Soutenance de thèse de l'Université Henri Poincaré Nancy 1

Robustesse et Identification des Applications Communicantes

–

Robustness and Identification of Communicating Applications

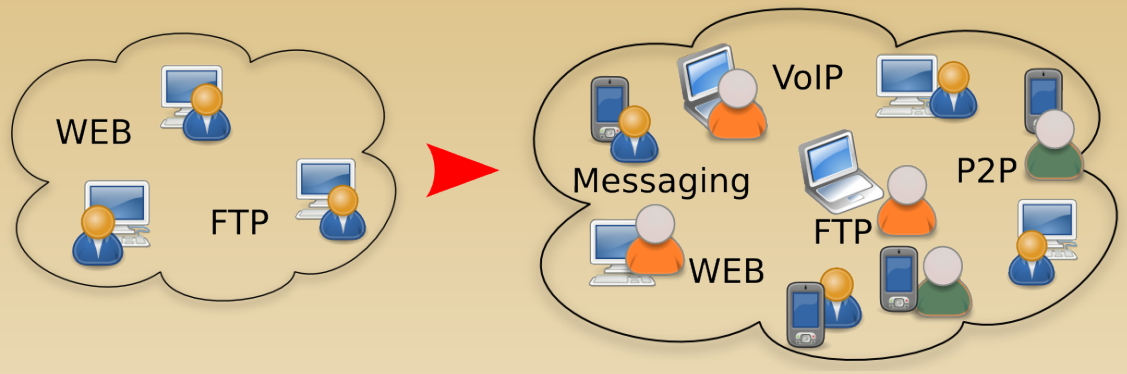
Jérôme François
sous la direction d'Olivier Festor et Radu State

Outline

Outline

Outline

Context



- Internet / networks growth (users, devices, protocols)
- emerging needs
  - New applications development
    - new protocol / reuse a protocol
    - efficiency (speed, scalability) $\rightarrow$ robustness = scalable + environment constraints (attacks, failures)
  - Security: denial of service, spam, fraud...
  - Business: new users habits, new online services
  - Network management: multiple heterogeneous devices and applications spread around the world

Contributions

- Robustness $\longrightarrow$ modeling of communication applications (applied to the botnets)
  - network management
  - security
  - legal limitations $\longrightarrow$ modeling
- Reverse-engineering
  - security: protocol robustness, unknown protocol fingerprinting
  - protocol reusing
- Identification / fingerprinting
  - network management (automatic inventory)
  - security: track abnormal devices (attackers), potential victims (0 day attacks)
  - business: personalized advertisement or service offers

Outline

SIP

- SIP = Session Initiation Protocol[1]
- Multimedia session management (opening, parameters negotiation, closing)
- Widely used for Voice over IP (VoIP)
- Many potential attacks: denial of service, SpIT, toll fraud...

SIP = protocol example $\longrightarrow$ generic methods

[1] Rosenberg et al., RFC 3261

Outline

Objective

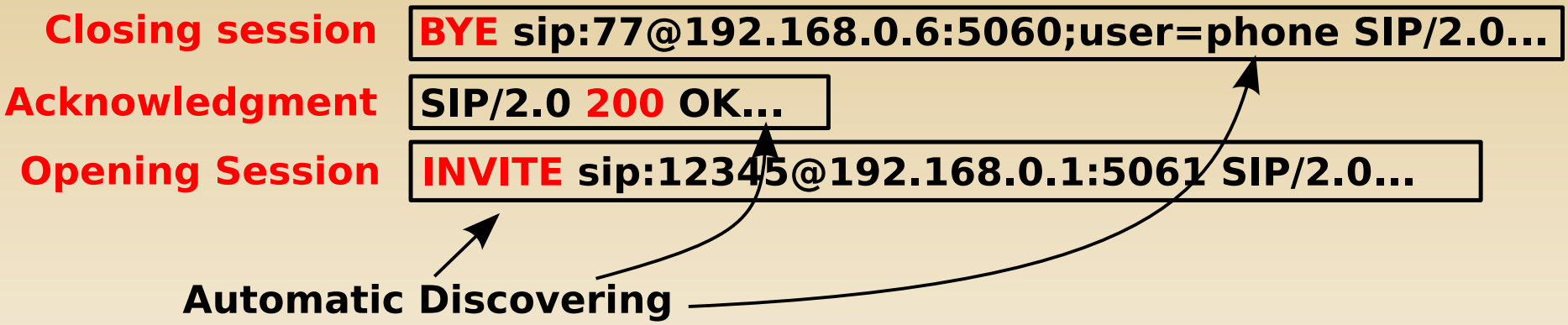
Reverse engineering = message types discovery + automatic construction of the protocol state machine → first contribution: the semantic of the messages

No strong knowledge about the syntax:

- grammar
- delimiters

Objective

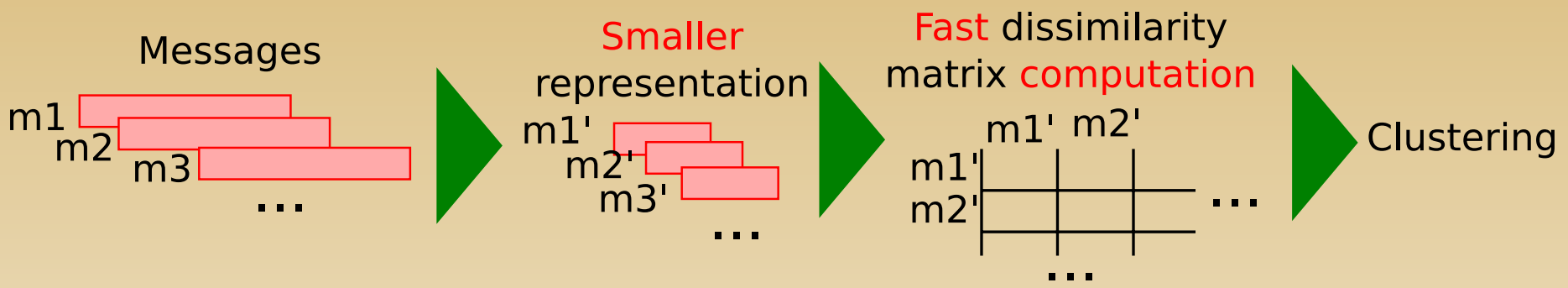
Reverse engineering = message types discovery + automatic construction of the protocol state machine → first contribution: the semantic of the messages



No strong knowledge about the syntax:

- grammar
- delimiters

Approach



Advantages / other techniques :

- no strong grammar knowledge / specific delimiters
- no tainting analysis (useful to retrieve a fine grained semantic)
- no manual analysis
- unsupervised (no learning samples)

$\sim$ sequence alignment techniques (high complexity)

Message representation

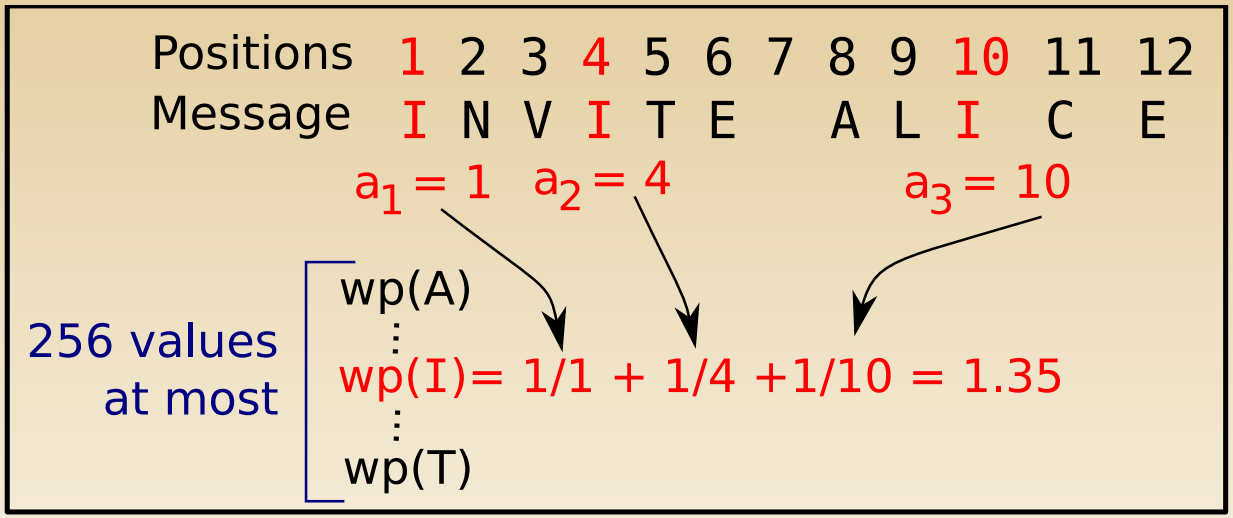
BYE sip:77@192.168.0.6:5060;user=phone SIP/2.0...

- Distinguish messages of different types
  - message length → too restrictive
  - characters distribution → many "garbage" characters without relationship with the type
  - protocol design → type at the beginning of the message → weighted average character positions
- Distribution comparison → Kullback-Leibler divergence (entropy measure), Gaussian kernel

Weighted positions

$$\forall \textit{ character } c,\ wp(c) = \frac{\sum_{i=1}^{k} pos(a_i)^{-1}}{k}$$

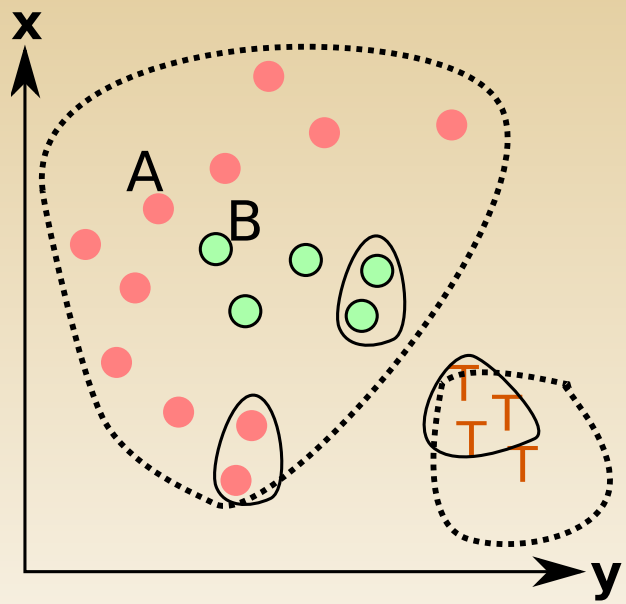
- Example



- Complexity to compare:
  - worst case = 256 values $\longrightarrow O(1)$
  - sequence alignment $\longrightarrow O(nm)$ where n and m are the length of messages to compare

Ascending hierarchical clustering

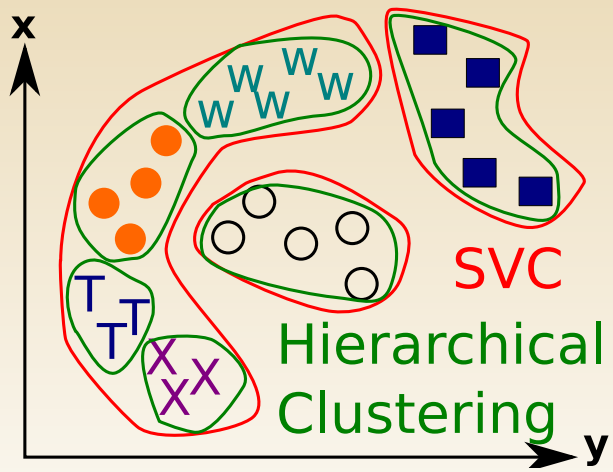
- Aggregative method
- Initialization: 1 message = 1 cluster
- Algorithm: merge the pair of closest clusters until the corresponding distance is higher than τ
- Advantages: simple, only one parameter: τ
- Drawback: bad efficiency with intertwined clusters



SVC

SVC = Support Vector Clustering[2]

- recent technique
- support vector machines → high accuracy + limited complexity in many domains (samples subset selection)
- advantages: irregular cluster shapes discovery
- Tests → global method with two passes: SVC + hierarchical clustering



[2] A. Ben-Hur, D. Horn, H.T. Siegelmann and V. Vapnik, A support vector clustering method, 2000

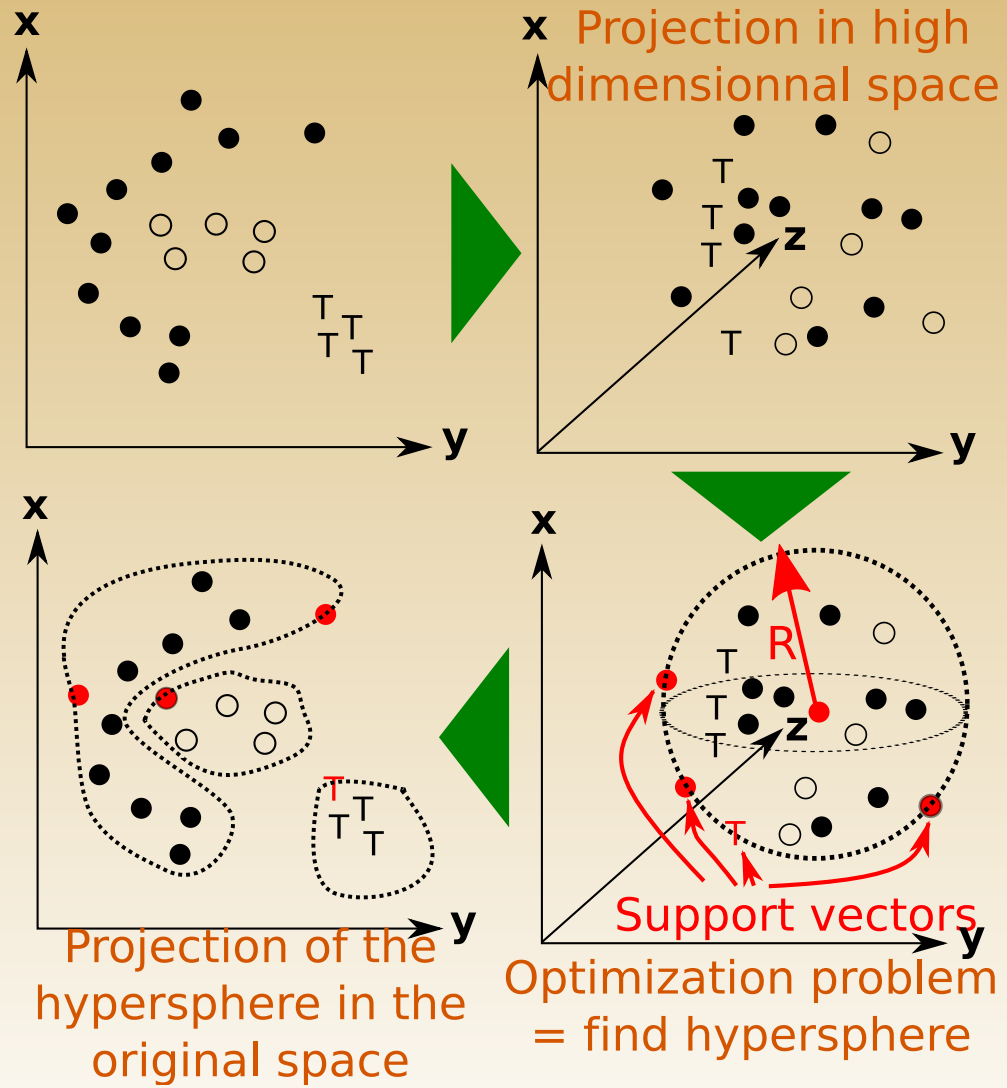
SVC

- Projection function $\Phi \longrightarrow \langle \Phi(m_i).\Phi(m_j) \rangle \Leftrightarrow$ kernel function
- Gaussian kernel:

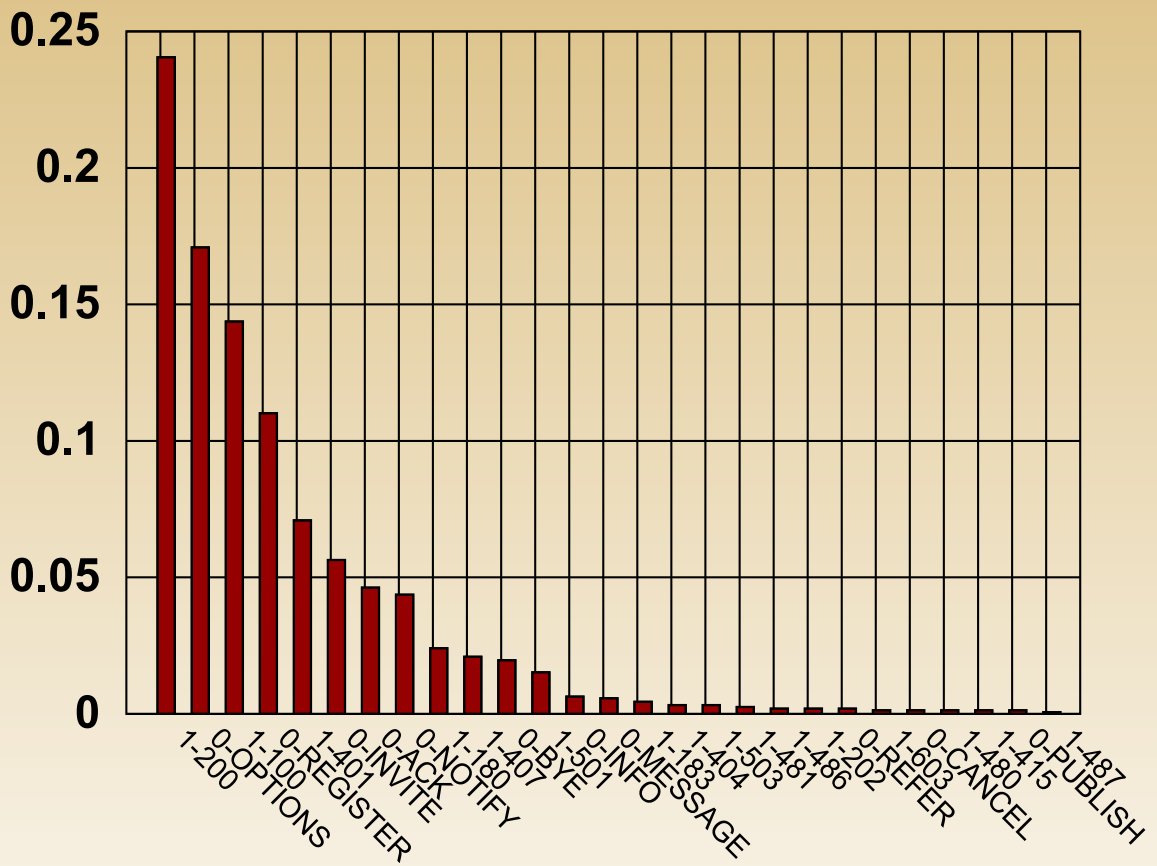
Parameter

$$K(m_i, m_j) = e^{-q\|m_i - m_j\|^2}$$

- One extra parameter (C) to avoid errors due to mislocated points (outside the hypersphere)



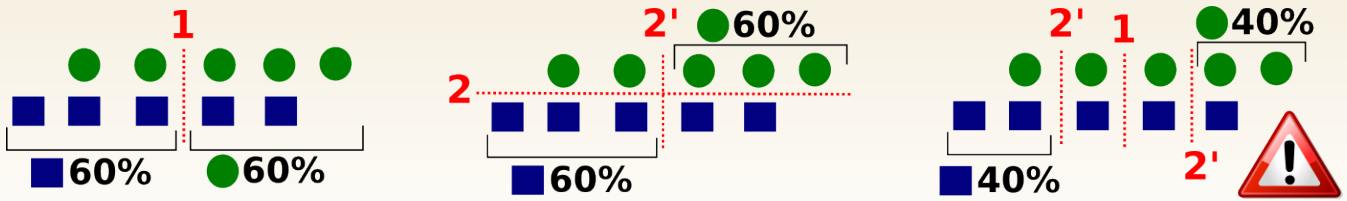
SIP dataset



- 1580 messages
- 27 types
- no uniform distribution

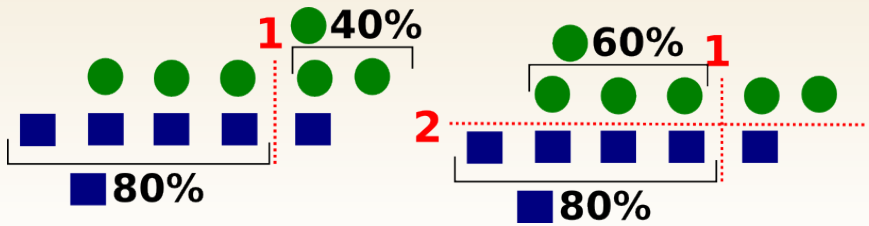
Results

- Cluster type = the most represented types within the contained messages + 1 cluster / type
- Metrics
  - accuracy = messages assigned to their real types
  - sensibility = messages of a specific real type assigned to this type
  - proportion of discovered types
- Results
  - hierarchical: accuracy = 0.85, all types are discovered
    - low standard deviation of the sensibility → all types are discovered with a similar accuracy > 50% → no need to apply a second pass
  - SVC: accuracy = 0.73, all types are discovered

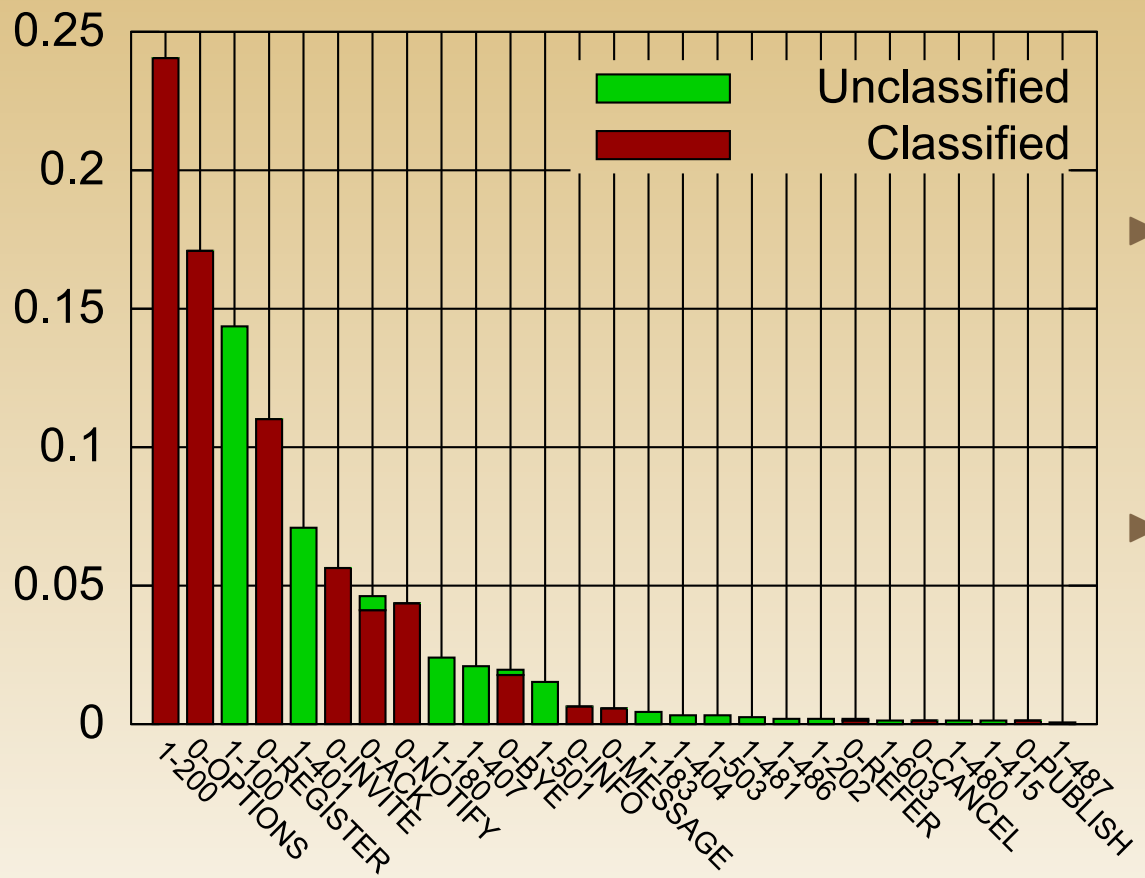


Results

- Cluster type = the most represented types within the contained messages + 1 cluster / type
- Metrics
  - accuracy = messages assigned to their real types
  - sensibility = messages of a specific real type assigned to this type
  - proportion of discovered types
- Results
  - hierarchical: accuracy = 0.85, all types are discovered
    - low standard deviation of the sensibility → all types are discovered with a similar accuracy > 50% → no need to apply a second pass
  - SVC: accuracy = 0.73, all types are discovered



Global method application



- Messages from undiscovered types within other clusters
- Meta-clusters including several types $\longrightarrow$ two-pass classification

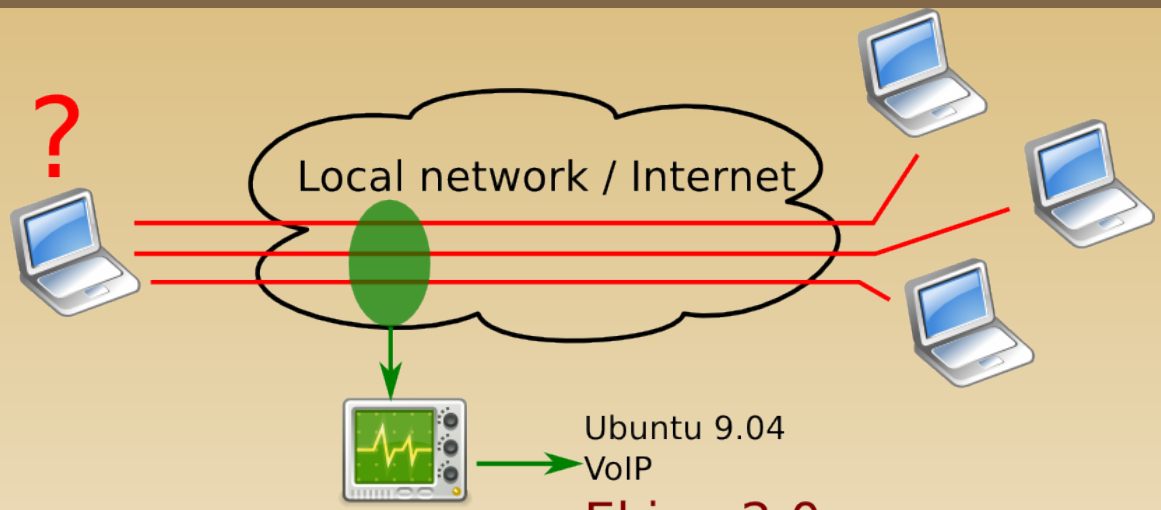
Global method (SVC + hierarchical clustering): accuracy = 91%, 96% of discovered types

Summary

- automatic message type discovery
- multiple experiments with or without SVC $\longrightarrow$ two pass classification (SVC + hierarchical)
- multiple experiments with different message characterization $\longrightarrow$ weighted character positions
- advantage: a very small representation of messages (low complexity)
- drawback: main assumption = type in first bytes of the message
- message type: state machine reconstruction, fingerprinting

Outline

Objective



- Passive approach
- Device fingerprinting = protocol stack = name + version (hardware / software)
- other techniques:
  - active / passive
  - OS / protocol (traffic) / device fingerprinting
  - device fingerprinting: active, specific protocol, "simple" signatures (specific field values)

Outline

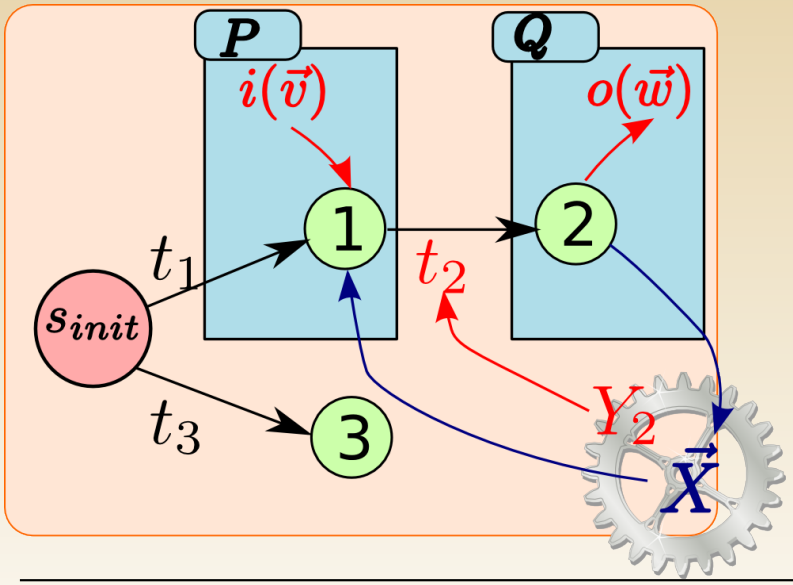
Key idea

- Use case:
  - unknown protocol
  - motivation: unavailable protocol specifications, fast $\rightarrow$ useful for general/preliminary studies
- Device behavior :
  - = interactions with other devices
  - user dependent but also device type dependent (functionalities, message types used)
- Methodology:
  - behavior = types of exchanged messages $\rightarrow$ reverse-engineering
  - available ressources can vary regarding the device $\rightarrow$ temporal aspect

A new formalization

TR-FSM : Random Tree Parameterized Extended Finite State Machine

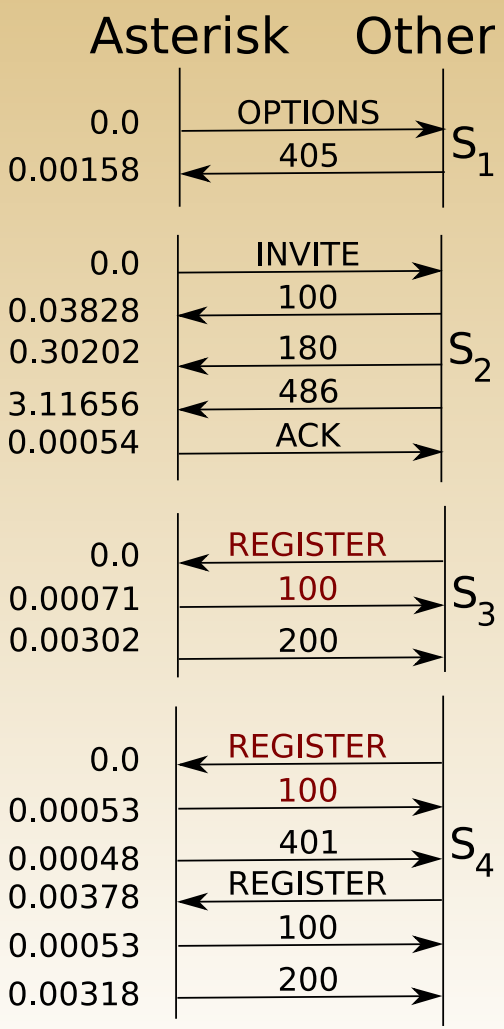
- Parameterized Extended Finite State Machine [3]

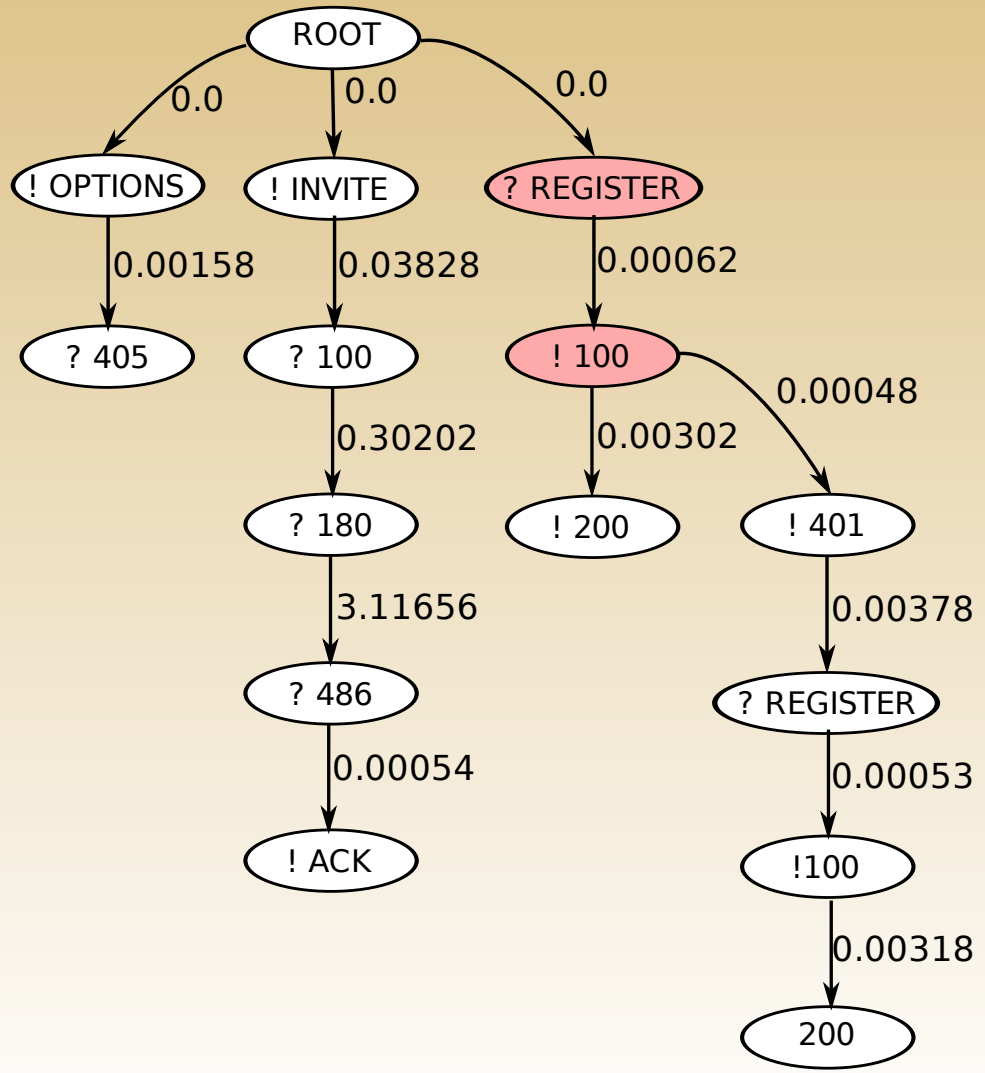


- extension:
  - one additional constraint = tree (interaction from the begin to the end)
  - random temporal variable added (transition time)

[3] G. Shu and D. Lee, *Network Protocol System Fingerprinting - A Formal Approach*, INFOCOM 06

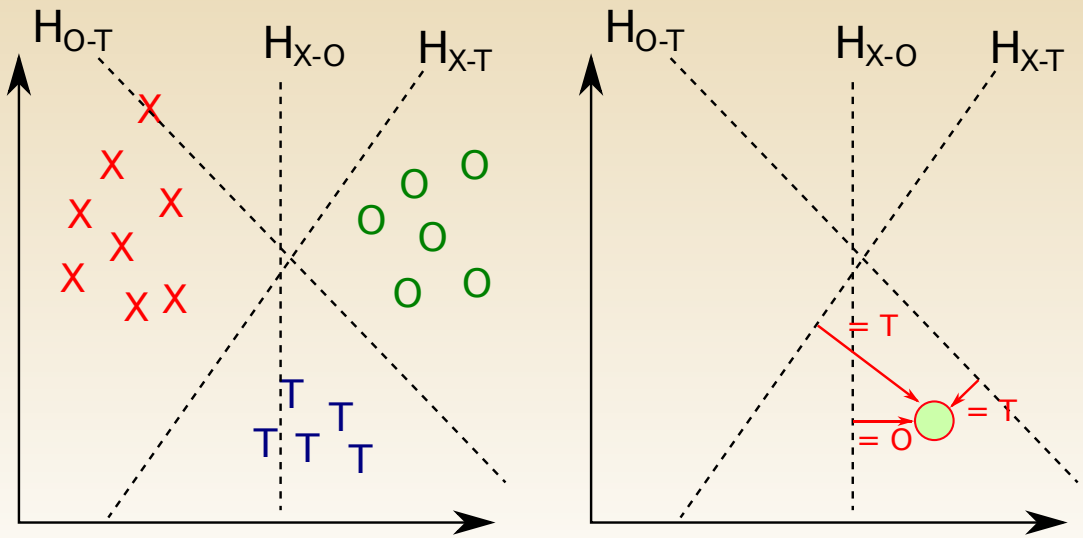
Example


Asterisk
Other
0.0
OPTIONS
405
0.00158
S1
0.0
INVITE
100
0.03828
180
0.30202
486
3.11656
ACK
0.00054
S2
0.0
REGISTER
100
0.00071
200
0.00302
S3
0.0
REGISTER
100
0.00053
401
0.00048
REGISTER
0.00378
100
0.00053
200
0.00318
S4


ROOT
0.0
0.0
0.0
! OPTIONS
! INVITE
? REGISTER
0.00158
0.03828
0.00062
? 405
? 100
! 100
0.00048
0.30202
0.00302
? 180
! 200
! 401
3.11656
0.00378
? 486
? REGISTER
0.00054
0.00053
! ACK
!100
0.00318
200

Identification

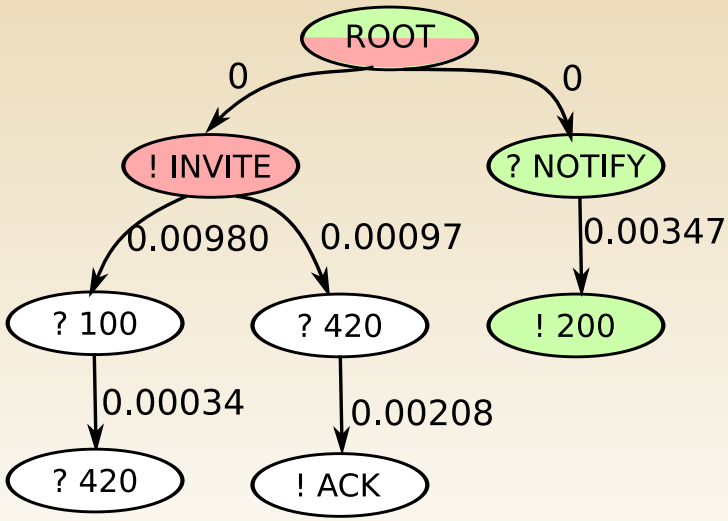
- Support vector machines (SVM)
  - points projected into a higher dimensional space $\longrightarrow$ possible linear separators
  - advantage: cluster irregular shapes
  - drawback: kernel function dedicated to TR-FSMs (*kernel trick*)



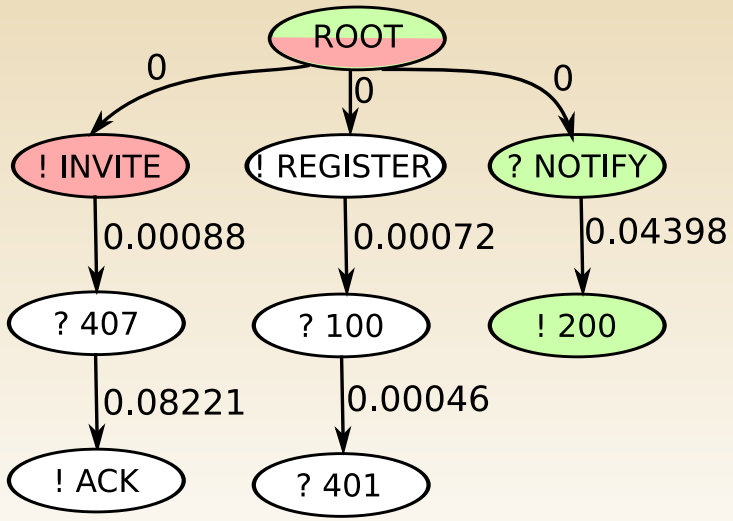
Kernel function

$$K(t_i, t_j) = \sum_{p \in I_{ij}} \sum_{n \in p} e^{-\alpha |f_{delay}(n,p_1) - f_{delay}(n,p_2)|}$$

- I_{ij} = shared paths (sequences of messages from the root without considering the delays) between t_i et t_j
- $f_{delay}(n, p)$ = delay of the node n in p



Twinkle 1.10



Cisco 7940 firmware 8.93

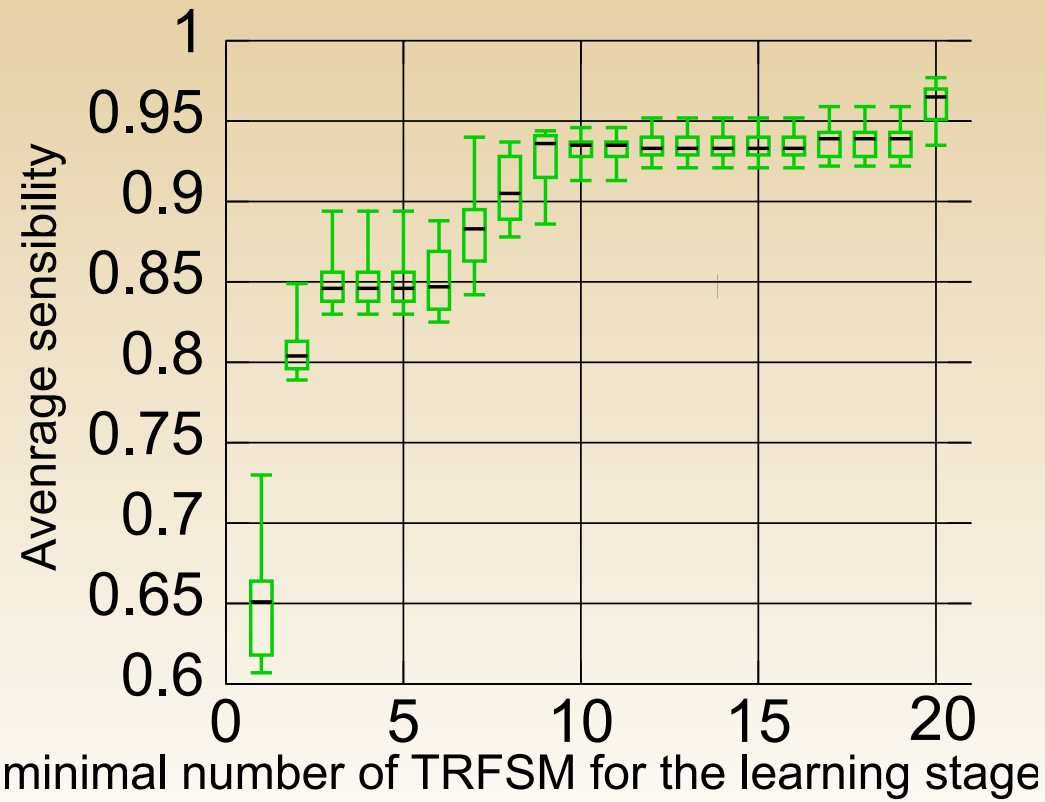
Evaluation (SVM)

- Two datasets: 1 testbed + VoIP operator (Internet → greater noise)
- Specificity = consistency of a cluster

	Test	Op.	
#device types #messages #sessions	26 18066 2686	42 95908 29775	op. more complete
% learning	40	10	
Average(#messages/session) Average(delay) (sec) Average cardinality	6.73 1.53 18.97	3.22 6.76 12.94	op. sessions → less messages, longer time
Accuracy Average specificity	0.91 0.91	0.86 0.81	good results
Average sensibility	0.64	0.58	types badly identified

Evaluation (SVM)

- Poor sensibility
  - devices with few sessions $\longrightarrow$ few learning trees
  - force a minimal number of trees for the learning



- 80 % $\longrightarrow$ at least 2 TR-FSM
- 2×5 sessions $= 10$ sessions are needed for the learning stage for each device type
- testing (10 sessions) $\longrightarrow$ one session based identification is impossible

Summary

- behavior $\longrightarrow$ sequences of message types + delays
- SVM classification: accuracy close to 85-90%
- message type = reverse-engineering
- advantages
  - no strong knowledge about the protocol
  - TR-FSM formalization + suited kernel
- drawback: need to wait several interactions before identification
- syntax knowledge $\longrightarrow$ message type + much information $\longrightarrow$ syntactic fingerprinting

Outline

Motivation

Key idea = each device sends specific information

- message content = discriminative information (User-agent) but easily alterable
- content organization (hierarchy, order) = discriminative also (depends on how the device constructs the message) + more difficult to alter without being meaningless

Syntactic tree construction

Protocol specified by an ABNF[4] grammar

- successive rules derivation
- example: “INVITE Accept: */*.Call-id:456ZE852.”

Message = Request SP *Header SP 0*1Body

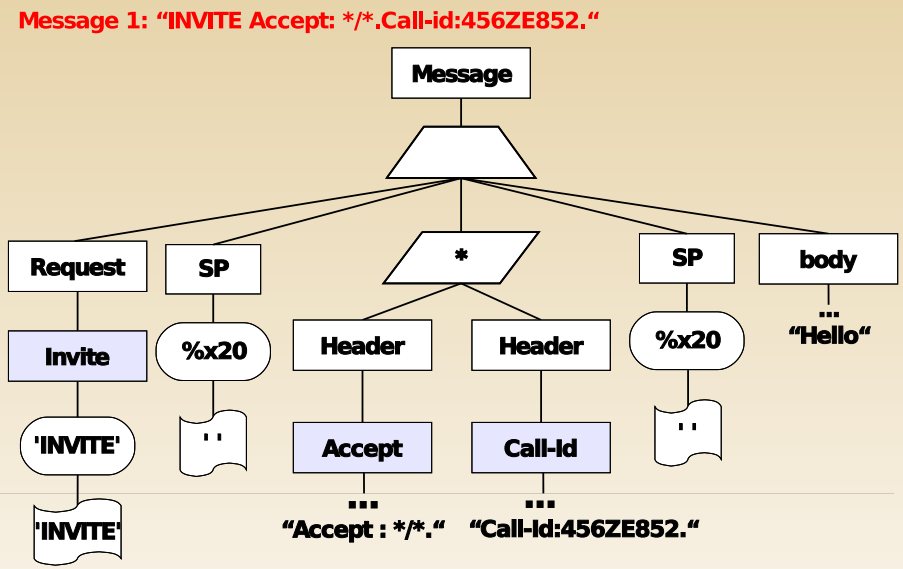
Request = Invite / Notify / Cancel

Invite = “INVITE”

Header = Accept / Date / Call-id / User-Agent

Accept = "Accept" HCOLON *Alpha "."

Alpha = %x41-5A / %x61-7A ; A-Z / a-z
HCOLON = *SP ":" *SP
SP = %x20 ; espace



[4] Augmented Backus-Naur Form

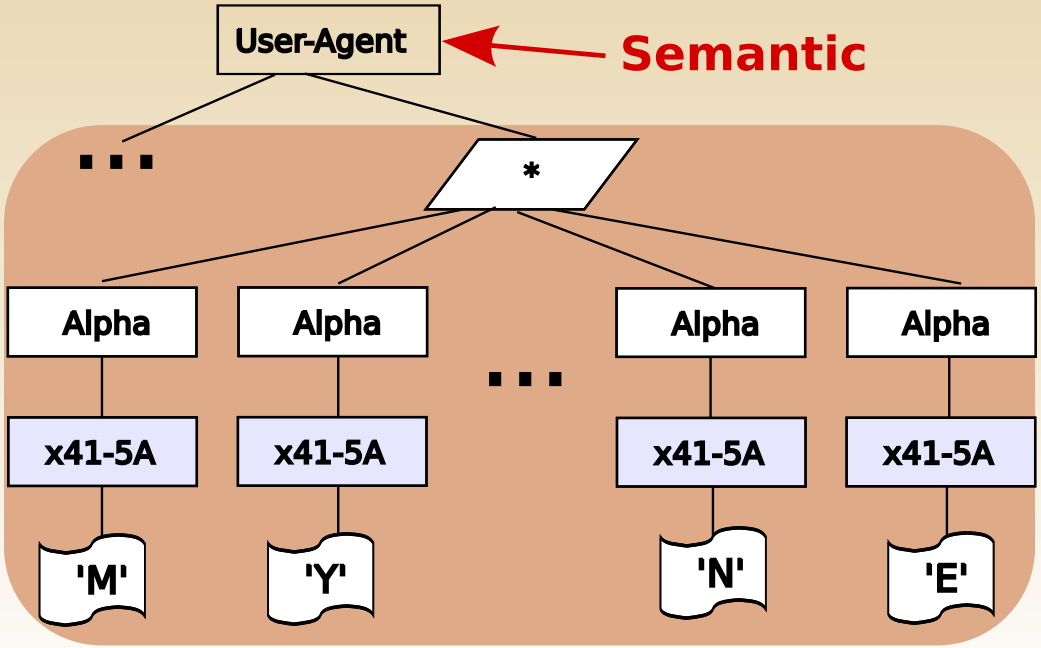
Comparison

- Syntactic tree = signature $\longrightarrow$ similarity between 2 signatures
- Tree $\neq$ value $\longrightarrow$ "usual" distances not adapted
- Edition distance (number of transitions) : NP-complete
- Polynomial complexity distance[5]
  - isomorphism ϕ between two trees $\longrightarrow$ subtrees with the same number of nodes interconnected in the same manner
  - similarity : $W(\phi) = \sum_{u \in H_1} \sigma(u, \phi(u))$
  - necessary to define the similarity metric between two nodes (σ)

[5] A. Torsello et al., *Polynomial-time metrics for attributed trees*, TPMAI, vol. 27, no. 7, 2005

Similarity

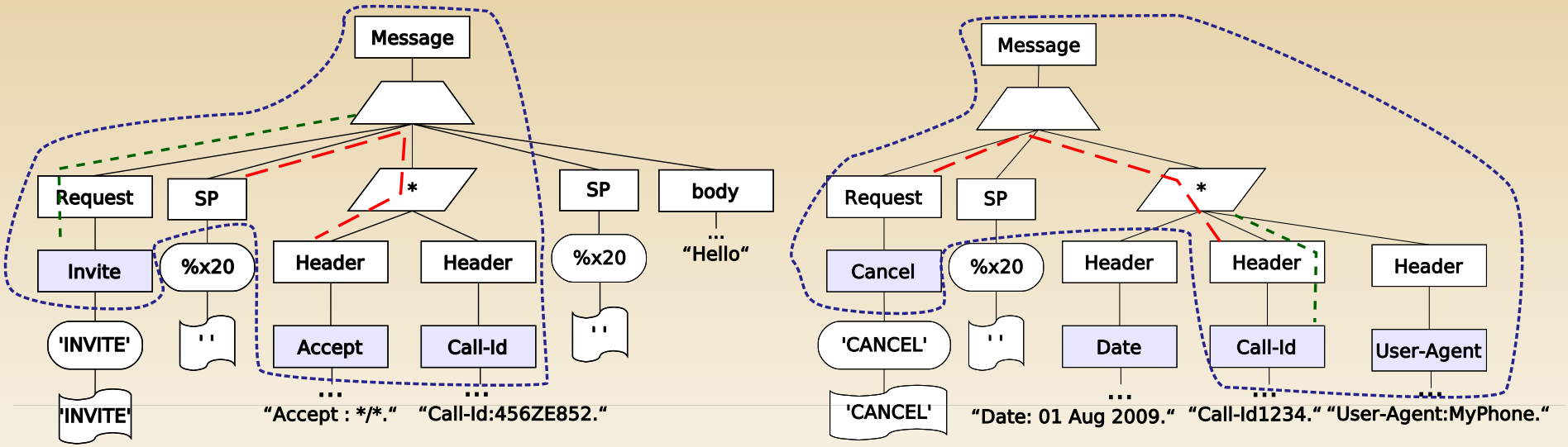
- $\sigma(u, v) = 1$ if equivalent node
  - sequence $\sim$ repetition
  - non terminal with the same name
  - semantic terminal depending on context are not taken in account (username, session id...)



- large generic structures problem

Similarity

- equivalent nodes $\longrightarrow$ equivalent ancestors
- $W(\phi_1) = 3$, $W(\phi_2) = 1$, $W(\phi_3) = 8$



Similarity

- shared ancestors $\longrightarrow$ isomorphism between subtrees rooted in the original root
  - $W(\phi_{12})$ = maximal similarity between t_1 et t_2 = cardinality of the intersection of the tree paths
  - complexity = $O(|t_1||t_2|)$
- distances
  - non-normalized :

$$d1(t_1, t_2) = |t_1| + |t_2| - 2W(\phi_{12})$$

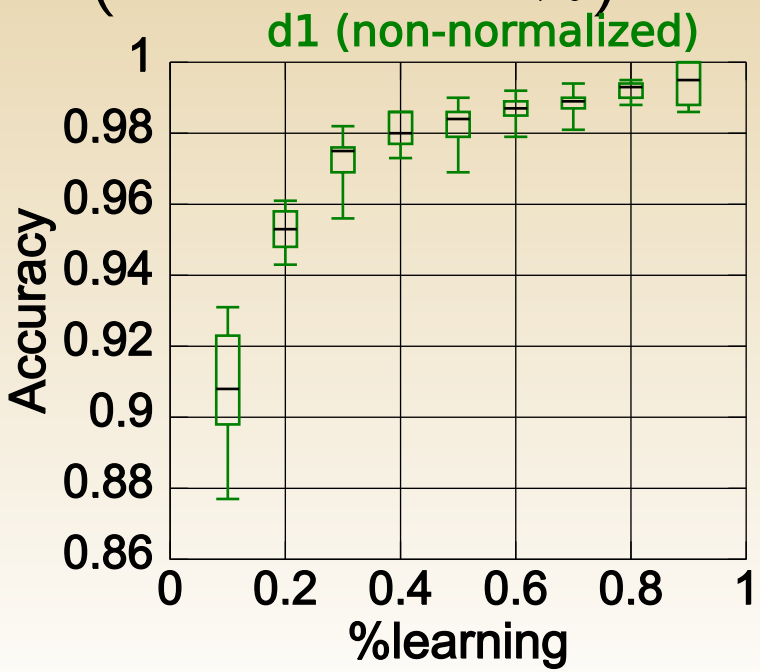
  - normalized :

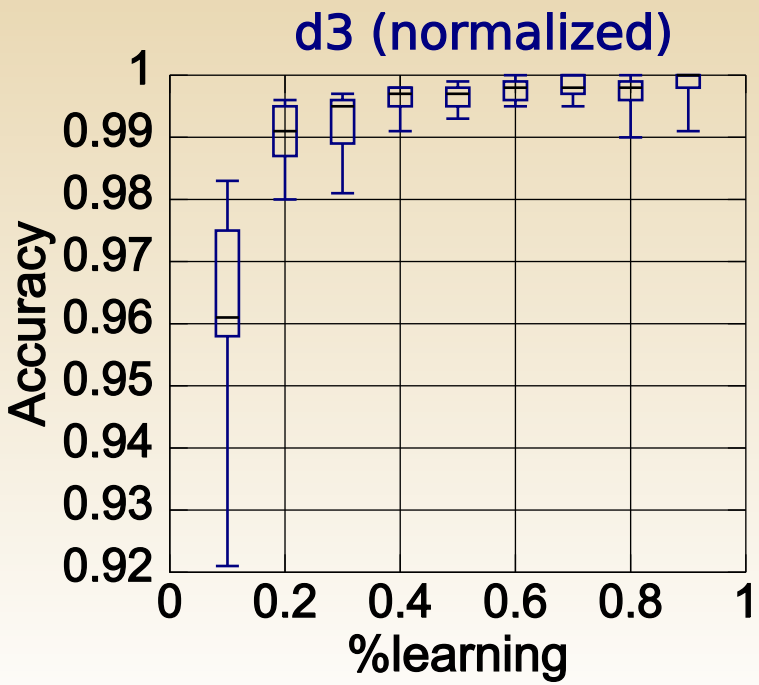
$$d2(t_1, t_2) = 1 - \frac{W(\phi_{12})}{max(|t_1|, |t_2|)}$$

$$d3(t_1, t_2) = 1 - \frac{W(\phi_{12})}{|t_1| + |t_2| - W(\phi_{12})}$$

Supervised fingerprinting

- SVM application by adapting the distances as kernel functions
- only one message to identify
- syntax $>$ message types $\longrightarrow$ better performance (TR-FSM: 90%)

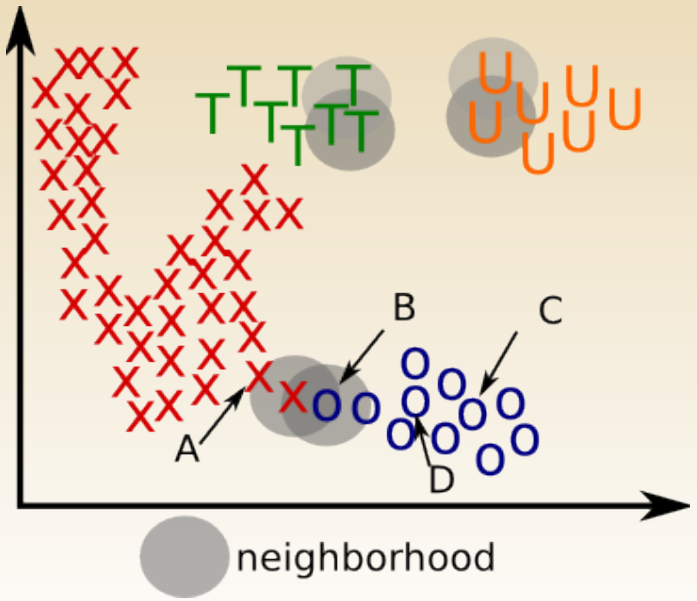




Unsupervised fingerprinting

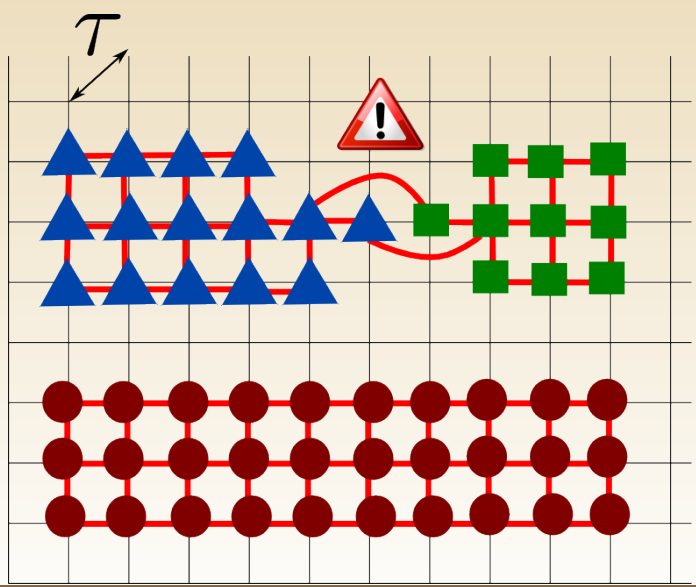
- SVC classification:
  - kernel function
  - clusters assignation → creation of intermediary trees (complexity increase)

- ROCK (S. Gua *et al.*, ICDE 99)
  - trees → categorical data
  - density of data points can help to discover irregular cluster shapes
  - initialization: 1 point = 1 cluster
  - merge clusters with shared neighbors



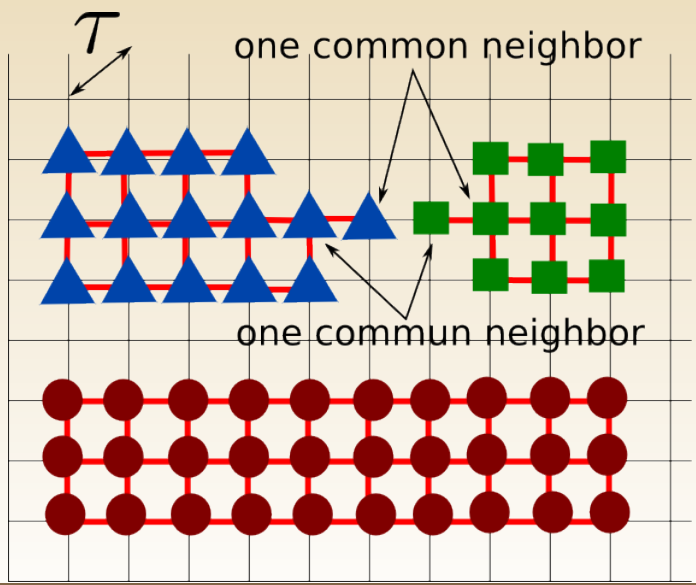
Unsupervised fingerprinting

- ROCK $\longrightarrow$ score metric to determine if two clusters are close regarding their shared neighbors:
  - number of shared neighbors + estimation of the maximal number
  - two neighbors $\leftrightarrow$ maximal inter-distance τ
  - needs to be recomputed for each modification
- QROCK (Dutta *et al.*, 2005)
  - graph representation, 1 edge = at least one shared neighbor $\longrightarrow$ very fast (x10 ROCK)
  - connected components $\longrightarrow$ too simple



Unsupervised fingerprinting

- ROCK $\longrightarrow$ score metric to determine if two clusters are close regarding their shared neighbors:
  - number of shared neighbors + estimation of the maximal number
  - two neighbors $\leftrightarrow$ maximal inter-distance τ
  - needs to be recomputed for each modification
- QROCK (Dutta *et al.*, 2005)
  - graph representation, 1 edge = at least one shared neighbor $\longrightarrow$ very fast (x10 ROCK)
  - connected components $\longrightarrow$ too simple
  - extension: 1 edge = at least γ (2) shared neighbors

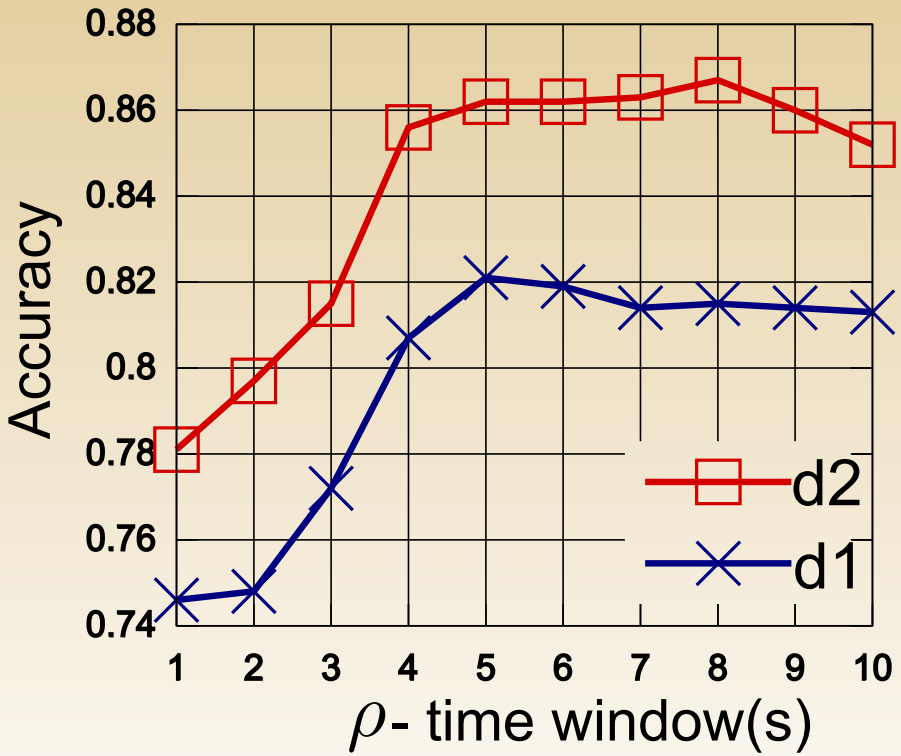


Unsupervised fingerprinting

- Direct application:
  - accuracy = 60%
  - the messages from the same device can be highly different regarding the context $\longrightarrow$ the messages of a single device are scattered among several clusters
- Solutions :
  - a priori creation of small clusters = messages sharing the same source IP address and ports within a small temporal windows (ρ seconds)
  - classify messages for a specific context $\sim$ message types

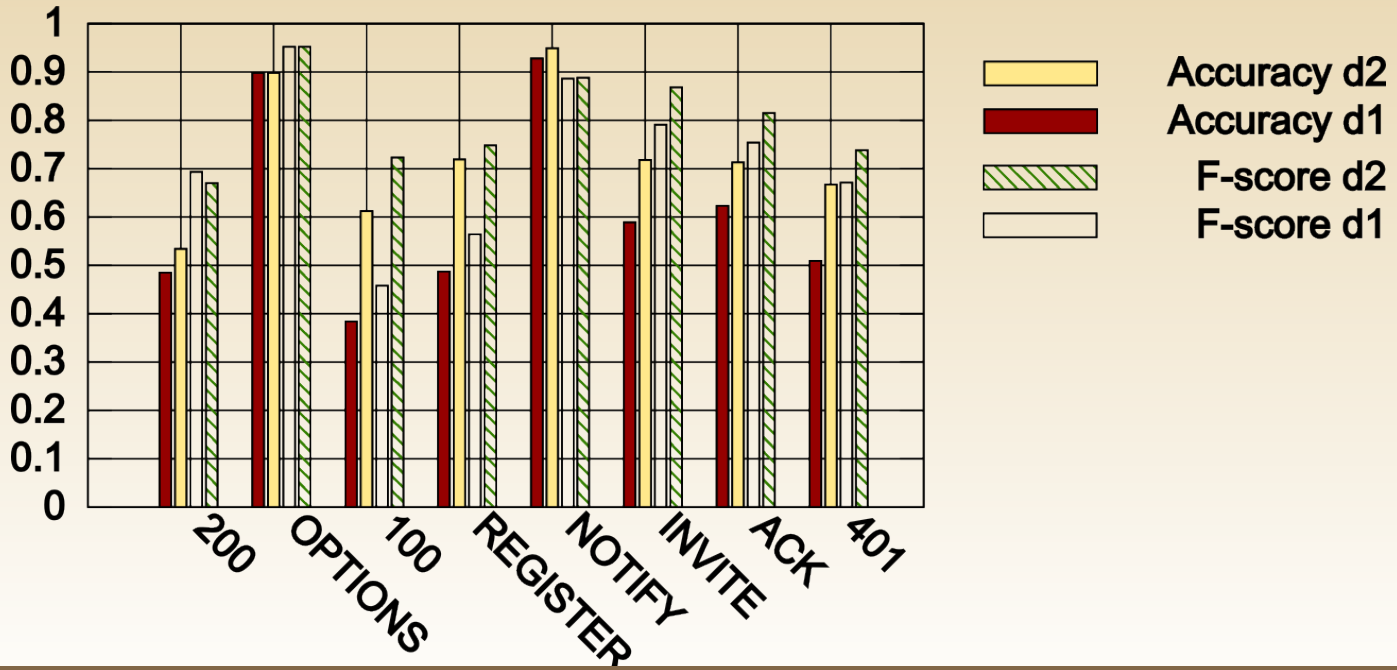
Unsupervised fingerprinting

- Temporal merging:
- $\tau = 0.1$, $\gamma = 15$
- $\rho = 5s.$ $\rightarrow$ micro-clusters with 2,8 messages
- QROCK:
  - $\sim \gamma = 1$
  - best case: accuracy = 76%
  - tradeoff ROCK/QROCK >QROCK with an equivalent complexity



Unsupervised fingerprinting

- Context dependant classification:
  - normalized distance $>$ non normalized distance
  - $\exists$ highly discriminative types : OPTIONS (90%)= device features
  - $\exists$ lowly discriminative types : 200 ($<$ 50%)= acknowledgement, response codes



Summary

- syntax $\longrightarrow$ hierarchical structure of message
- advantages:
  - syntax structure is a good discriminative feature (compared to message types)
  - tree comparison with a limited complexity
- drawbacks:
  - complete syntax knowledge
  - parsing / tree construction is time consuming (compared to message type discovery)
- unsupervised fingerprinting (ROCK/QROCK): number of distinct device types, device distribution / type
- supervised fingerprinting (SVM): training samples, device name identification

Outline

Summary

- Contributions to several fields: security, network management...
- Protocol reverse-engineering :
  - passive
  - network traces
  - new discriminative representation $\longrightarrow$ low complexity
- fingerprinting :
  - passive
  - behavioral: unknown protocol (reverse engineering)
  - syntactic: faster, better performance

Other contributions

- Botnets robustness :
  - parameterized models
  - different architectures (pseudo-centralized, P2P)
  - definition of new metric for the robustness study
  - help to choose the more suiteable architecture regarding the context (number of machines, security, anonymization, delays)
- work correlated to classification techniques
  - collaboration with Y. Guermeur, ABC team, CNRS / LORIA
  - application of recent and highly performant techniques (support vectors machines)
  - deep analysis of parameters impact $\longrightarrow$ help for a real use
  - generic contribution to the classification domain : SVC (automatic parameter determination) + tradeoff ROCK/QROCK

Perspectives

- Real instantiation of botnets models + add other factors (human impact)
- Reverse-engineering + fingerprinting: validation with other protocols
- Online fingerprinting:
  - study about the complexity / computation time
  - method adaptation (simplification, sampling, parallel computing)
- New fingerprinting techniques:
  - combination of behavior and syntax
  - application profiling

Publications

- international conferences / workshops:
  - "Automated behavioral fingerprinting", *RAID*, 2009, 28,3%.
  - "Towards malware inspired management frameworks", *NOMS*, Avril 2008, 27,5%.
  - "Botnet based scalable network", *DSOM*, 2007, 31,3%.
  - "Malware Models for Network and Service Management", *AIMS* 07 24%
  - "A collaborative approach for proactive detection of distributed denial of service attacks", *MonAM*, 2007.
- french speaking journal:
  - "Les botnets et la supervision à large échelle", *TSI*, 2009.
- french speaking conference:
  - "Les botnets et la supervision à large échelle", *JDIR*, 2008.
- research reports (Submitted):
  - "Behavioral and Temporal Fingerprinting", 2009.
  - "Advanced Fingerprinting For Inventory Management", 2009.
  - "FireCol: a collaborative protection network for the early detection of flooding DDoS attacks", 2009.

Soutenance de thèse de l'Université Henri Poincaré Nancy 1

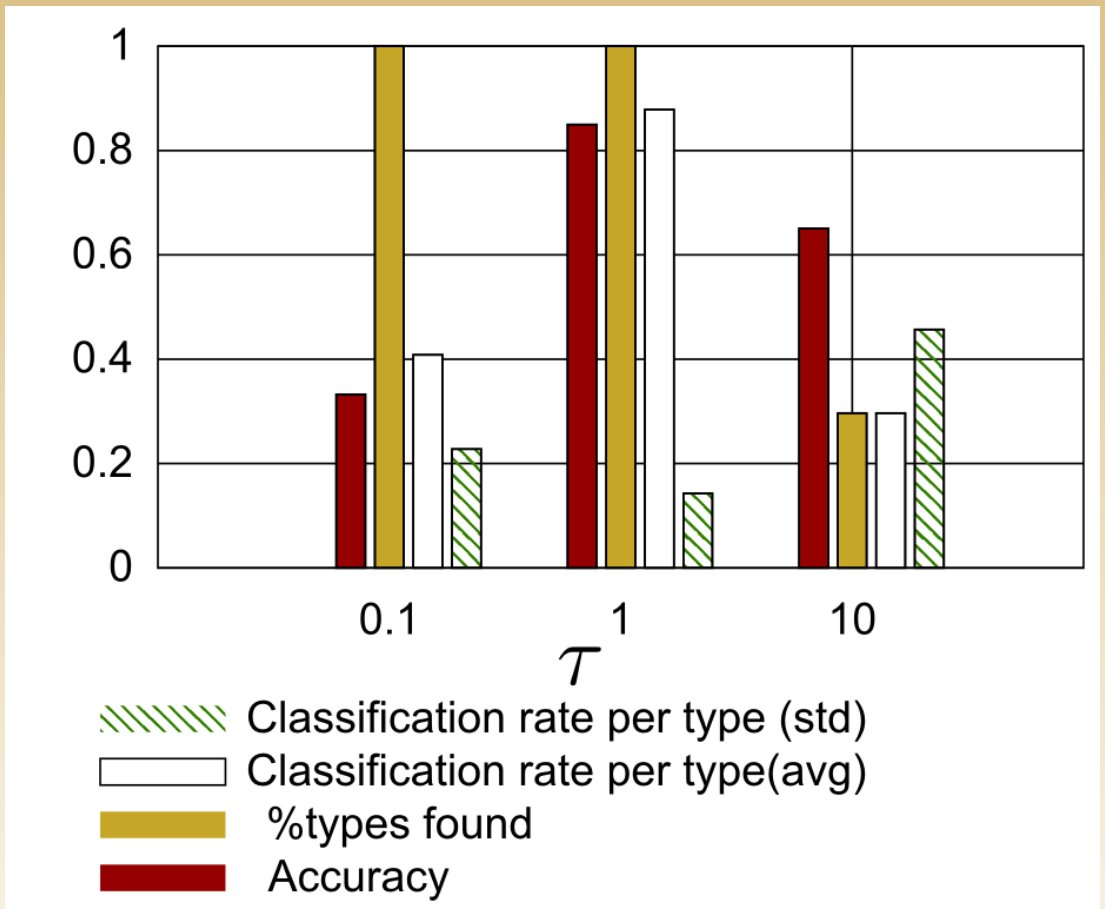
Robustesse et Identification des Applications Communicantes

–

Robustness and Identification of Communicating Applications

Jérôme François
sous la direction d'Olivier Festor et Radu State

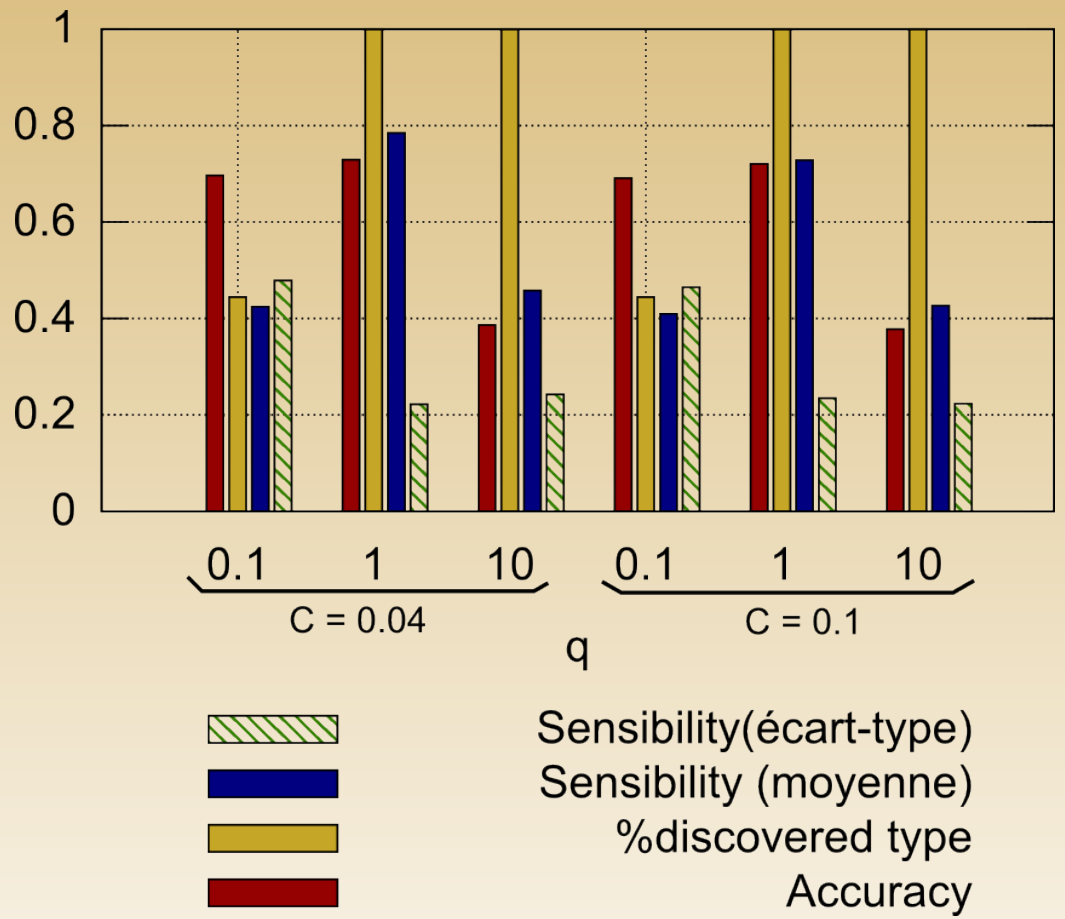
Nearest neighbors only results



- τ increases
  - larger clusters
  - begin: group similar messages
  - end: group different messages
- low standard deviation per type + manual analysis $\longrightarrow$ mixed types

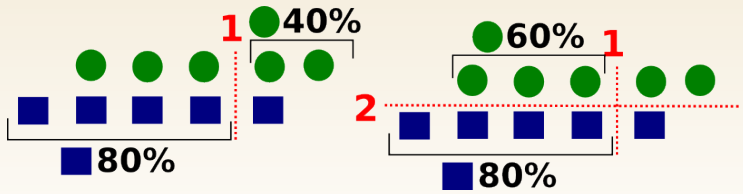
- best case: accuracy = 0.85, all message types found

Results



- best case: accuracy = 0.73, all types are discovered

- SVC application
- no impact of C
- q increases $\longrightarrow$ message differences emphasized $\longrightarrow$ more clusters
- q = 0.1: high standard deviation of the sensibility

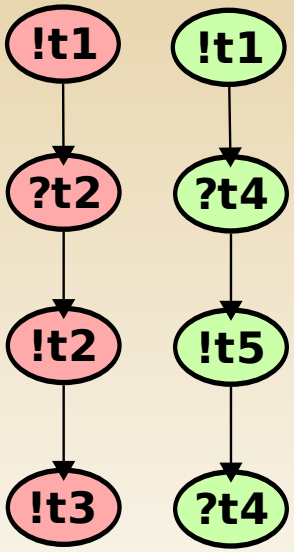


Construction

1 session = 1 séquence entre 2 entités (mêmes adresses IP, même ports)

t1(A→B), t2(B→A), t2(A→B), t3(A→B)

t1(A→B), t4(B→A), t5(A→B), t4(A→B)

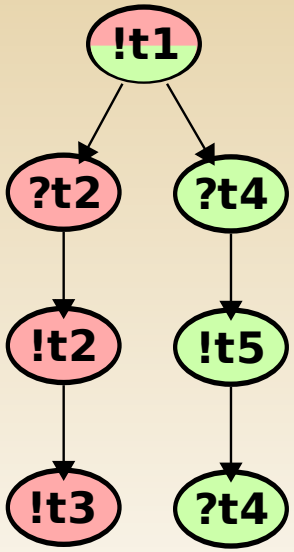


Construction

1 session = 1 séquence entre 2 entités (mêmes adresses IP, même ports)

t1(A→B), t2(B→A), t2(A→B), t3(A→B)

t1(A→B), t4(B→A), t5(A→B), t4(A→B)

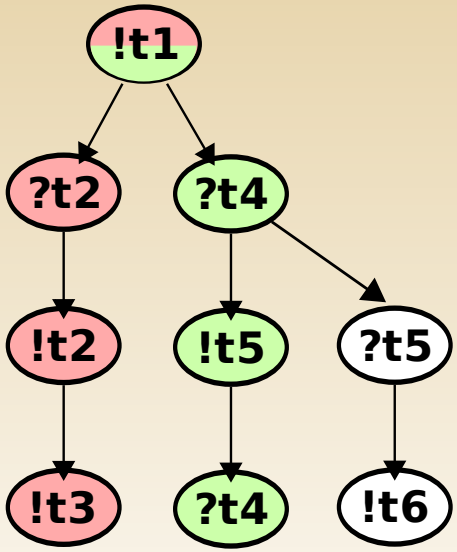


Construction

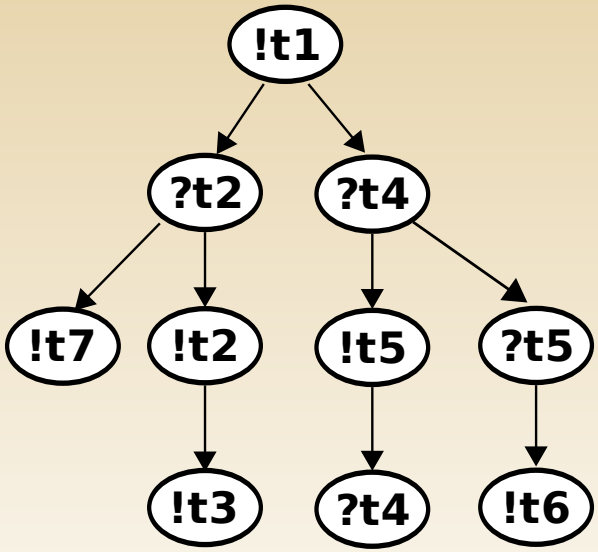
1 session = 1 séquence entre 2 entités (mêmes adresses IP, même ports)

t1(A→B), t2(B→A), t2(A→B), t3(A→B)

t1(A→B), t4(B→A), t5(A→B), t4(A→B)



Équipement A



Équipement B

Evaluation

- accuracy

#sessions / TR-FSM (learning)	#sessions / TR-FSM (testing)				
	1	5	10	20	40
1	0.682	0.819	0.830	0.805	0.745
5	0.469	0.858	0.905	0.883	0.800
10	0.376	0.809	0.894	0.873	0.819
20	0.272	0.656	0.821	0.864	0.837
40	0.221	0.469	0.627	0.764	0.762

- Best configuration :
  - testing = 10 sessions / TR-FSM $\rightarrow$ one session based identification is impossible
  - learning = 5 sessions / TR-FSM + 40% $\rightarrow$ at least 13 sessions minimum to build one signature