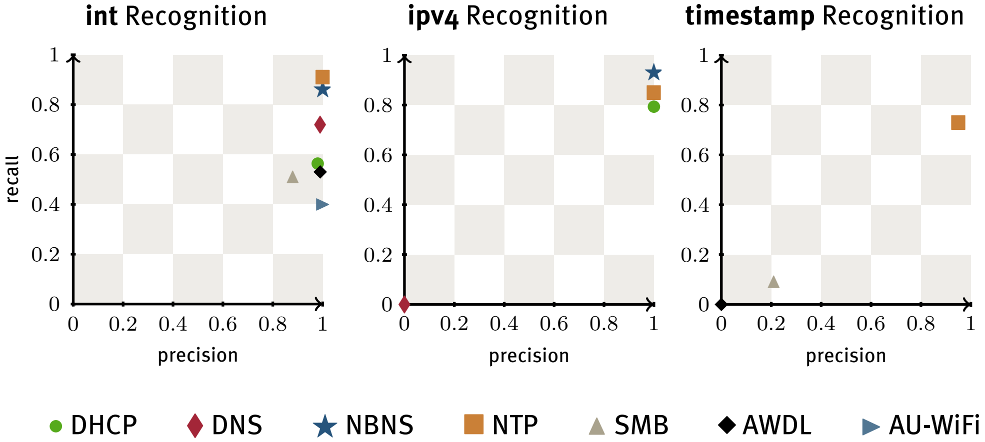
Field Type Recognition Process[1]

- Recognize specific field data types in unseen traces
- Learned from field patterns of a mix of sample protocols



[1] Stephan Kleber and Frank Kargl. „Poster: Network Message Field Type Recognition“. In: *Proceedings of the 26th Conference on Computer and Communications Security*. CCS. 2019.

Field Type Recognition Quality



Byproducts

- Preprocessing for STA: Diversification by value commonality filter and discrimination of textual from binary protocols and parts of mixed protocols
- Data representation in support of PRE
- Character string detection heuristic supporting Unicode
- Enhanced PCAP importer
- JSON parser for tshark dissectors
- Scapy, Wireshark, and Sulley exporters for message formats
- tikz visualization of message formats

- Dynamic Binary Analysis by Automated Architecture-Independent Extraction of Message Formats

Limitations

- Encryption, compression, and obfuscation
- Gracefully deals with embedded text parts, but does not analyze
- Encoding and language (non-western) may prevent text detection
- Heuristic method limits optimum
- Mostly empirical determination of parameters. Robustness thoroughly tested but not provably optimal
- Misinterpretation of structurally similar message and field types
- Memory requirement for dissimilarity matrix
- STA depends on trace contents: Limited by missing and implicit information
- Typically only positive samples observed
- Human involvement for interpretation (message types and data)

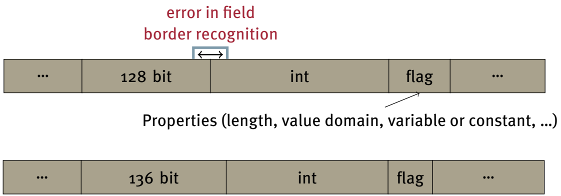
Future Work

- Filter traces for increased variance while retaining valid chronologically sorted message sequences
- Optimizations for NEMESYS: Alternative features like Value Delta, Slit Pivot Bit Congruence
- Performance optimizations: reduce memory consumption of dissimilarity measure
- More sophisticated rule sets for deducing boundaries from relations between principal components
- Alternatives to sequence alignment, e.g., LDA, LSTM
- More fine-grained, robust, and diverse recognition rules for data types
- Supervised learning of cluster properties for unattended recognition by a machine-learning model
- User studies with visual message inspection

Detailed List of Contributions

1. Static Traffic Analysis process model
2. Decomposition of Static Traffic Analysis tools
3. Traffic trace input optimization
4. Clustering topology plots
5. Efficient segmentation: Delta of bit congruence and NEMESYS
6. Canberra-Ulm dissimilarity for comparing sequential binary data
7. Kneedle auto-configuration for DBSCAN
8. Custom auto-configuration for DBSCAN
9. Segmentation refinement by NEMEPCA
10. Message type identification by NEMETYL
11. Field data type clustering
12. Field data type recognition
13. Format Match Score evaluation measure
14. Dynamic Traffic Analysis by PREPROBE

Format Metrics



Quantify Format Inference Quality

Validate format inference method:

Measure correctness by benchmarking with a known protocol

Format Match Score

$$\mathrm{FMS} = \underbrace{\exp\left(-\left(\frac{|R| - |I|}{|R|}\right)^2\right)}_{\text{Specificity penalty}} \cdot \underbrace{\frac{1}{|R|}\sum_{r \in R} \exp\left(-\left(\frac{\delta_r}{\gamma}\right)^2\right)}_{\text{Match gain}}$$

Format Match Score

$$\mathrm{FMS} = \underbrace{\exp\left(-\left(\frac{|R| - |I|}{|R|}\right)^2\right)}_{\text{Specificity penalty}} \cdot \underbrace{\frac{1}{|R|}\sum_{r \in R} \exp\left(-\left(\frac{\delta_r}{\gamma}\right)^2\right)}_{\text{Match gain}}$$

Quality aspects:

$|R|$ Number of real field boundaries

$|I|$ Number of inferred field boundaries

Format Match Score

$$\mathrm{FMS} = \underbrace{\exp\left(-\left(\frac{|R| - |I|}{|R|}\right)^2\right)}_{\text{Specificity penalty}} \cdot \underbrace{\frac{1}{|R|}\sum_{r\in R}\exp\left(-\left(\frac{\delta_r}{\gamma}\right)^2\right)}_{\text{Match gain}}$$

Quality aspects:

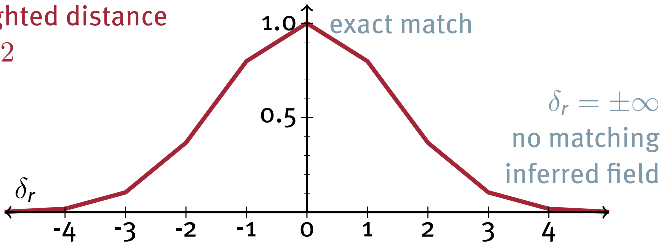
- $|R|$ Number of real field boundaries
- $|I|$ Number of inferred field boundaries
- δ_r Distance of real boundary r from next inferred one
- γ Required accuracy

Format Match Score

$$\mathrm{FMS} = \underbrace{\exp\left(-\left(\frac{|R|-|I|}{|R|}\right)^2\right)}_{\text{Specificity penalty}} \cdot \underbrace{\frac{1}{|R|}\sum_{r\in R}\exp\left(-\left(\frac{\delta_r}{\gamma}\right)^2\right)}_{\text{Match gain}}$$

Weighted distance

$\gamma = 2$



Format Match Score

$$\mathrm{FMS} = \underbrace{\exp\left(-\left(\frac{|R| - |I|}{|R|}\right)^2\right)}_{\text{Specificity penalty}} \cdot \underbrace{\frac{1}{|R|}\sum_{r \in R} \exp\left(-\left(\frac{\delta_r}{\gamma}\right)^2\right)}_{\text{Match gain}}$$

Quantify format correctness

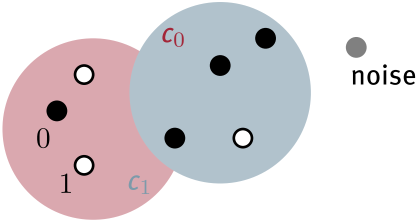
Combinatorial Cluster Quality Measure

Test runs with **known** protocols: Compare to **ground truth**

true message types

message classification

	l	$\bar{l}$
c_l	TP	FP
$c_{\bar{l}}$	FN	TN



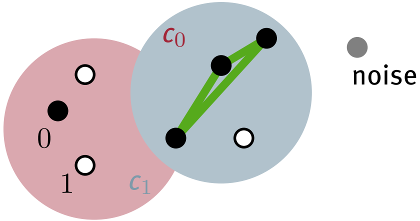
$$P = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad \text{and} \quad R = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Combinatorial Cluster Quality Measure

Test runs with **known** protocols: Compare to **ground truth**

true message types

message classification	l	$\bar{l}$
c_l	TP	FP
$c_{\bar{l}}$	FN	TN



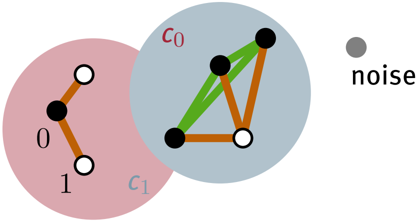
$$P = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad \text{and} \quad R = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Combinatorial Cluster Quality Measure

Test runs with **known** protocols: Compare to **ground truth**

true message types

message classification	l	$\bar{l}$
c_l	TP	FP
$c_{\bar{l}}$	FN	TN



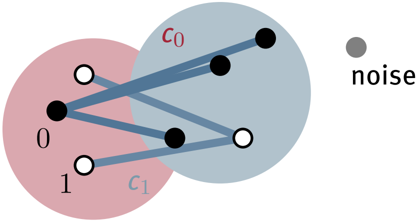
$$P = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad \text{and} \quad R = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Combinatorial Cluster Quality Measure

Test runs with **known** protocols: Compare to **ground truth**

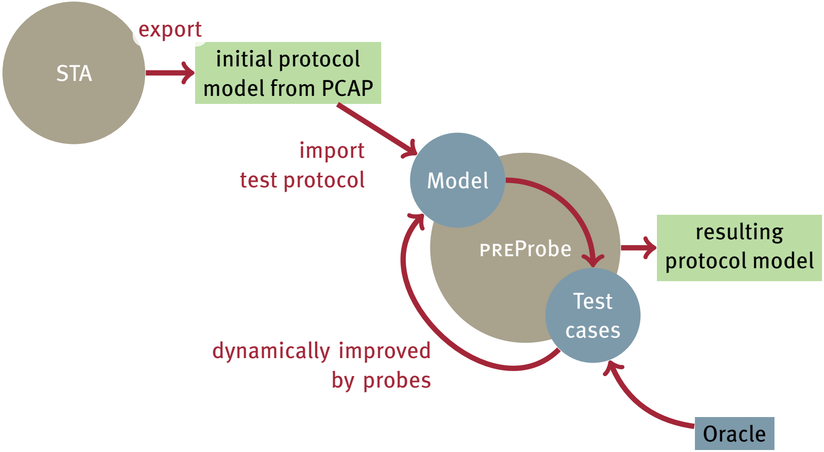
true message types

message classification	l	$\bar{l}$
c_l	TP	FP
$c_{\bar{l}}$	FN	TN



$$P = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad \text{and} \quad R = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Dynamic Traffic Analysis



Dynamic Traffic Analysis: PREPROBE