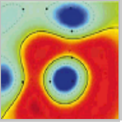


Motivation
PRISMA
Preprocessing
Embedding
Feature Selection
Event Clustering
Markov Model
Templates and Rules
Example
Evaluation
Data Sets
Correctness
Outlook

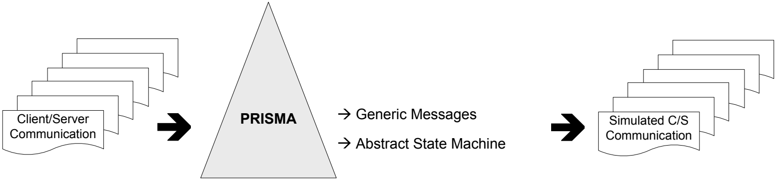
Learning Stateful Models for Network Honeypots

Tammo Krueger
Hugo Gascon[†]
Nicole Krämer[⋆]
Konrad Rieck[†]

TU Berlin, [†]University of Göttingen, [⋆]TU München

10/19/2012
5th ACM Workshop on
Artificial Intelligence and Security

Motivation



Given a pool of client/server communication infer generic messages and abstract state machine

1. To emulate services (honeypots)
2. Lure attackers
3. Gather information about threat potential

Motivation – Protocol Inspection and State Machine Analysis (PRISMA)

Learning Stateful Models for Network Honeypots

Tammo Krueger
Hugo Gascon
Nicole Krämer
Konrad Rieck

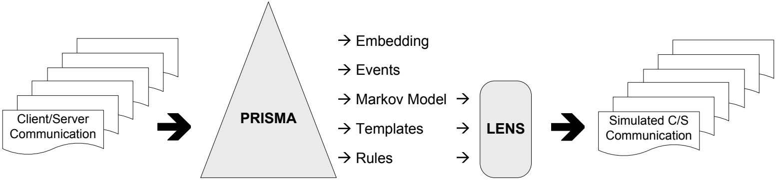
Motivation

PRISMA
Preprocessing
Embedding
Feature Selection
Event Clustering
Markov Model
Templates and Rules

Example

Evaluation
Data Sets
Correctness

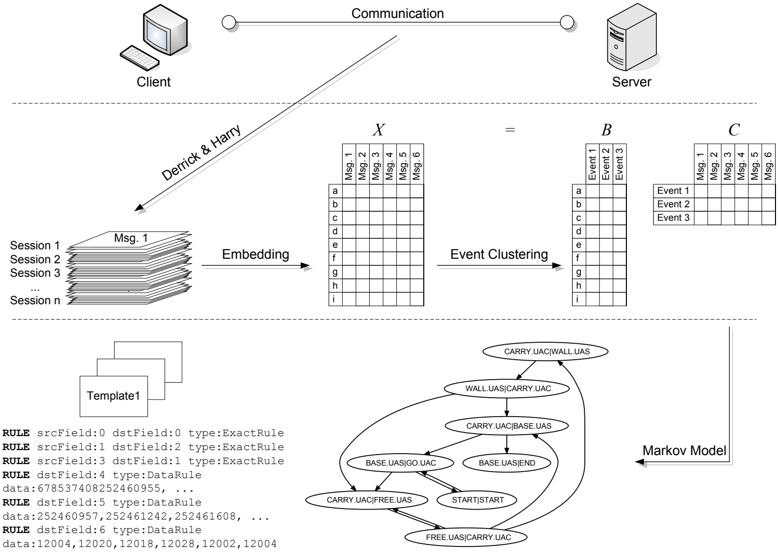
Outlook



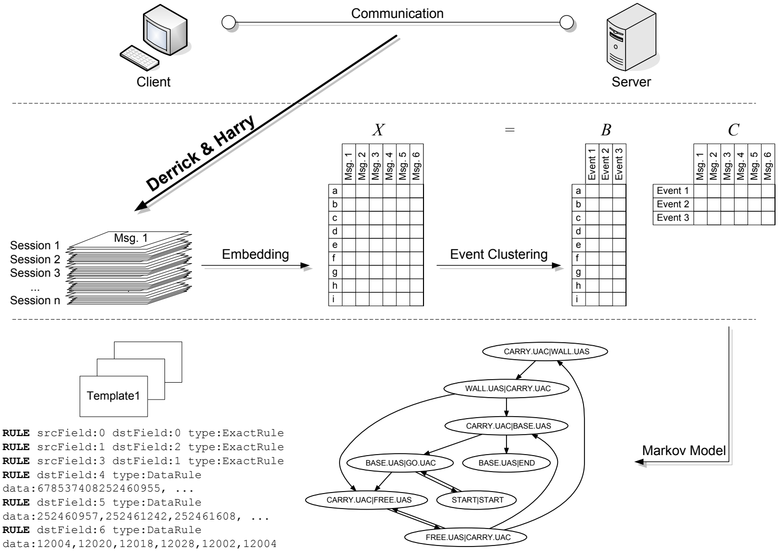
By embedding and event clustering approximate abstract state machine and message types:

- Infer *Markov model* of the behavior
- Find inherent structure of the messages (*templates*)
- Gather information flow between states (*rules*)

System Overview



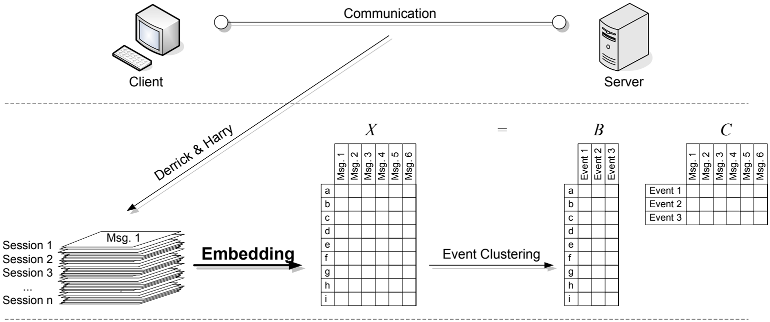
System Overview – Preprocessing

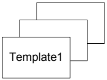


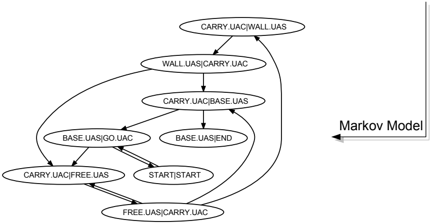
Preprocessing

- Data acquisition via `tcpdump`
- Tool chain needed, to process these binary dump files
  - **Derrick** reassembles packets (removes IP fragmentation)
  - **Harry** concatenates packets to messages and extracts session information
- Data available for the next steps:
  1. messages as sequence of bytes (input for *embedding*)
  2. sessions as sequence of messages (input for *Markov model*)

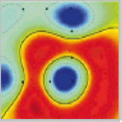
System Overview – Embedding








```
RULE srcField:0 dstField:0 type:ExactRule
RULE srcField:1 dstField:2 type:ExactRule
RULE srcField:3 dstField:1 type:ExactRule
RULE dstField:4 type:DataRule
data:678537408252460955, ...
RULE dstField:5 type:DataRule
data:252460957,252461242,252461608, ...
RULE dstField:6 type:DataRule
data:12004,12020,12018,12028,12002,12004
```



Embedding

1. **N-grams:** Given the set of all possible n-grams over byte sequences $S = \{0, \dots, 255\}^n$, we define the embedding function $\phi : \{0, \dots, 255\}^\star \mapsto \mathbb{R}^{|S|}$ as

$$\phi(x) = (\phi_s(x))_{s \in S} \quad \text{with} \quad \phi_s(x) = \text{occ}_s(x).$$

Example ($n = 3$):

$$\phi(\text{"Hello"}) = (0, \dots, \overset{\text{Hel}}{1}, \overset{\text{ell}}{1}, \overset{\text{llo}}{1}, \dots, 0)^T \in \mathbb{R}^{16777216}$$

2. **Tokens:** Given a set of separators *Sep* we can split the byte sequence into tokens; example ($Sep = \{\text{␣}\}$):

$$\phi(\text{"We'll meet again"}) = (0, \dots, \overset{\text{We'll}}{1}, \overset{\text{meet}}{1}, \overset{\text{again}}{1}, \dots, 0)^T \in \mathbb{R}^?$$

Feature Selection via Statistical Testing

- Some features do not carry real information:
  - Fixed, *constant* tokens (protocol, e.g. `HTTP/1.1`)
  - Random, *volatile* tokens (cookies, nonces, ...)
- Focus the analysis by splitting the feature set F:

$$F = F_{constant} \cup F_{variable} \cup F_{volatile}$$

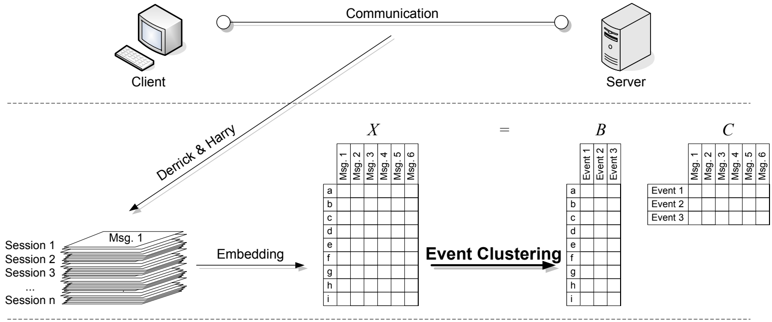
- Calculate frequency π_f and test:

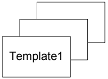
$$\begin{aligned} p_{constant} &= binom.test(H_0 : \pi_f \approx 1.0) \\ p_{volatile} &= binom.test(H_0 : \pi_f \approx 0.0) \end{aligned}$$

- Adjust significance level α for multiple testing
- Keep *variable* features, which are not *constant* **and** are not *volatile*: $p_{constant} \leqslant \hat{\alpha} \wedge p_{volatile} \leqslant \hat{\alpha}$

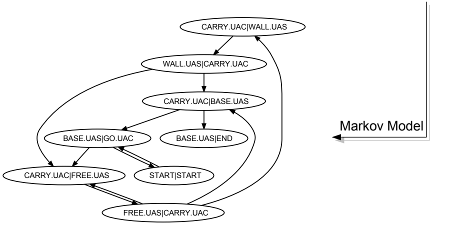
Motivation
PRISMA
Preprocessing
Embedding
Feature Selection
Event Clustering
Markov Model
Templates and Rules
Example
Evaluation
Data Sets
Correctness
Outlook

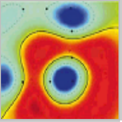
System Overview – Event Clustering






```
RULE srcField:0 dstField:0 type:ExactRule
RULE srcField:1 dstField:2 type:ExactRule
RULE srcField:3 dstField:1 type:ExactRule
RULE dstField:4 type:DataRule
data:678537408252460955, ...
RULE dstField:5 type:DataRule
data:252460957,252461242,252461608, ...
RULE dstField:6 type:DataRule
data:12004,12020,12018,12028,12002,12004
```





Event Clustering

- Clustering as factorization of embedding matrix $X \in \mathbb{R}^{f,N}$ with $B \in \mathbb{R}^{f,e}$, $C \in \mathbb{R}^{e,N}$, $b_i \in \mathbb{R}^{f,1}$, $c_i \in \mathbb{R}^{e,1}$, $\underline{e \ll f}$:

$$X \approx BC = \overbrace{\begin{bmatrix} b_1 & \dots & b_e \end{bmatrix}}^{\text{event basis}} \underbrace{\begin{bmatrix} c_1 & \dots & c_N \end{bmatrix}}_{\text{event assignments}}$$

  via *Non-Negative Matrix Factorization*:

$$\begin{aligned}(B, C) \quad &= \quad \operatorname*{arg\,min}_{B,C} \|X - BC\| \\ & \quad\quad \text{s.t. } b_{ij} \geqslant 0,\ c_{jn} \geqslant 0\,.\end{aligned}$$

- Optimized, replicate-aware NMF which works on duplicate-free $\widetilde{X}$
- Other techniques (e.g. hierarchical clustering, expert knowledge) can be incorporated easily

Event Clustering – Replicate-Aware NMF

function $\textsc{nmfWithReplicates}(\widetilde{X}, e, B, W)$
    $err = \infty$
    while $|err - \frac{1}{2}\|\widetilde{X} - BC\|^2| < \epsilon$ **do**
        $err = \frac{1}{2}\|\widetilde{X} - BC\|^2$
        $\lambda = \mathrm{RRbyCV}(\widetilde{X}, B, I_{e,e})$
        $C = (B^\top B + \lambda I_{e,e})^{-1}(B^\top \widetilde{X})$
        $C[C < 0] = 0$ ▷ Set all negative coordinates to 0
        $\lambda = \mathrm{RRbyCV}(\widetilde{X}^\top, C^\top, W)$
        $B = (\widetilde{X} W C^\top)(C W C^\top + \lambda I_{e,e})^{-1}$
        $B[B < 0] = 0$ ▷ Set all negative coordinates to 0
        $B = B\,\mathrm{diag}(1/\|b_1\|, 1/\|b_2\|, \ldots, 1/\|b_e\|)$
        $C = C\,\mathrm{diag}(\|b_1\|, \|b_2\|, \ldots, \|b_e\|)$

Event Clustering – NMF Tricks

- Some useful tricks:
  - Initialize the B matrix with positive and negative components of a replicate-aware PCA:

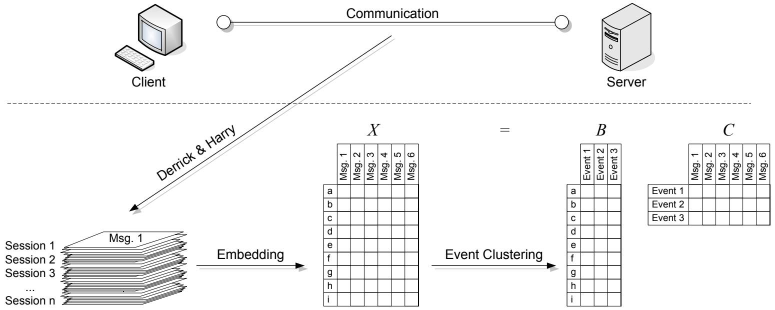
$$
\begin{aligned}
b_i &= \arg\max_{\|b\|=1} \operatorname{var}\left(\widetilde{X}^\top b\right) \\
&\quad \text{s.t. } b \perp b_j,\ j < i
\end{aligned}
$$

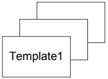
  - $\mathrm{RRbyCV}(Y, Z, W)$ estimates the optimal λ parameter for the ridge regression problem

$$
\arg\min_{\beta} \|Y - Z\beta\|_W^2 + \lambda\|\beta\|^2
$$

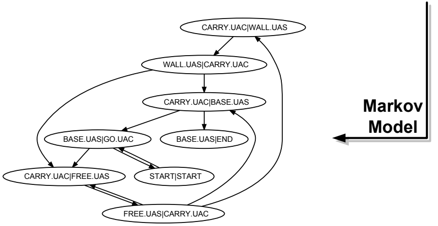
    $\rightarrow$ efficient leave-one-out cross-validation possible!
  - Estimating the inner dimension by simulated noise model
- Implemented in CRAN package PRISMA

System Overview – Markov Model






```
RULE srcField:0 dstField:0 type:ExactRule
RULE srcField:1 dstField:2 type:ExactRule
RULE srcField:3 dstField:1 type:ExactRule
RULE dstField:4 type:DataRule
data:678537408252460955, ...
RULE dstField:5 type:DataRule
data:252460957,252461242,252461608, ...
RULE dstField:6 type:DataRule
data:12004,12020,12018,12028,12002,12004
```



Markov Model

Motivation

PRISMA
Preprocessing
Embedding
Feature Selection
Event Clustering
Markov Model
Templates and Rules

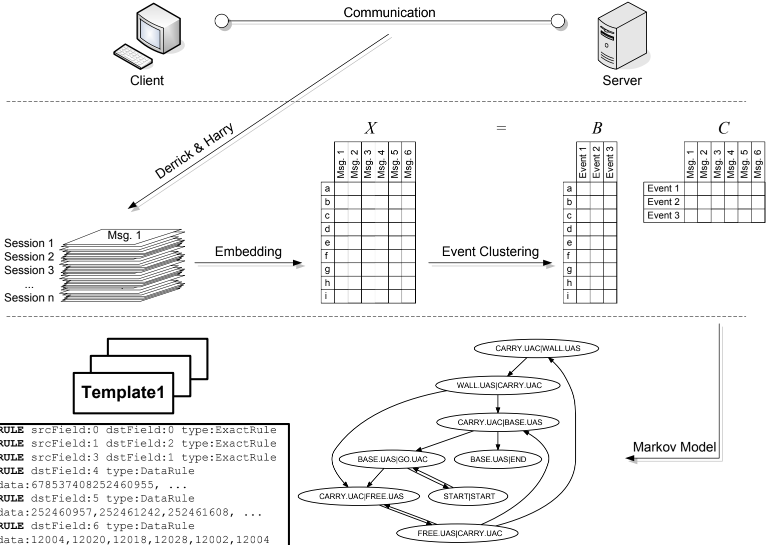
Example

Evaluation
Data Sets
Correctness

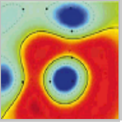
Outlook

- Each message is assigned to an *event* from the event space E, so a session $S = [e_1, e_2, \ldots e_{|S|}], e_{1,2,\ldots,|S|} \in E$
- Represent the dynamics for the system by a Markov model of order $k \geqslant 2$:
  1. Estimate the frequencies of the initial events (i.e. $P(e), e \in E$)
  2. Estimate the frequencies of an event given the m predecessors in time (i.e. $P(e_t | e_{t-k}, \ldots, e_{t-2}, e_{t-1})$)
- Resulting networks can be big (potentially $|E|^k$ nodes):
  - Markov model can be transformed in a DFA
  - Compress structure via DFA minimization algorithm

System Overview – Templates and Rules




```
RULE srcField:0 dstField:0 type:ExactRule
RULE srcField:1 dstField:2 type:ExactRule
RULE srcField:3 dstField:1 type:ExactRule
RULE dstField:4 type:DataRule
data:678537408252460955, ...
RULE dstField:5 type:DataRule
data:252460957,252461242,252461608, ...
RULE dstField:6 type:DataRule
data:12004,12020,12018,12028,12002,12004
```



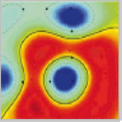
Templates and Rules

	State A_S	State B_C	State C_S
Session 1	`ftp 3.14`	`USER anon`	`331 User anon ok`
Session 2	`ftp 3.12`	`USER ren`	`331 User ren ok`
	⋮	⋮	⋮
Session n	`ftp 2.0`	`USER liz`	`331 User liz ok`
Template	`ftp` $\square$	`USER` $\square$	`331 User` $\square$ `ok`

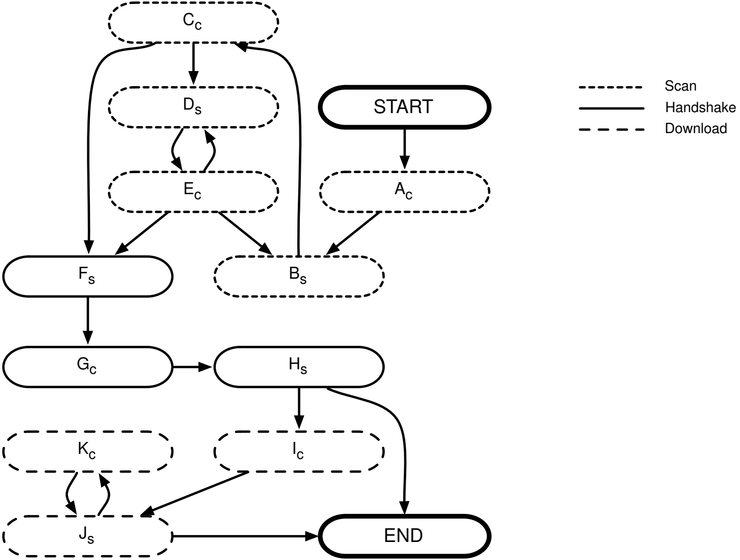
- Template generation:
  - Assign each message to its corresponding state
  - Align messages and find static and changing parts (*fields*)
- Rules between templates:
  - **Copy** Exact copy of one field to another.
  - **Seq.** Copy of a numerical field incremented by d.
  - **Add** Copy field and add data d to the front or back.
  - **Part** Copy front/back part of a field splitted by separator s.
  - **Data** Fill field with data d which we have seen before.

Example – Koobface

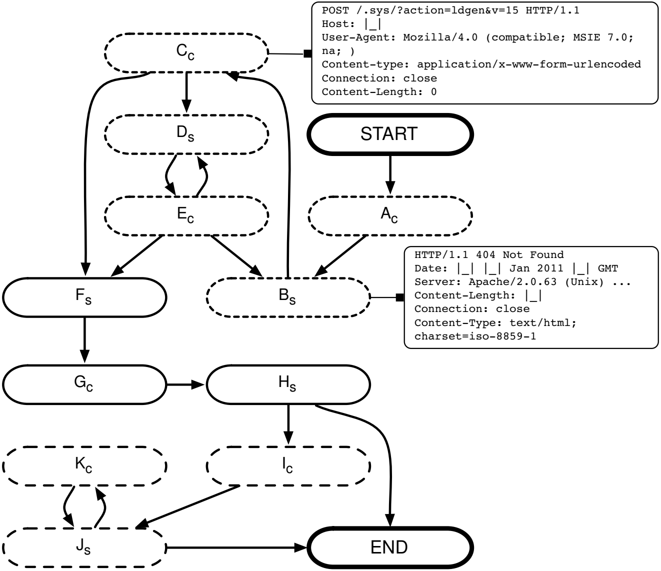
- Data from G. Jacob et al. *Jackstraws: Picking command and control connections from bot traffic*. USENIX 2011
- Extraction of C&C communication via dynamic taint analysis
- Network traffic of one class of malware:
  - 147 sessions
  - 6,674 messages
- Model learning:
  - Token embedding
  - Replicate-aware NMF
  - Markov model $k = 2$



Example – Koobface Model



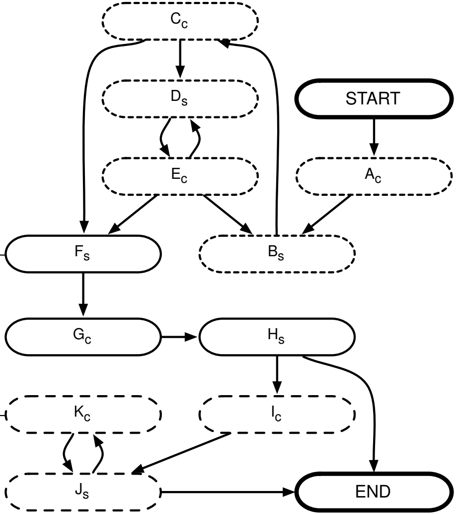
Example – Koobface Scanning



Example – Koobface C&C Server Found and Downloading


```
HTTP/1.1 200 OK
Date: |_| |_| Jan 2011 |_| GMT
Server: Apache/2.2.9 (Debian) DAV/2 mod_ssl/2.2.9 ...
Content-Type: text/html
#BLACKLABEL
#GEO=FR
...
STARTONCE|http://www.xx.com/.sys/?getexe=go.exe
STARTONCE|http://www.xx.com/.sys/?getexe=fb.76.exe
...
START|http://www.xx.com/.sys/?getexe=v2captcha.exe
START|http://www.xx.com/.sys/?getexe=v2googlecheck.exe
#CACHE
MD5|ffd6c11a8dde1687943d4a53021ae9ca
#SAVED 2009-12-11 04:00:28
```

```
GET |_| HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 ...
Host: www.xx.com
Connection: Keep-Alive
```



Evaluation – Data Sets and Models

	Msgs.	(kept)	Feat.	(kept)	# nodes (opt.)
SIP	34,958	2.6%	72,937	0.4%	148 (100)
DNS	5,539	35.6%	6,625	13.2%	381 (153)
FTP	1,760,824	0.2%	87,140	2.2%	1,305 (653)

- **SIP:** 7 days/20 users of telephony data
- **DNS:** Domain Name System requests of 7 devices inside a home network
- **FTP:** 10 days of File Transfer Protocol data set from the Lawrence Berkeley National Laboratory
- Train on 90% of the sessions and use rest for testing
- Evaluation of *completeness* shows that model is capable of generating content of hold-out sessions

Evaluation – Correctness

- Test *correctness* of the models:
  - Check syntactical (`wireshark`) and semantical features of the simulated sessions
  - Unidirectional simulation uses one side of the hold-out sessions and simulates the other side
  - Bidirectional simulation in which both sides of the session are simulated (`M-x psychoanalyze-pinhead`)
- Check session semantic:
  - **SIP:** the `CallID`, `from`- and `to`-tag are preserved
  - **DNS:** If message is a reply check whether it was queried before and has the same query id
  - **FTP:** check, whether request and the returned reply code is valid according to the RFCs (959, 3659, 775, 2389)

Evaluation – Correctness

	Syntax		Semantic	
	Unidir.	Bidir.	Unidir.	Bidir.
SIP	100.0%	100.0%	98.8%	94.5%
DNS	100.0%	100.0%	100.0%	99.4%
FTP	99.9%	82.1%	93.4%	57.6%

- Apply syntax and semantic check on sessions
- Count sessions which consist solely of syntactical and semantical correct messages

Motivation

PRISMA
Preprocessing
Embedding
Feature Selection
Event Clustering
Markov Model
Templates and Rules

Example

Evaluation
Data Sets
Correctness

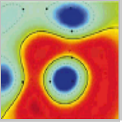
Outlook

Conclusion and Future Work

- Protocol Inspection and State Machine Analysis:
  1. Embed messages in a suitable vector space
  2. Transform sequences of messages to a sequence of *events*
  3. Learn the event machine with a *Markov model*
- Application as "Honey-Service"
- CRAN package PRISMA (feature selection and replicate-aware NMF/PCA)
- Future work:
  - Package Markov model learner for download
  - Apply PRISMA in the context of stateful anomaly detection and deep fuzzing

Questions? Remarks?

Thanks for your attention!



Evaluation – Models

- Markov model properties:

	# nodes	Coverage	Min. DFA	Coverage
SIP	148	14.5%	100	9.8%
DNS	381	0.8%	153	0.3%
FTP	1,305	0.8%	653	0.4%

- Number and type of rules:

	Copy	*Seq.*	*Add*	*Part*	*Data*	Total
SIP	1,916	77	135	52	1,793	3,972
DNS	3,142	4	0	0	3,527	6,673
FTP	532	18	253	35	4,671	5,509

Evaluation – Completeness

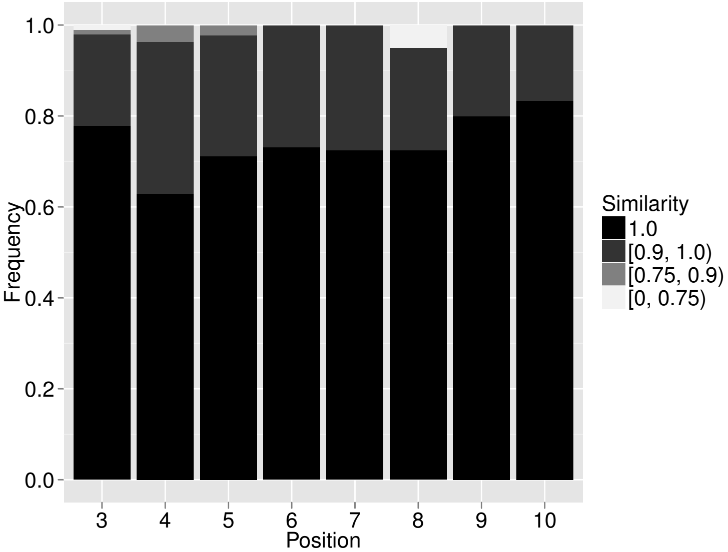
Tammo Krueger
Hugo Gascon
Nicole Krämer
Konrad Rieck

Motivation
PRISMA
Preprocessing
Embedding
Feature Selection
Event Clustering
Markov Model
Templates and Rules
Example
Evaluation
Data Sets
Correctness
Outlook

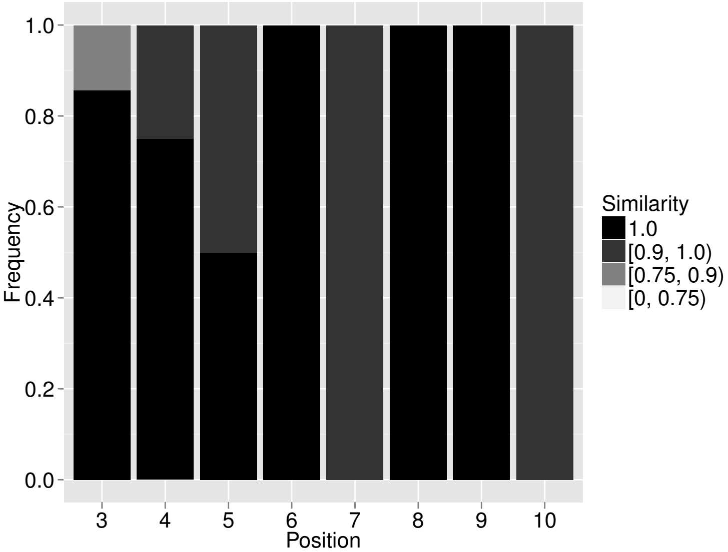
Test *completeness* of the models, i.e., is the learned model capable of replaying sessions as observed in the data pool:

- Comparison against the held-out sessions
- Replay 100 times both from the client and server perspective (unidirectional simulation)
- Calculate normalized edit distance for each generated message

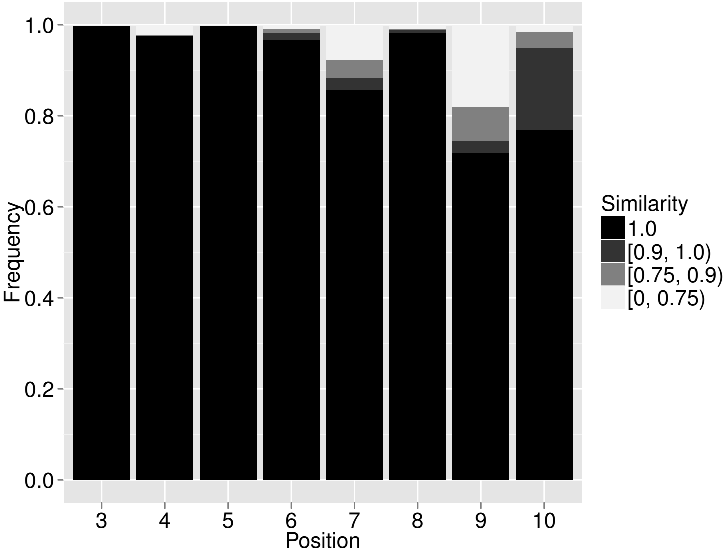
Evaluation – Completeness SIP



Evaluation – Completeness DNS



Evaluation – Completeness FTP



Evaluation – Correctness

Learning Stateful Models for Network Honeypots

Tammo Krueger
Hugo Gascon
Nicole Krämer
Konrad Rieck

Motivation

PRISMA
Preprocessing
Embedding
Feature Selection
Event Clustering
Markov Model
Templates and Rules

Example

Evaluation
Data Sets
Correctness

Outlook

- Breakdown of cumulative syntactical and semantical correctness of sessions for the FTP data:

Msgs. Correct	Syntax Unidir.	Syntax Bidir.	Semantic Unidir.	Semantic Bidir.
100%	0.999	0.821	0.934	0.576
$> 90\%$	1.000	0.953	0.988	0.878
$> 80\%$	1.000	0.996	1.000	0.982