



DECEMBER 3 - 6, 2012
EMIRATES PALACE | UNITED ARAB EMIRATES

In partnership with:



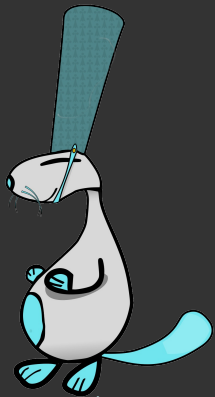
Supported by:



Reverse and Simulate your Enemy Botnet C&C

Blackhat Abu Dhabi
December 5 2012

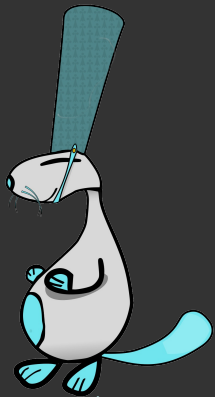
Please complete the Speaker
Feedback Surveys.



Authors...

« You talkin' to me? »

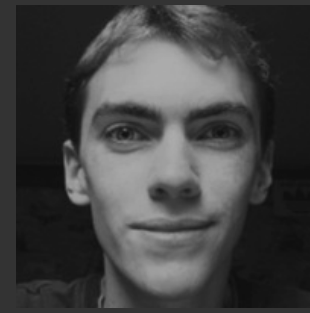
Travis Bickle





Frédéric GUIHERY

- IT security engineer
 - Reverse engineering
 - System analysis and hardening
 - Trusted Computing



Georges BOSSERT

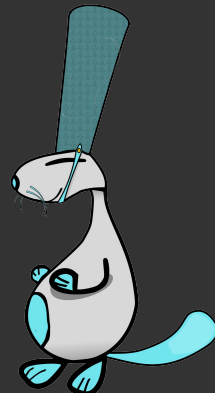
- PhD student
 - Intrusion Detection
 - Botnet simulation
 - Protocol learning

Supelec CIDre
Research team



Advisers :

- *Guillaume Hiet*
- *Ludovic Mé*

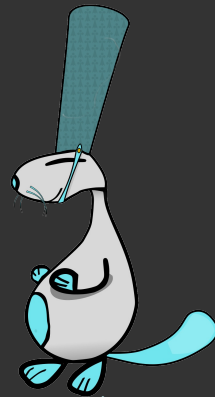


netzob.org



AMOSSYS, France

- Audit and evaluation
 - ITSEF lab (Common Criteria, CSPNs, ...)
 - Pentest lab
- R&D



netzob.org

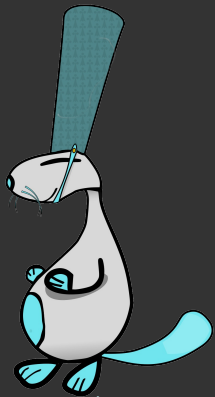


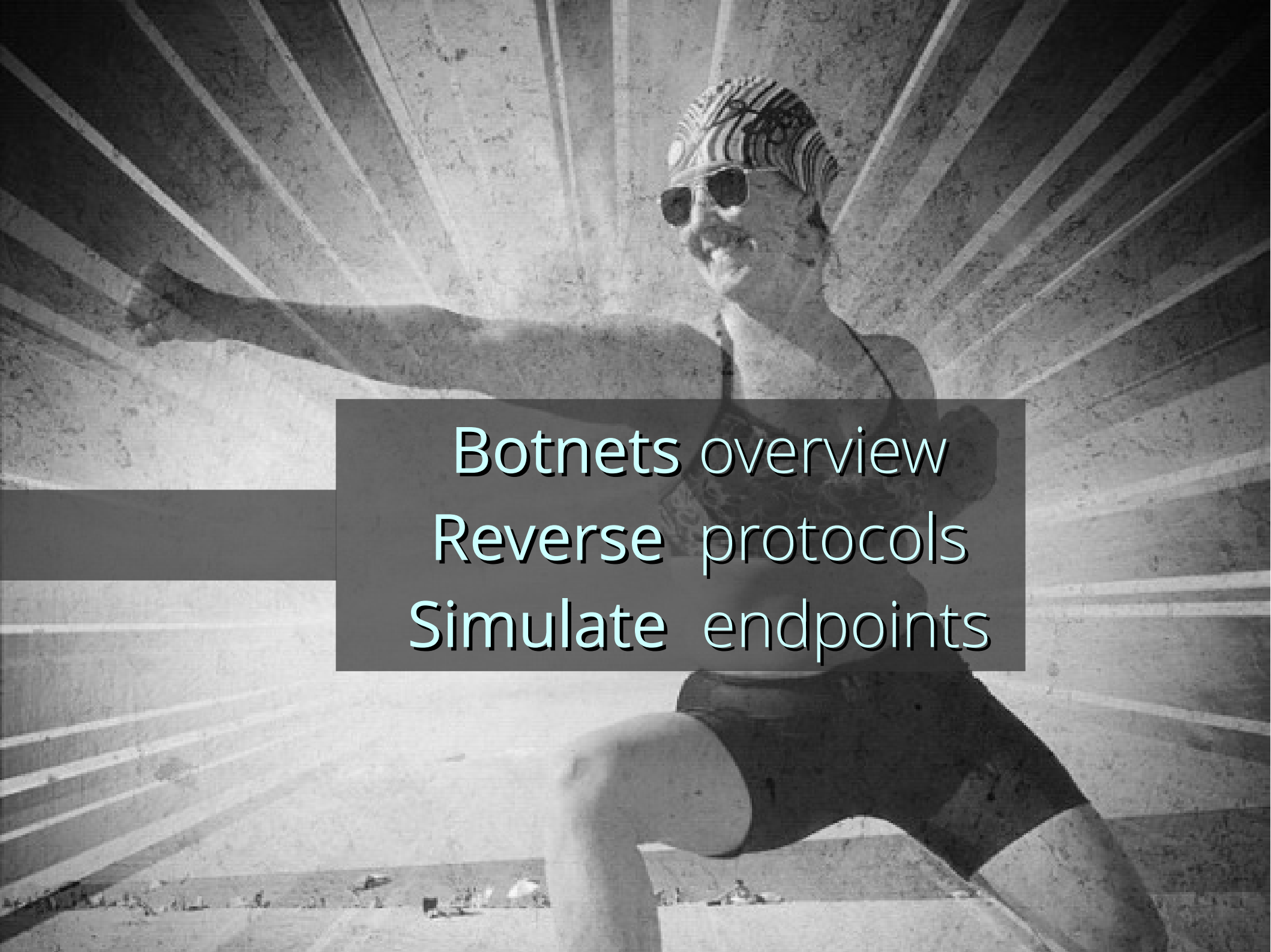
Yes, we're French !

Topics...

« Go ahead, make my day »

Harry Callahan





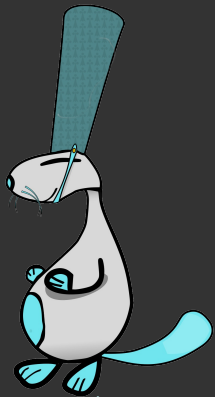
Botnets overview
Reverse protocols
Simulate endpoints



Why ?

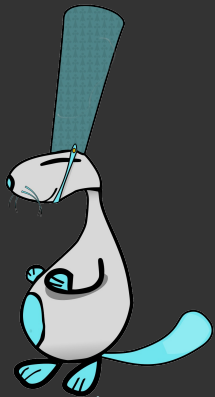
Botnets

- *2 different binaries (or part of)*
 - *the master*
 - *the zombies*
- *A set of actions to perform*
- *A Communication Channel*
 - *A network topology*
 - *A proprietary protocol*



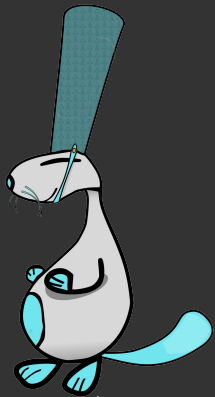
Botnets

- *2 different binaries (or part of)*
 - *the master*
 - *the zombies*
- *A set of actions to perform*
- *A Communication Channel*
 - *A network topology*
 - *A proprietary protocol*



Botnets

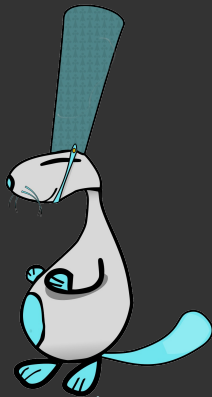
- *2 different binaries (or part of)*
 - *the master*
 - *the zombies*
- *A set of actions to perform*
- *A Communication Channel*
 - *A network topology*
 - *A proprietary protocol*



Botnets:

- *2 different binaries (or part of)*
 - *the master*
 - *the zombies*
- *A set of actions to perform*
- *A Communication Channel*
 - *A network topology*
 - *A proprietary protocol*

**Analysts'
point-of-view**

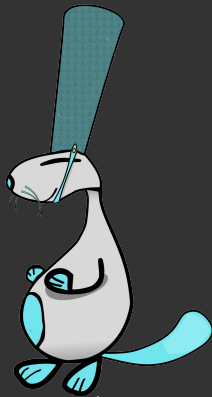


Botnets:

- *2 different binaries (or part of)*
 - *the master*
 - *the zombies*
- *A set of actions to perform*
- *A Communication Channel*
 - *A network topology*
 - *A proprietary protocol*

**Analysts'
point-of-view**

Binary RE
(static/dynamic)



Botnets:

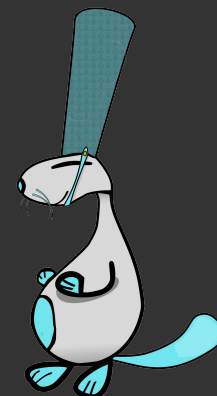
- *2 different binaries (or part of)*
 - *the master*
 - *the zombies*
- *A set of actions to perform*
- *A Communication Channel*
 - *A network topology*
 - *A proprietary protocol*

**Analysts'
point-of-view**

Binary RE

(static/dynamic)

Behavioural Analysis
(AV/HIDS)



Botnets:

- *2 different binaries (or part of)*
 - *the master*
 - *the zombies*
- *A set of actions to perform*
- *A Communication Channel*
 - *A network topology*
 - *A proprietary protocol*

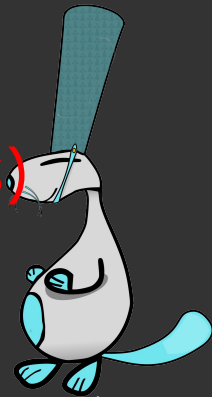
**Analysts'
point-of-view**

Binary RE

(static/dynamic)

Behavioural Analysis
(AV/HIDS)

Macro Analysis
(ISPs/consortiums)



Botnets:

- *2 different binaries (or part of)*
 - *the master*
 - *the zombies*
- *A set of actions to perform*
- *A Communication Channel*
 - *A network topology*
 - *A proprietary protocol*

**Analysts'
point-of-view**

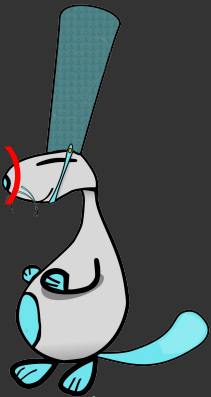
Binary RE

(static/dynamic)

Behavioural Analysis
(AV/HIDS)

Macro Analysis
(ISPs/consortiums)

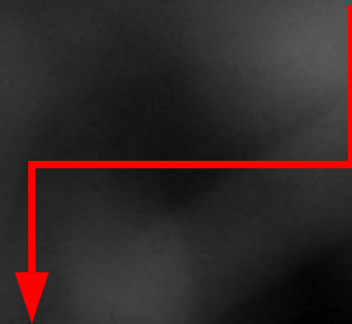
Protocol RE
(????????????????????)



netzob.org

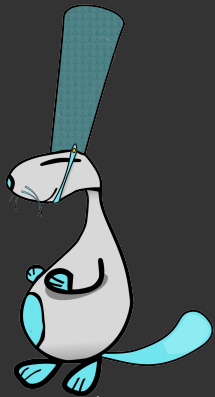
A black and white photograph of a door handle. The handle is oval-shaped and has a cutout in the center. Inside the cutout is a black silhouette of a question mark. The background is blurred, showing what appears to be a door frame and some foliage.

No available tool to reverse
a **proprietary** protocol...



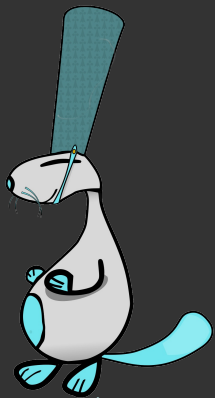
Should we create one?

Other « use-cases » for protocol RE



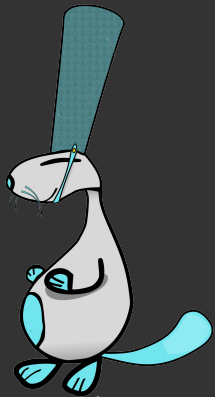
To assess the robustness of implementations

- *Ex : Fuzz the control API of a centrifuge*



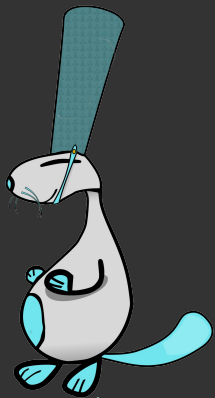
To analyze traffic and identify potential data leakage

- ▶ *Ex : Are you sure your « IP Reputation Appliance » doesn't leak your emails ?*



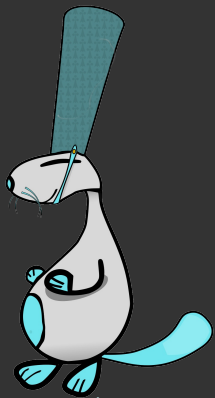
To compare the implementation of a protocol
with its official specifications

- *Ex : CC evaluations of crypto products*

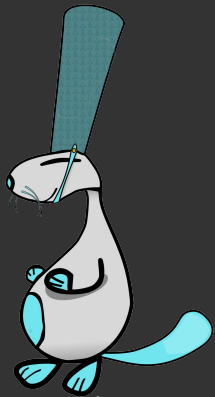


To develop a free version of a proprietary implementation

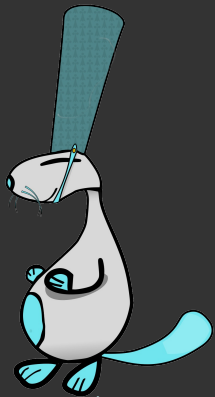
- *Ex : Drew Fisher's talk @ 28C3 on Kinect RE*



Current reverse engineering approach...



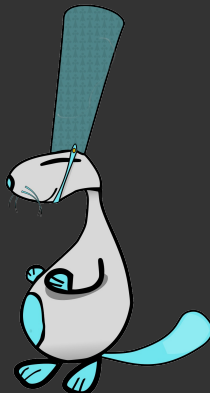
Do you remember the last time you reversed a
protocol ?



Do you remember the last time you reversed a protocol ?

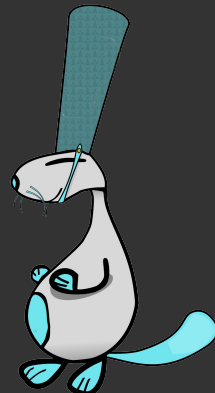
4f945a3c427c99812598f179e0bb1c2722165b3497f4c607deb34efb5ef1878aea36bce6ae7d9be
89546ef16a59c959f592ea5297385abf71c297a8f66ff7ca8f11efff23b441fb4a88bc2b851c14c0b
a2078fcd3b19c25348309bec164aaad64f84f6868d4495bd50d332340276904ef00620800f7ffa2f2
70c5c3c66a3edde50b8347fae58ed2bcb2d287401c86c4e1600e9ff4956ce847ea3138ef80035272
09416f06d6c3b42a4fa7dad6c2aa7f0c3da7caf5d2311dd6d0e8640c298b3d6bd613400cbff19589b
c1200e0142955a7d4ab3e40859d76878af9391c6014700b8ff3ed10cf93d2565eb53c1c8728ac0
b42f37be5199f4aa33400cbff1188df51ce08ce8e78e1b60c1907678b16f1f1b66ed3023929acb2
4ccb894ea3d546f030eba364100beff776f888bec5ed91448b780cd1d382c5a95a0411acd8850927
f5f4b9eb3bf83ff10656f2b734e9f3340f75ba9a62db0f09707adf62feeeda7b887e0d66311e46a9f
479a2ca40c82ab1a948cb4346b891d7888d89e057b521ccb5a759888bb568d2bbec4a406e41fc3028
32af01d017e2a8edbb7501b75b9c0e395095300cffff49015a7447fac2cfba6b836d8e7dbe2497
bbbb0df3b088703f8e8984200bdf9359b4c09979744fff393639023b62190cecac64aeede5d757ac
bffdff7db1004effbc99859183ba69a30f4749400ab2401de222723cb77db211c99d23d5770abe9fc
16a58d51d5ba10377d469ff8e1e0ff8c6afad206c20110ca24f171475905acb33188a2had258d7
119480f8ade92c00d3ff508ca9cec387776fecc080a77ef6e1140062e474ab25fa80c939fe14882
0a691e8a3fe3c9423d986427ba81200edff842d1c115254e573d4a2730927d3e4d03e693300ccffdd
4313000cfff872cc756b40aa3e01490ebcea54cf3def3fc4f91edb5674840d9446204b5667613dad9
4178d0aade5039d0fe3d4662dfbf1043c642d6d34581337eedb736287d9bd35249911a426914e038d
a78b3fc0cca831f8f57bbb4f0dc216bb67c354149ad639add4bd026cefa0110cefa0210100e7ffcc
ebb3990a8ed5e9daf6ff6d0c01663c0f305c1d39110098a0bdec40ed34660306bf0eb09582339d88
ff46e530e92b21bf1fe07e6ed1fe942f0a177a1ba802ab2081573171d1c2c402760900f7ff8deac3
ee58d858d6dae222d1dd32f00d0ff3127e2003add173565570f127c31b0f60ccca7ba97a680750832
79aec8bad48b3929e603c2731480321869fec0df02d45c3a95d3f036c356c984ee031992cd2d53e50
d668979894b8a61b00e4ff523f80e7bb03d57f99445edf68846ff5504454f8774581955b67243f00
7385557232c7f492ead9f9fdbecd27f34200cdf02d5422ffa685f3ec0452554f6c403df9d939655
333f3e210446d6b43674100beff256f071e7047ba8af6d2a131f587baf61b9e286f0d5c6c24579a3
01fa5b7c5f5469b7af29e972ccb4800b7fadf844279d51bdd46df1c5b1e84ee78d1d75a54d2df372
41800e7ff4871180f123aef0a820fd9805556c4495106b0f6370b5adc1b00e4ffa32993f0c8743f8a
aeacf90a0d7cdca9e9c0a56e6176cbf1bd402d913cceb2216cfb61d97321e19e93368df6e7706728d
888a9c9f56f879976a0bd34c00a00f5ffcafc05ef8f1118fe604c0800f7ffcfacacbdb8a02621800
9e8b0c038aad085d41100eeff233a767ce75b801493af803e57a76683e31400ebff8baed3dfedc446

SIZE(X)
B64(Y)
GZIP(BASE64(Y))



Do you remember last time you reversed a protocol ?

- ▶ Complex
- ▶ Time-consuming
- ▶ Mostly Manual

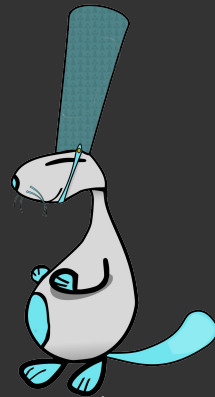


netzob.org

Do you remember last time you reversed a protocol ?

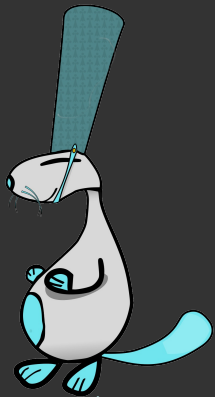
- ▶ Complex
- ▶ Time-consuming
- ▶ Mostly Manual

MOSTLY VISUAL



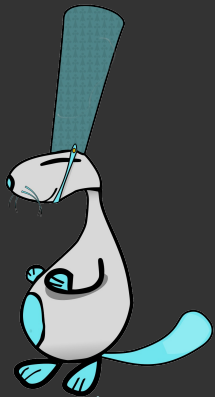
netzob.org

But wait... Couldn't we automate
many RE tasks ?



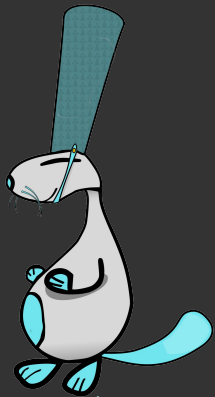


Examples of tasks we would like to automate



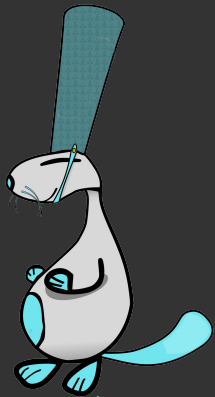


... capture **samples** from its network communications



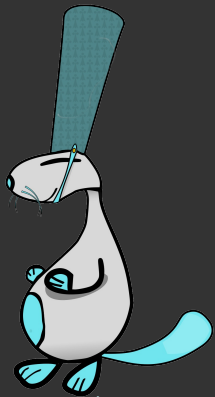


... split messages in « equivalent » groups



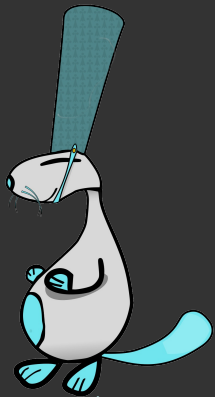


... understanding field semantics



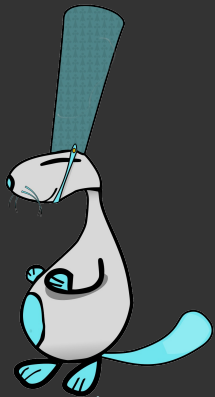


... find **size fields** and associated payloads



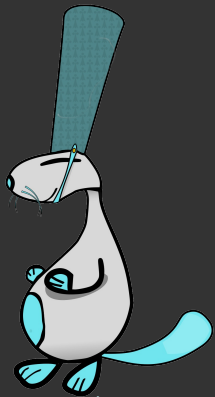


... find CRCs, hashes and other
relations between bits



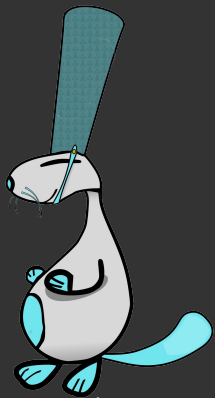


... understand the **valid** sequences of
exchanged messages





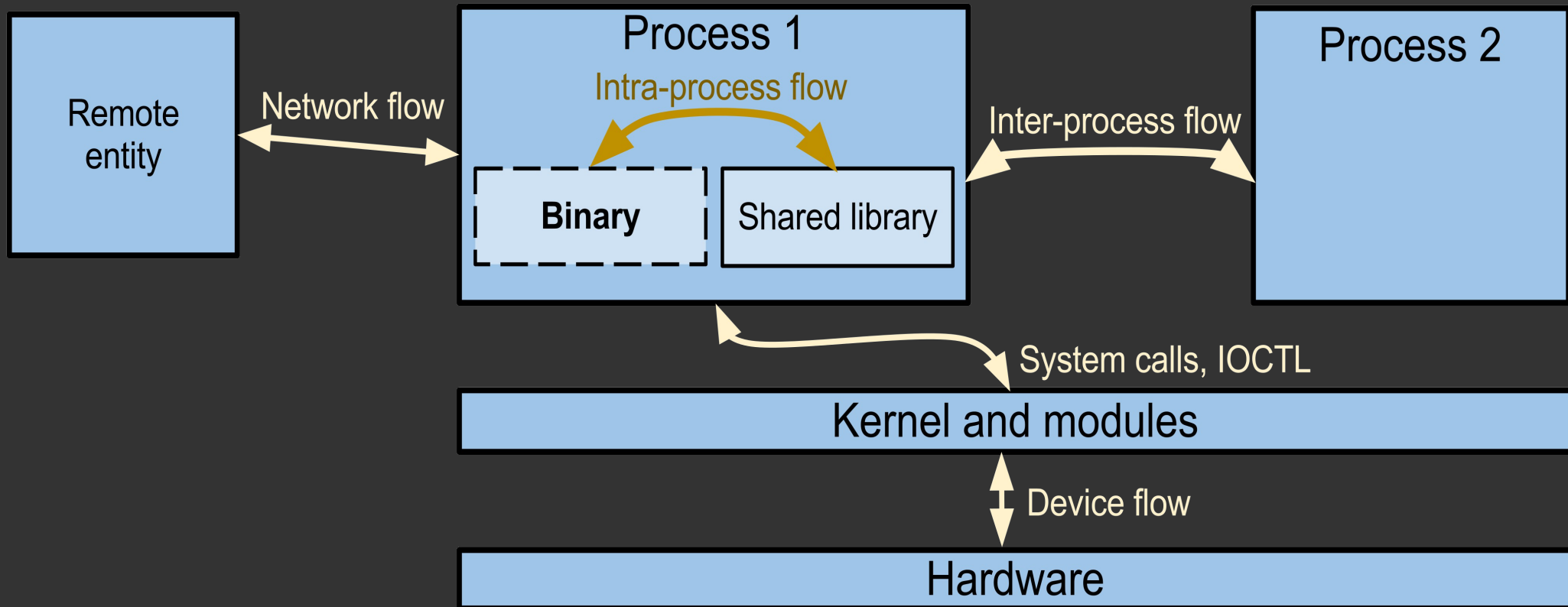
... simulate **realistic** and **controllable** actors



Some reminders about protocols

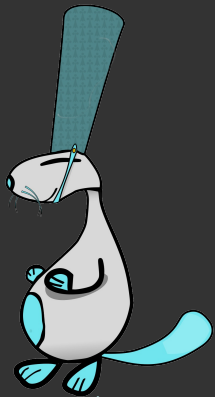
A black and white photograph showing a rectangular sign with the text "SECURITY WILL BE RIGHT BACK" in large, capital letters. The sign is placed on a lawn. In the background, a chair is visible. The text "SECURITY WILL BE RIGHT BACK" is centered on the sign.

SECURITY
WILL BE
RIGHT BACK



Protocols are everywhere

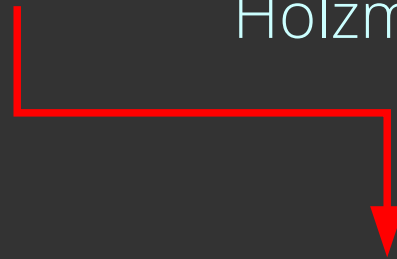
Before reversing we need a model for
protocols



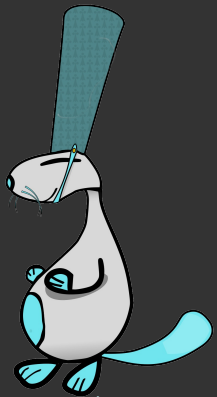


Academics are very good with models :)

« *Design and Validation of Computer Protocols* » by G.
Holzmann

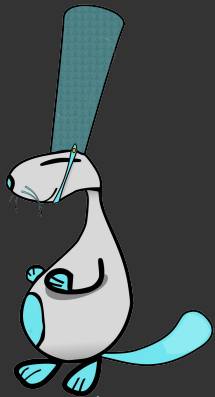


A Communication Protocol is made of
5 distinct parts ...

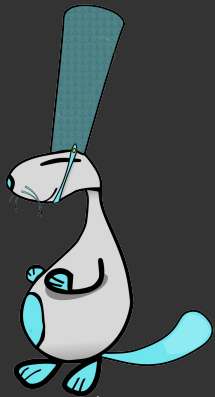


netzob.org

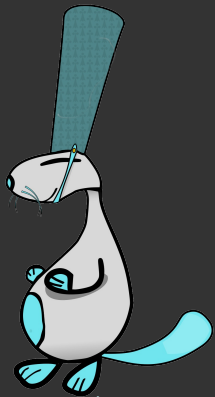
a service (1/5)



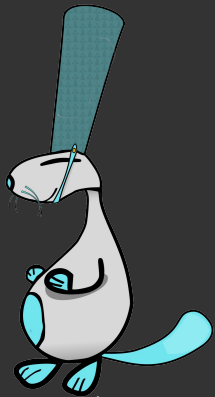
some **assumptions** about the
environment (2/5)



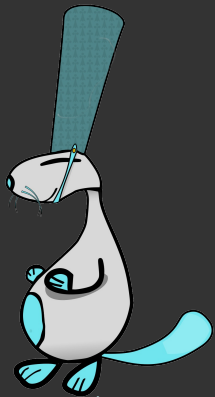
a vocabulary of messages (3/5)



the **encoding** (format) of each message
(4/5)

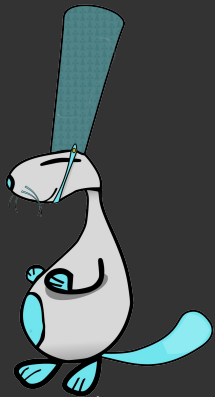


the procedure rules (5/5)

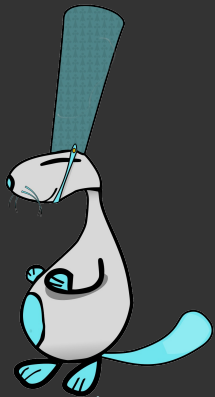
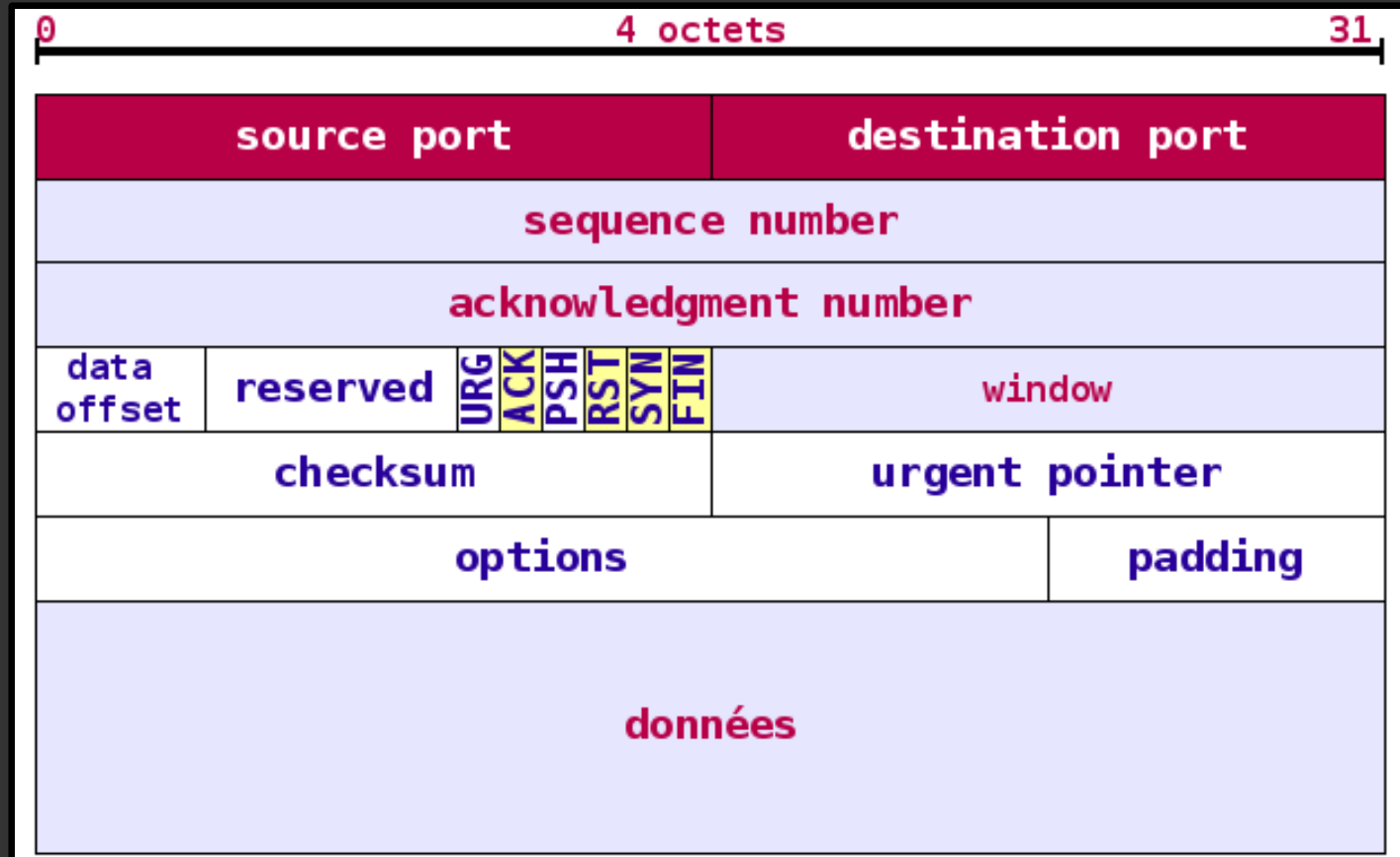


Reduced model for a Protocol

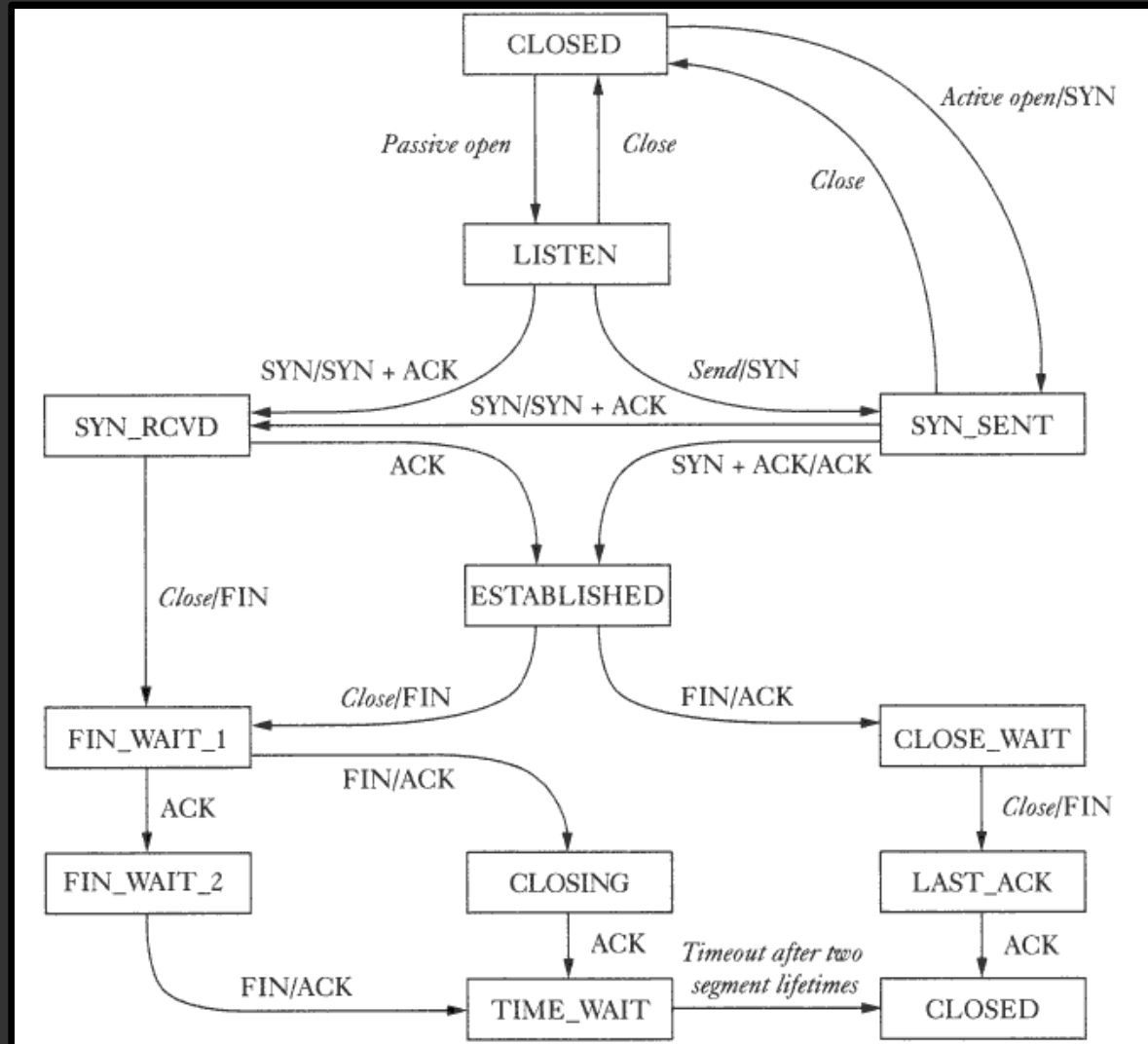
- a vocabulary → a list of **Message Format**
- a grammar → **State Machine**

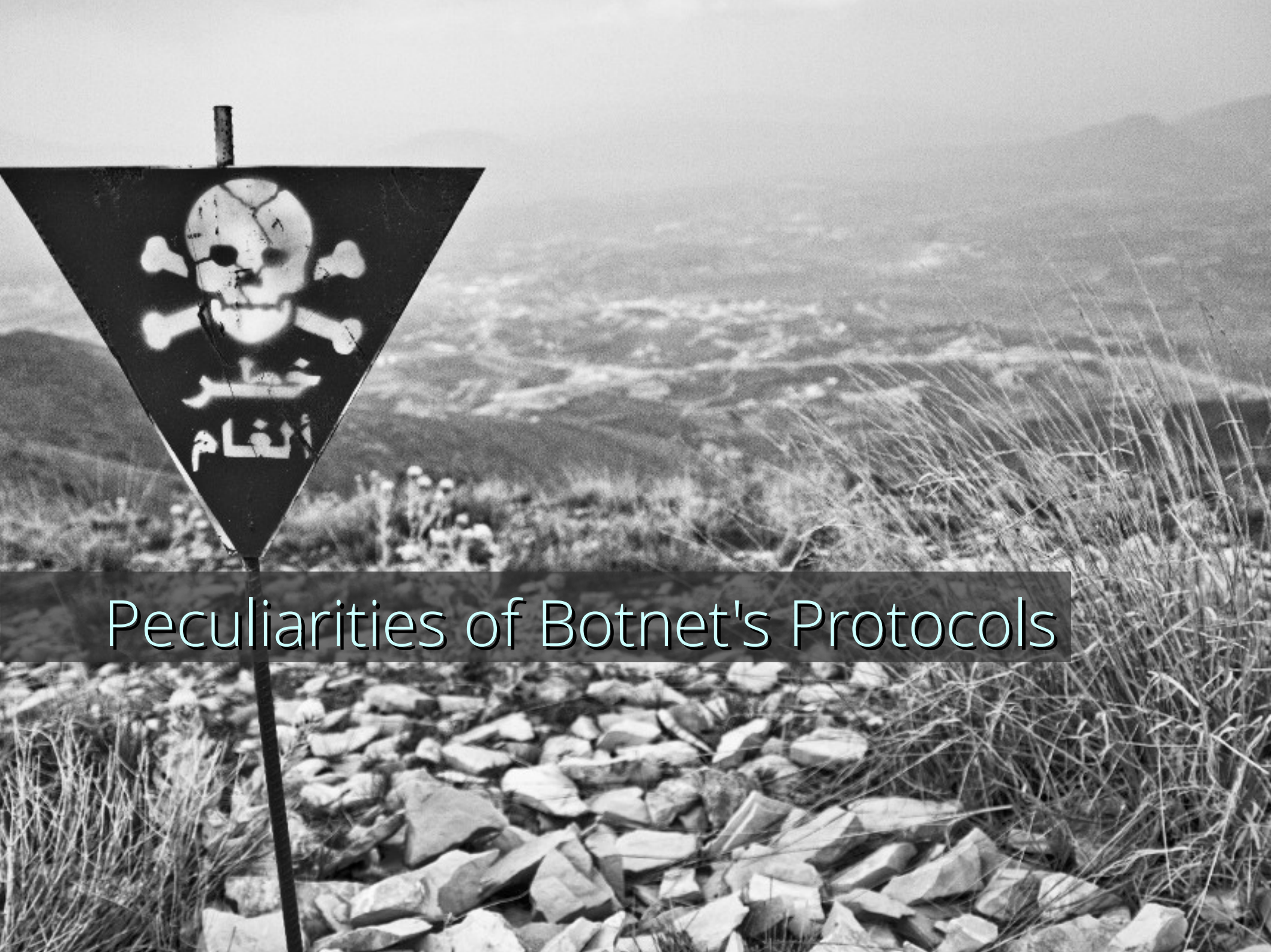


Message Format



State Machine

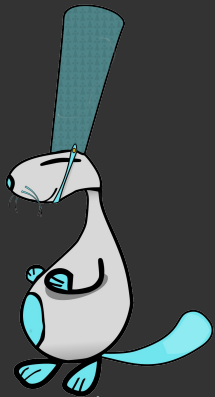




Peculiarities of Botnet's Protocols

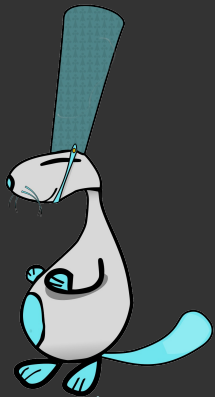
Focus on C&C protocols

- *Send orders to zombies*
- *Receive status from zombies*
- *Zombie « Heartbeat » channel*



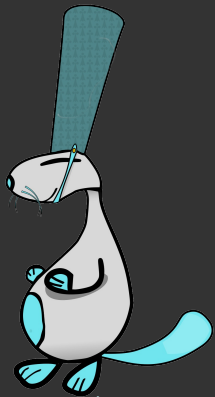
Focus on C&C protocols

- *Send orders to zombies*
- *Receive status from zombies*
- *Zombie « Heartbeat » channel*



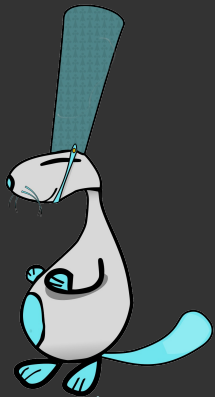
Focus on C&C protocols

- *Send orders to zombies*
- *Receive status from zombies*
- *Zombie « Heartbeat » channel*



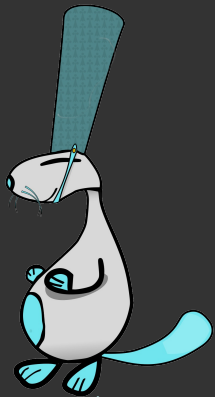
Two types of protocols

- *ASCII explicit messages*
- *High diversity of messages*
- *High interactivity in C&C*
- *Single version*
- *Very-little encryption*
- *Binary messages*
- *Low diversity of messages*
- *Low/No interactivity in C&C*
- *Multiple versions*
- *Little encryption*



Two types of protocols

- *ASCII explicit messages*
- *High diversity of messages*
- *High interactivity in C&C*
- *Single version*
- *Very-little encryption*
- *Binary messages*
- *Low diversity of messages*
- *Low/No interactivity in C&C*
- *Multiple versions*
- *Little encryption*





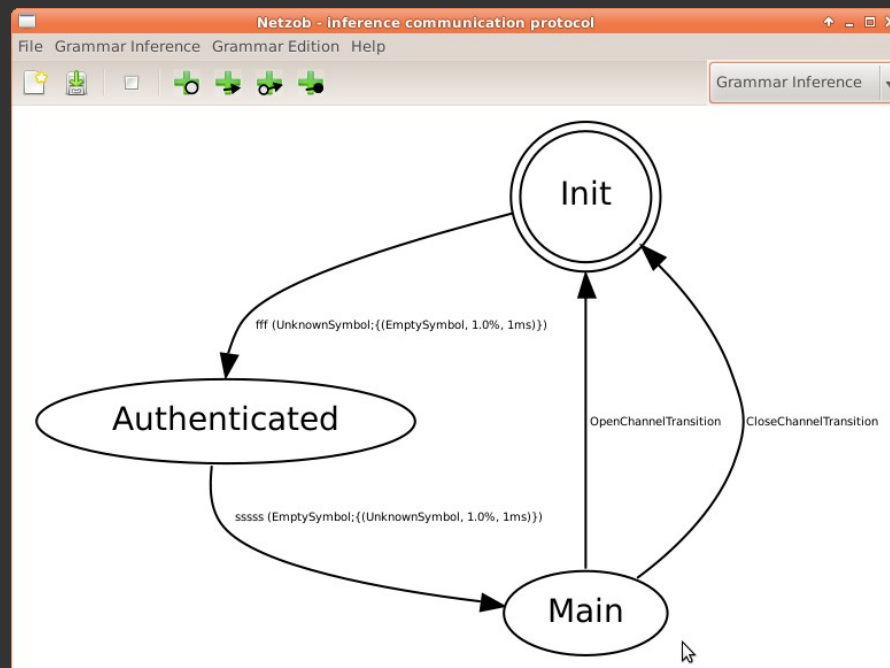
Introducing Netzob ...

Goals of Netzob

- Infer proprietary protocols

Simulate actors of a communication

Smart-Fuzz targeted implementations

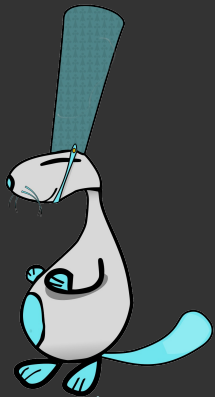
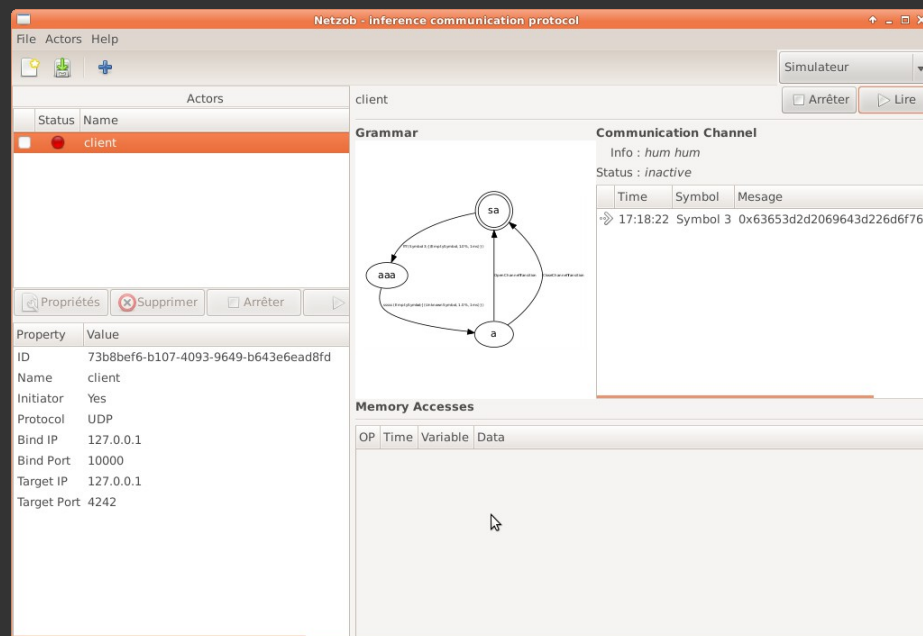


Goals of Netzob

Infer proprietary protocols

- Simulate actors of a communication

Smart-Fuzz targeted implementations

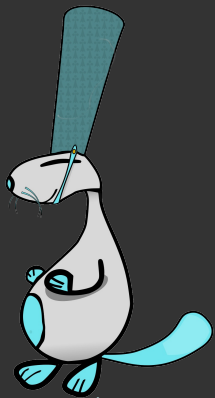
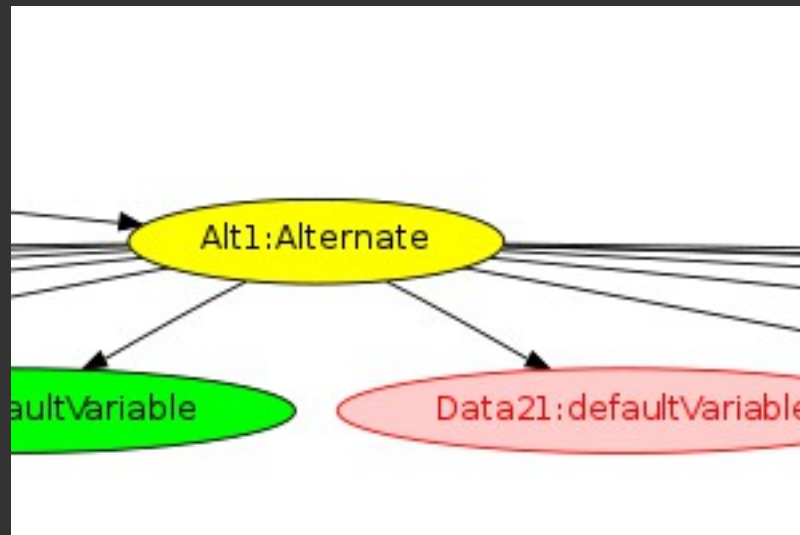


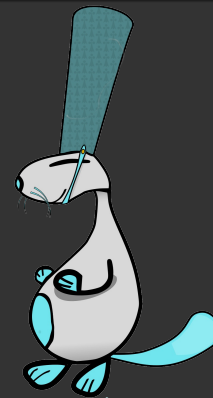
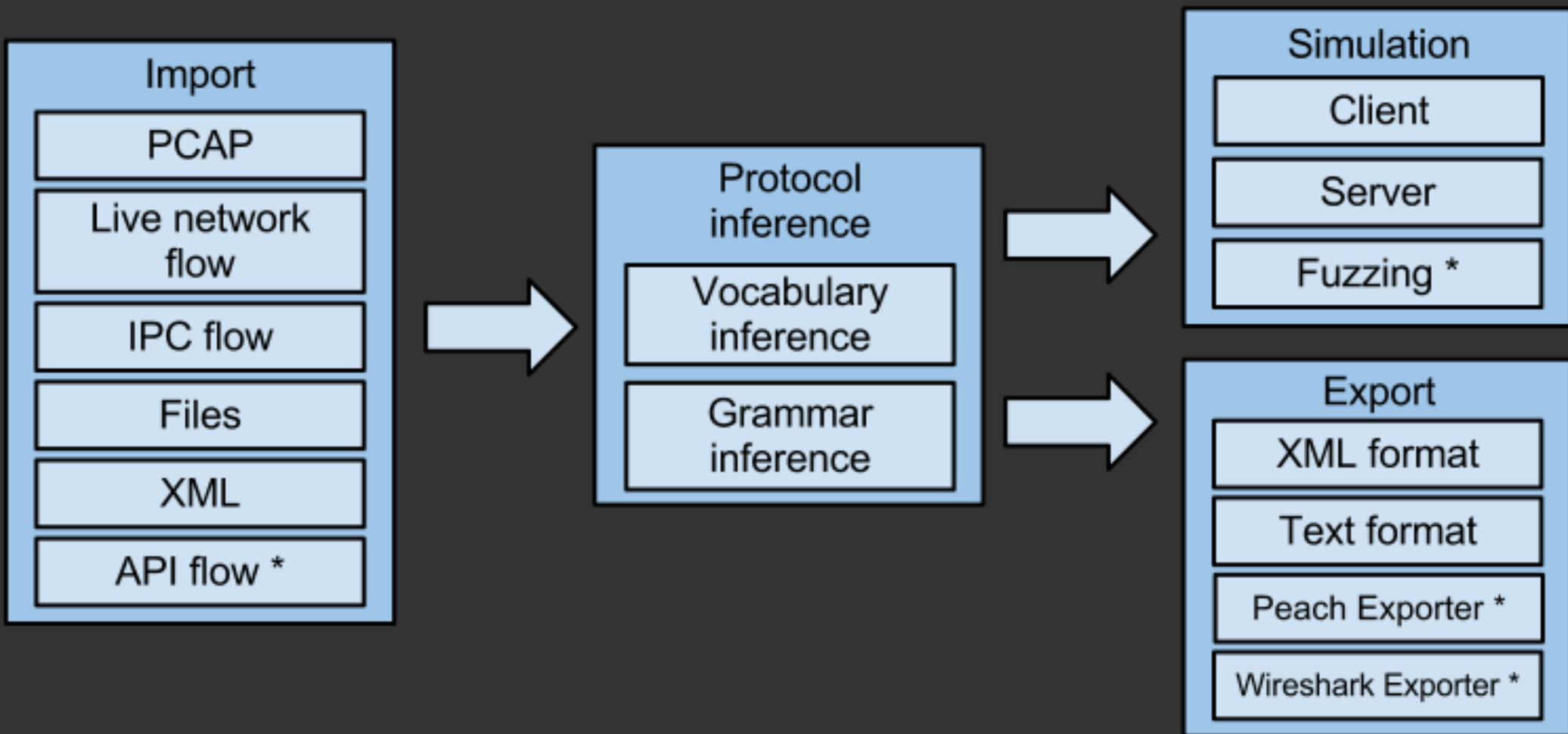
Goals of Netzob

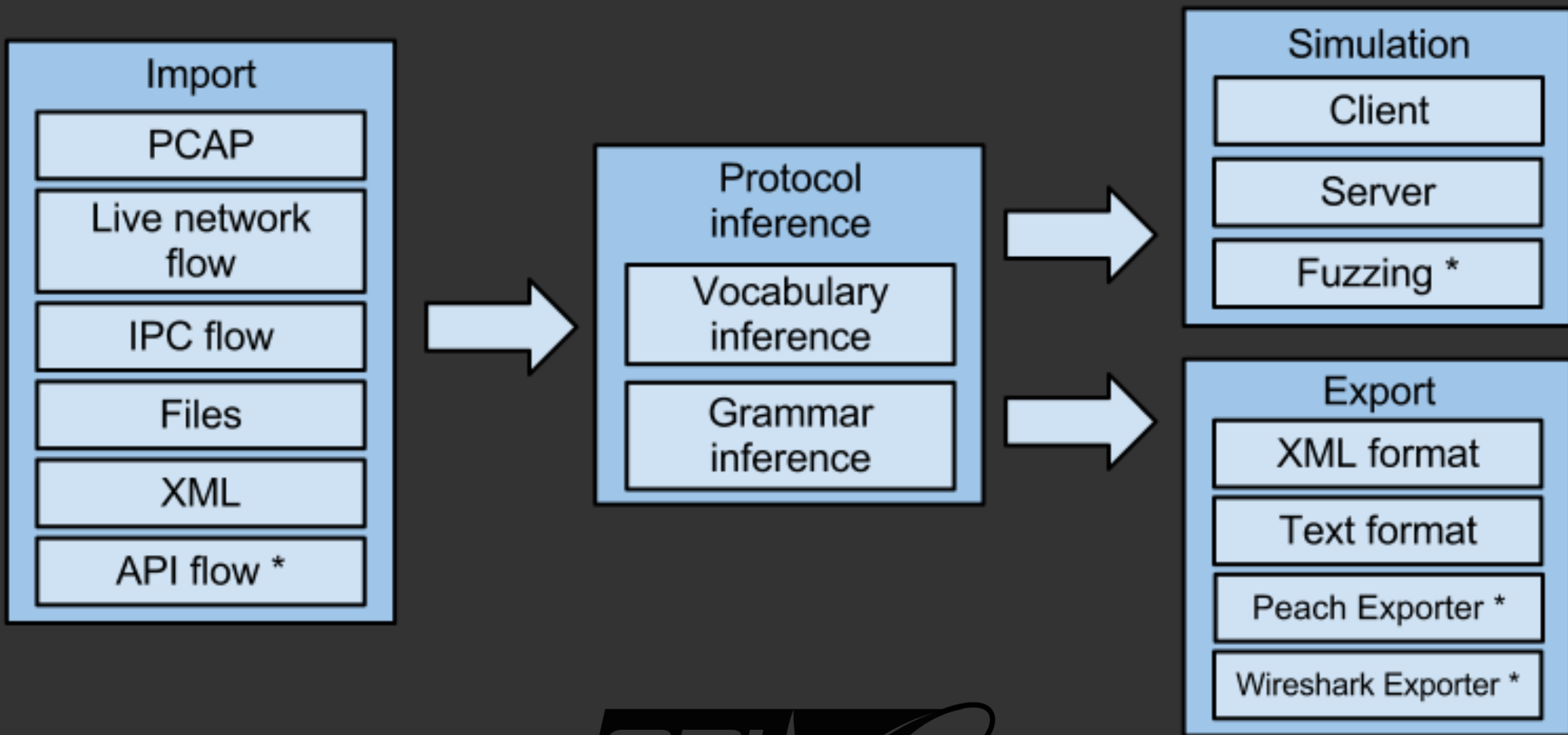
Infer proprietary protocols

Simulate actors of a communication

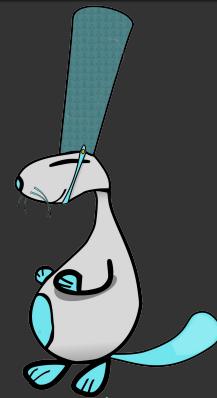
- ▶ Smart-fuzz targeted implementations







GPLv3
Free Software
Free as in Freedom



netzob.org

« State of the art » boundaries



Fuzzing

Language Theory

Netzob

Reverse Engineering

Grammar Inference

Botnet Behavioural Analysis

Sum of human knowledge

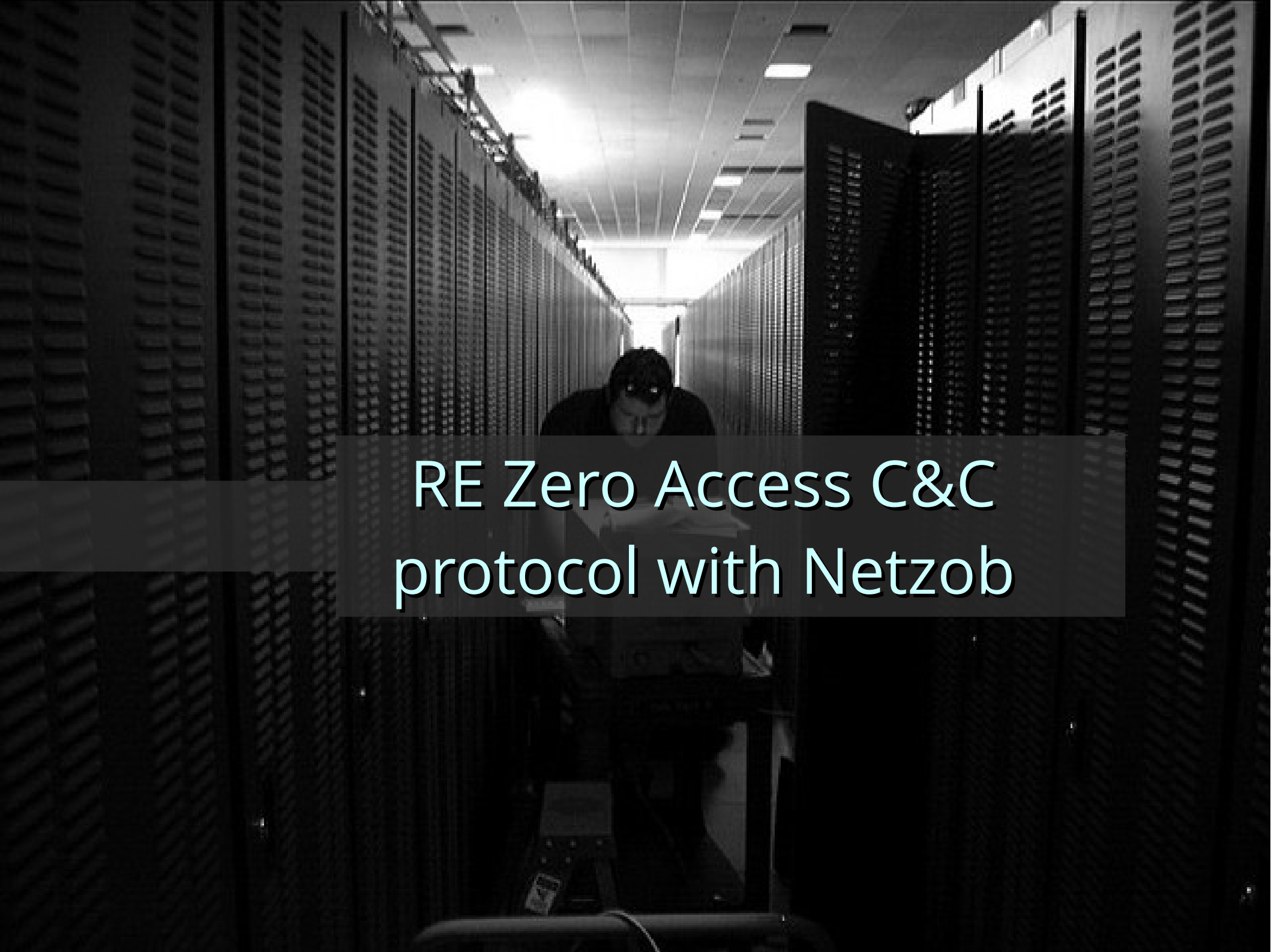
The unknown

NEW « State of the art » boundaries



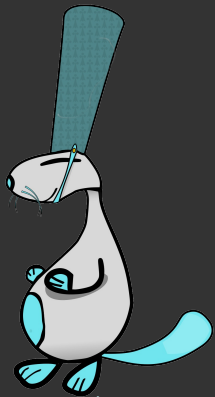
Netzob

The unknown

A black and white photograph of a person in a server room. The person is in the center, looking down at a device. They are wearing a dark shirt and glasses. The room is filled with rows of server racks on both sides, creating a perspective that leads to a bright light at the far end of the aisle. The racks have a perforated metal front. The ceiling has recessed lighting.

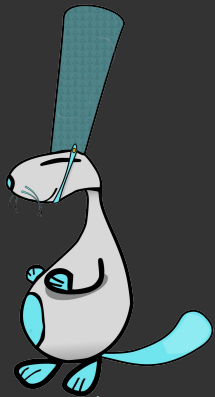
RE Zero Access C&C protocol with Netzob

Requirements

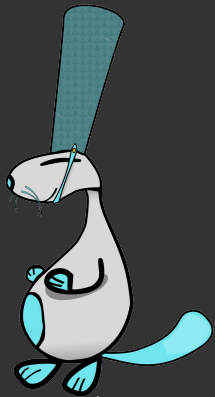


Few **real** communication traces

ZAccess : some traces were provided by *Kevin McNamee* and
from an **infected machine**

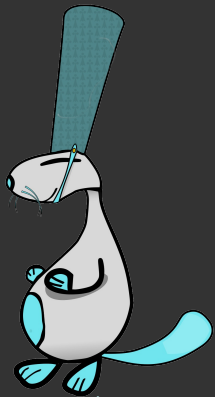


The malware **binary**



A **confined** environment

Adapted Virtual Machines + Firewalls + Torify + management
system

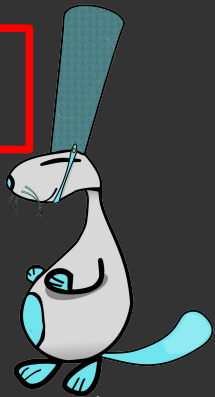


netzob.org

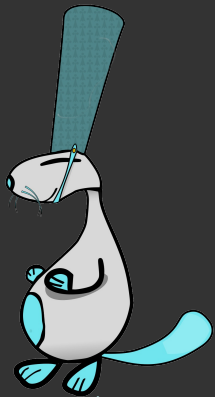
A **confined** environment

Adapted Virtual Machines + Firewalls + Torify + management system

Warning : Consider legal issues before dealing with this !

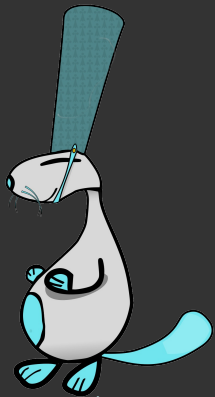


Step 1 : Get messages



Capture **dataflows**

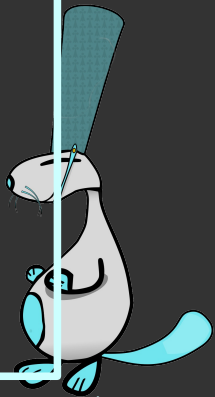
(Network, USB, IPC, API Hooking, Raw files, ...)



Capture **dataflows**

(Network, USB, IPC, API Hooking, Raw files, ...)

```
?..b(.....@.p...aU(.....3.Mi.L..."..f3.8:+.6..d...x,'3...>4...L.....:3.8;..q..d...Q..3.....  
L.QsY.8:3..m.0...d.E....3.Zj."...L.F...8:3-.>ad...s  
.....3%.GgM....|.C.8:...}...o.....(t0...iW.....oj..`?...~!.c.p./e..z.....%gl.T.s.mE..+-.;  
R.{eoDz.}At..d.QP3.SX=):.JU.7`..t...g.V..5.2.....OW..f2p.X8(.....r..7.....S..&8...!  
'..p.....<.  
..g3...C....&(P.T...y.....5..2.....v...&.....iM..e  
....].LP....N..X.SV.JA.M.....#..Y.{....;XVptd....t..  
....=.O..?... Qpga("...;...^:....LU..>!.i.1=...ui.  
L8.7c....@.....}1iN.7...25t.G.)53,.....S.....9!.L5.J.  
?..b(.....@.p.gN..(.....3V...L.....f3.8:?..e..d.....3....x.F.L...u...:3.8;Wu..d.|....3.._.'!..L.[Z.  
8:3.h..4...d.b....3.....L..k.8:3}.e.d.....3%.GgM....|.C.8:...}...o.....(t0...iW.....oj..  
`?...~!.c.p./e..z.....%gl.T.s.mE..+-.;R.{eoDz.}At..d.QP3.SX=):.JU.7`..t...g.V..5.2.....  
OW..f2p.X8(.....r..7.....S..&8...!  
'..p.....<.  
..g3...C....&(P.T...y.....5..2.....v...&.....iM..e  
....].LP....N..X.SV.JA.M.....#..Y.{....;XVptd....t..  
....=.O..?... Qpga("...;...^:....LU..>!.i.1=...ui.  
L8.7c....@.....}1iN.7...25t.G.)53,.....S.....9!.L5.J.
```



Split **dataflows** in **messages**

(sub protocol knowledge, time based, delimiter...)

?..b(.....@.p...aU(.....3.Mi.L..."..f3.8:+.6..d...x,'3...>4...L.....:3.8;;q..d...Q.3.....

Message 1

L...Y...3...d.E....3.Zj."...L.F...8:3->ad...s

...8:...}...o.....(t0...iW.....oj..`?...~!.c.p./e..z.....%gl.T.s.mE..+-.;

R.{eoDz.}At..d.QP3.SX=):.JU.7`.t...g.V.5

Message 2

'p.....<.

..g3...C....&(P.T...y....5..2.....v...&.....iM..e

....].LP....N..X.SV.JA.M.....#Y.{...;XVptd....t..

....=.O..?... Qpga("...;...^:....LU..>!.i.1=...ui.

L8.7c....@.....}1iN.7...25t.G.)53,.....S.....9!.L5.J.

?..b(.....@.p.gN..(.....3V...L.....f3.8?:e..d.....3....x.F.L...u...:3.8;Wu..d.|....3.._'!...L.

[Z.

Message 3

8:3...L...k.8:3}.e.d.....3%.GgM....|.C.8:...}...o.....(t0...iW.....oj..

`?...~!.c.p./e..z.....%gl.T.s.mE..+-.;R.{eoDz.}At..d.QP3.SX=):.JU.7`.t...g.V..5.2.....

OW..f2p.X8(.....r..7.....S..&8...!..

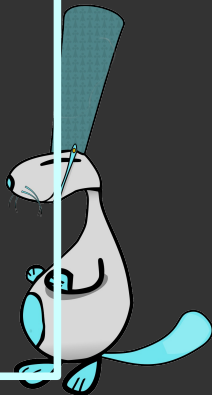
'p.....<...g3...C....&(P.T...y....5..2.....v...&.....iM..e

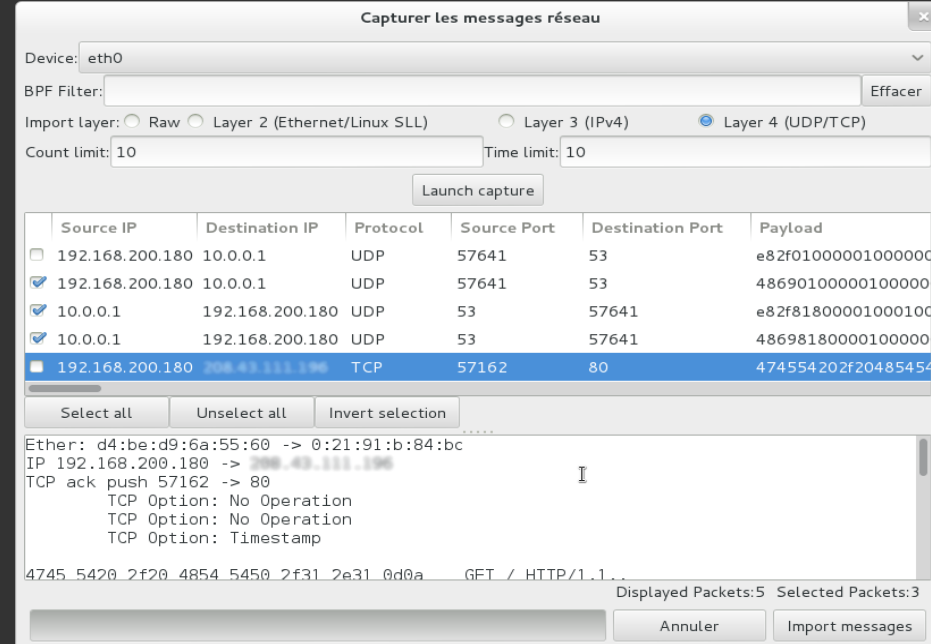
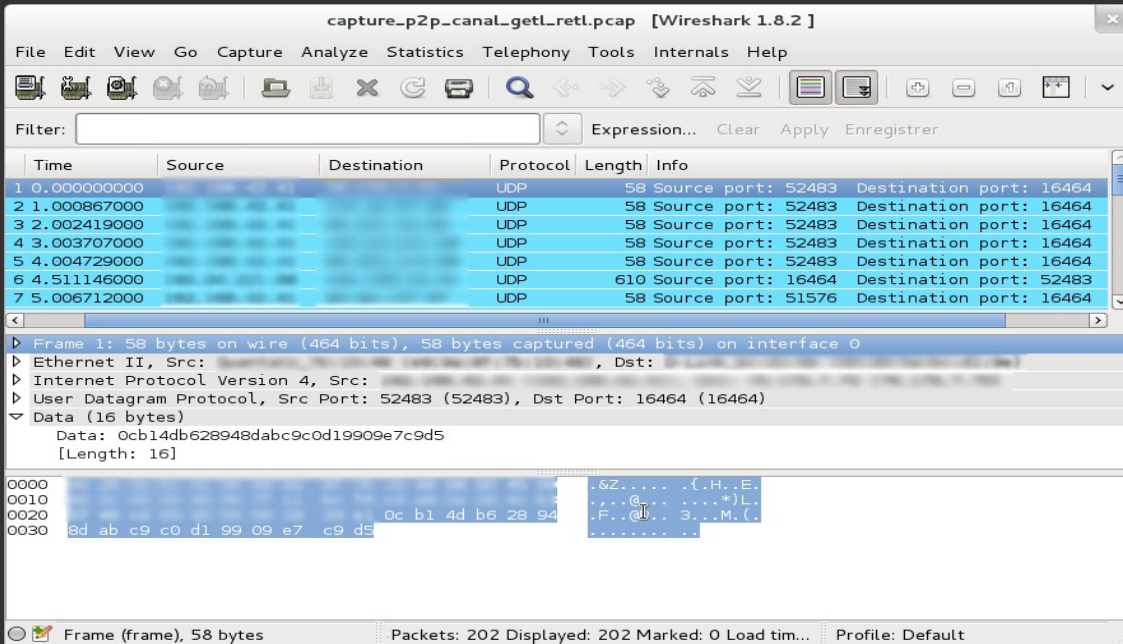
....].LP....N..X.SV.JA.M.....#Y.{...;XVptd....t..

....=.O..?... Qpga("...;...^:....LU..>!.i.1=...ui.

Message 4

L8.7c....@.....}1iN.7...25t.G.)53,.....S.....9!.L5.J.





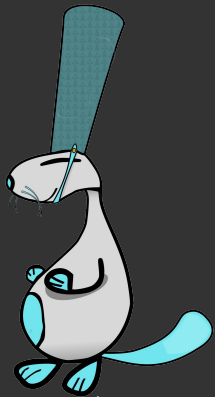
Import

Capture

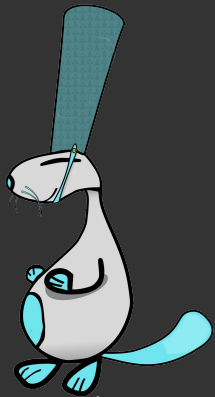
Netzob framework

Filter imported messages
Choose layer of import

Step 2 : RE vocabulary



Abstract messages

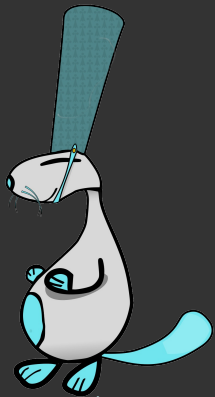


1 message = a sorted **received or sent**
sequence of bits



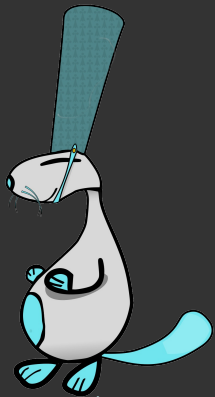
specific to a context

Emails, IPs, Timestamps, BID, AddID, ...

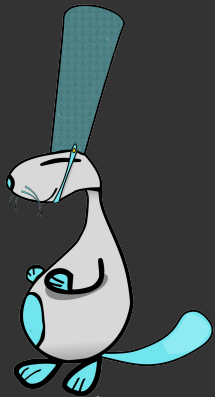


netzob.org

We have to **decontextualize** messages
and regroup **similar** ones



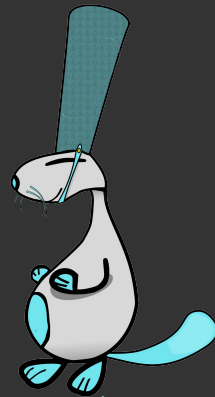
Messages are splitted in **Fields** using



Messages are splitted in **Fields** using

- Simple Alignment
- Delimiter-based Alignment
- Sequence Alignment

3e341eb5ce	4c068e	c2d5baed3a	331938	3b108271e8
dc18fcb8ce	4c068e	2da8f3e33a	331938	c f48cd8fe8
dc18fcb8ce	4c068e	2da8f3e33a	331938	c f48cd8fe8



Messages are splitted in **Fields** using

- Simple Alignment
- Delimiter-based Alignment
- Sequence Alignment

cfa0	4c	5519e43e2e27fa6916313dc89ac77569a00d	4c	38f
cfa0	4c	5519e43e2e27fa6916313dc89ac77569a00d	4c	38f
cfa0	4c	5519e43e2e27fa6916313dc89ac77569a00d	4c	38f

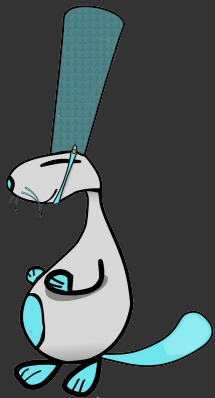


Messages are splitted in **Fields** using

- Simple Alignment
- Delimiter-based Alignment
- Sequence Alignment

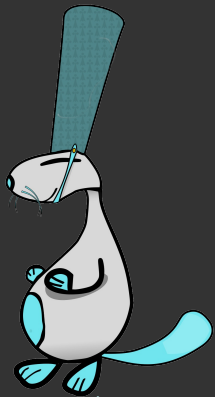
Needleman & Wunsch

3a8	70	832f65bd867ad2	00	d9aedd
3a8	70	832f65bd867ad2	00	d9aedd
	70	c400	00	
	70	c400	00	



Static Fields

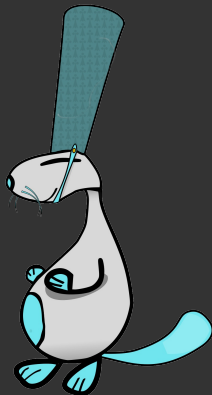
3a8	70	832f65bd867ad2	00	d9aedd
3a8	70	832f65bd867ad2	00	d9aedd
	70	c400	00	
	70	c400	00	



Static Fields

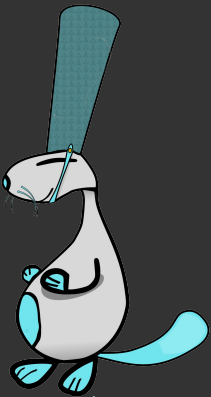
3a8	70	832f65bd867ad2	00	d9aedd
3a8	70	832f65bd867ad2	00	d9aedd
	70	c400	00	
	70	c400	00	

Dynamic Fields



Sequence alignment with Needleman-Wunsh

Idea proposed by Bedoe

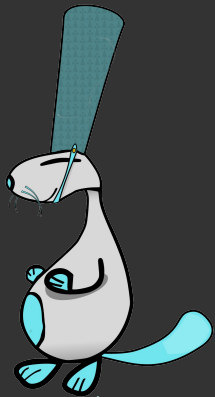


Sequence alignment with Needleman-Wunsh

70 83 2f 65 bd 86 7a d2 00

70 c4 00 00

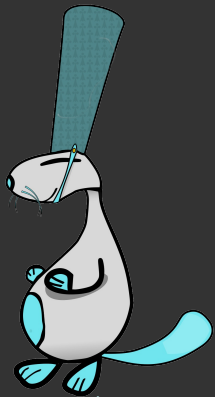
We start with 2 messages



Sequence alignment with Needleman-Wunsh

		70	83	2f	65	bd	86	7a	d2	00
70										
c4										
00										
00										

We build a distance matrix



Sequence alignment with Needleman-Wunsh

		70	83	2f	65	bd	86	7a	d2	00
	0	0	0	0	0	0	0	0	0	0
70	0									
c4	0									
00	0									
00	0									

We initialize the matrix

Sequence alignment with Needleman-Wunsh

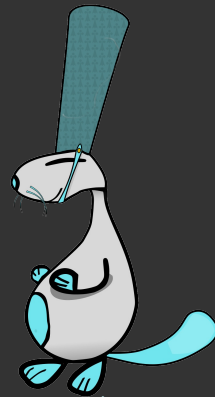
		70	83	2f	65	bd	86	7a	d2	00
	0	0	0	0	0	0	0	0	0	0
70	0	?								
c4	0									
00	0									
00	0									

We fill the matrix with to the formula:

$$M(i,j) = \text{Max}(M(i-1, j-1) + S, M(i, j-1) + W, M(i-1, j) + W)$$

S: Match/Mismatch score (+/- 10)

W: Gap score (0)

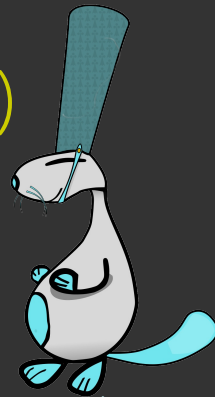


Sequence alignment with Needleman-Wunsh

		70	83	2f	65	bd	86	7a	d2	00
	0	0	0	0	0	0	0	0	0	0
70	0	?								
c4	0									
00	0									
00	0									

We fill the matrix with to the formula:

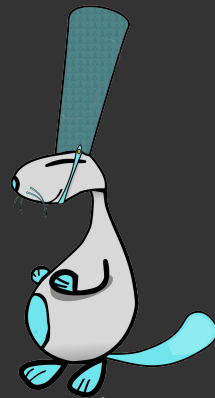
$$M(i,j) = \text{Max}(M(i-1, j-1) + S, M(i, j-1) + W, M(i-1, j) + W)$$



Sequence alignment with Needleman-Wunsh

		70	83	2f	65	bd	86	7a	d2	00
	0	0	0	0	0	0	0	0	0	0
70	0	10	10	10	10	10	10	10	10	10
c4	0	10	10	10	10	10	10	10	10	10
00	0	10	10	10	10	10	10	10	10	20
00	0	10	10	10	10	10	10	10	10	30

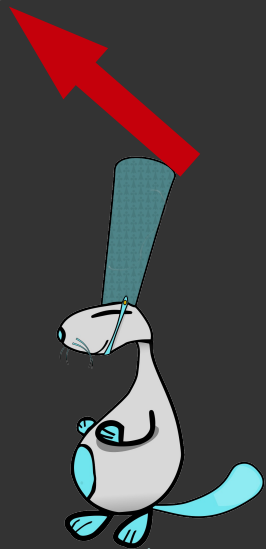
We fill the entire matrix



Sequence alignment with Needleman-Wunsh

		70	83	2f	65	bd	86	7a	d2	00
	0	0	0	0	0	0	0	0	0	0
70	0	10	10	10	10	10	10	10	10	10
c4	0	10	10	10	10	10	10	10	10	10
00	0	10	10	10	10	10	10	10	10	20
00	0	10	10	10	10	10	10	10	10	30

We do a traceback



Sequence alignment with Needleman-Wunsh

		70	83	2f	65	bd	86	7a	d2	00
	0	0	0	0	0	0	0	0	0	0
70	0	10	10	10	10	10	10	10	10	10
c4	0	10	10	10	10	10	10	10	10	10
00	0	10	10	10	10	10	10	10	10	20
00	0	10	10	10	10	10	10	10	10	20

We compute the common pattern

70 83 2f 65 bd 86 7a d2 00

70 c4 00 -- -- -- -- 00

Sequence alignment with Needleman-Wunsh

We finally build a regex for the model

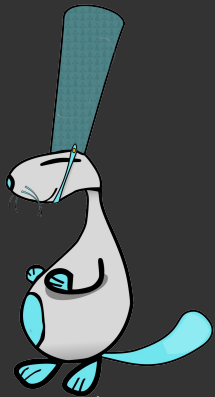
70 83 2f 65 bd 86 7a d2 00

70 c4 00 -- -- -- -- 00

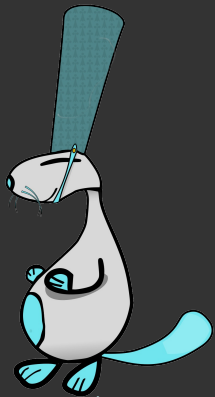
(70)

(.*{2,7})

(00)



How to evaluate messages similarities ?

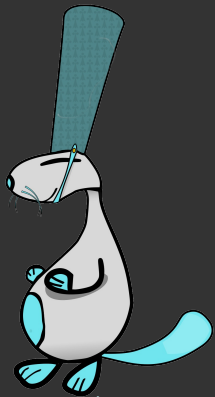


Measure of the Quality of Symbols

0 % < **Similarity Score** < 100 %

Messages have
Nothing in common

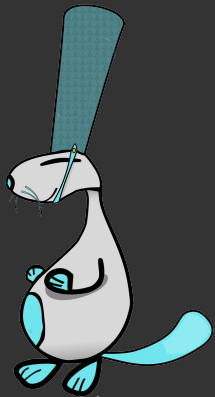
Messages are
identicals



Similarity factors between messages

Currently two factors are used

- ▶ F1: ratio of dynamic fields / static bytes
- ▶ F2: ratio of common dynamic bytes

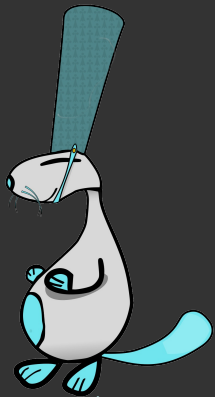


Similarity factors between messages

Currently two factors are used

- ▶ F1: ratio of dynamic fields / static bytes
- ▶ F2: ratio of common dynamic bytes

The design of Netzob allows for inclusion of future factors



Similarity factors between messages

F1: ratio of dynamic fields / static bytes

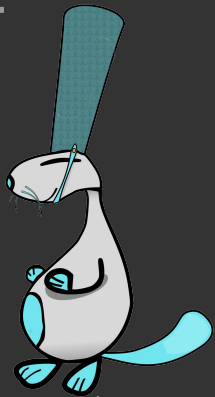
F2: ratio of common dynamic bytes

70 83 2f 65 bd 86 7a d2 00

70 c4 00 -- -- -- -- 00

$$F2 = 100 / 2 * 7$$

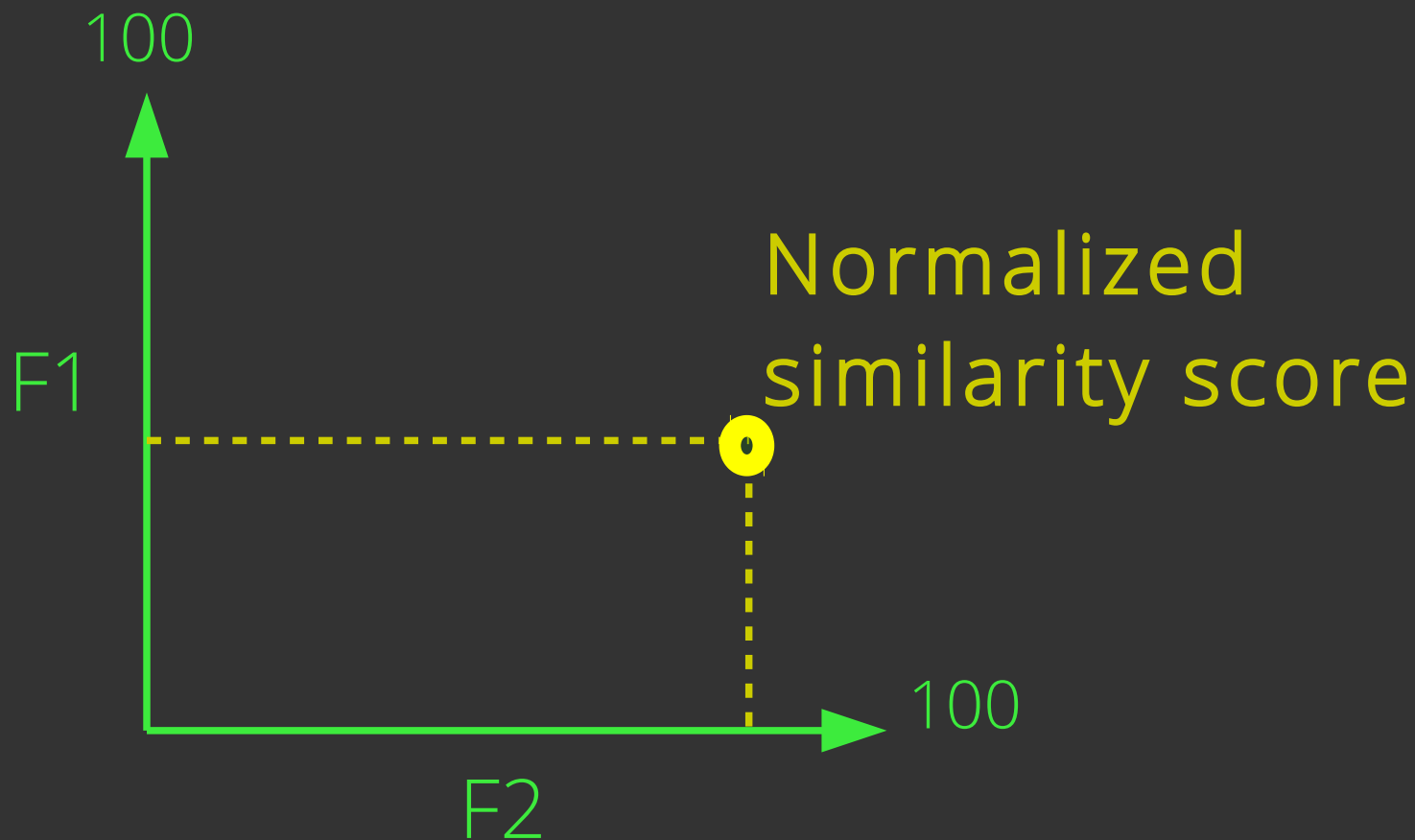
$$F1 = 100 / (1 + 2) * 2$$



Similarity factors between messages

F1: ratio of dynamic fields / static bytes

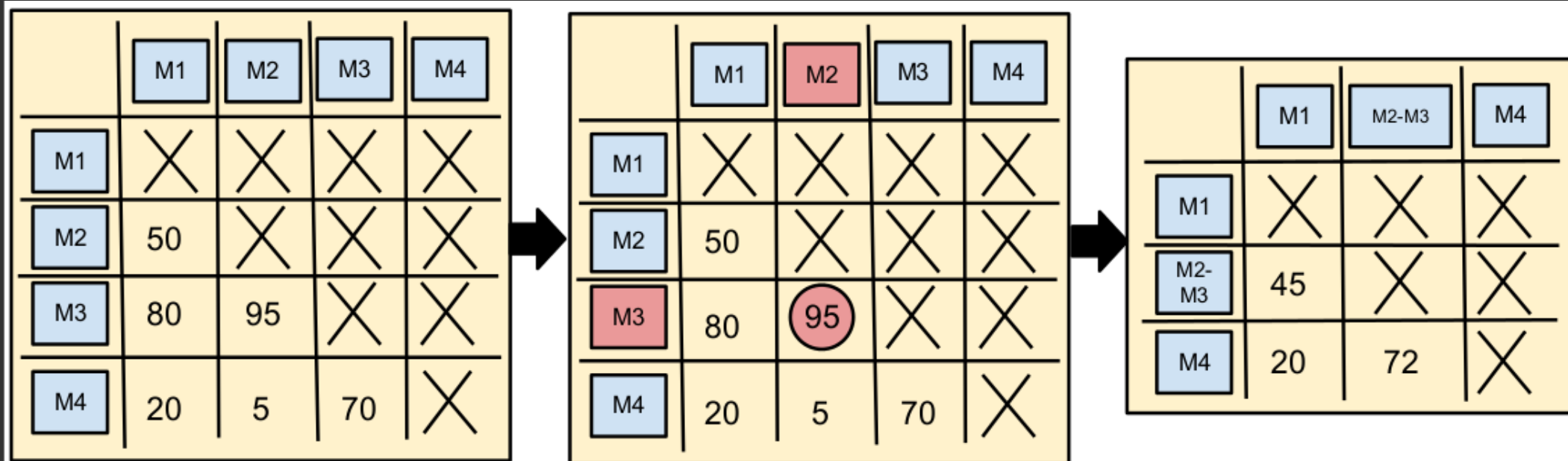
F2: ratio of common dynamic bytes



Hierarchical Clustering by similarities:

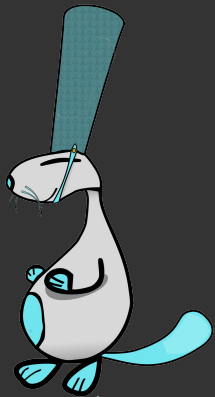
Similarity matrix

UPGMA

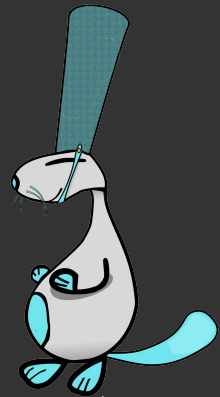
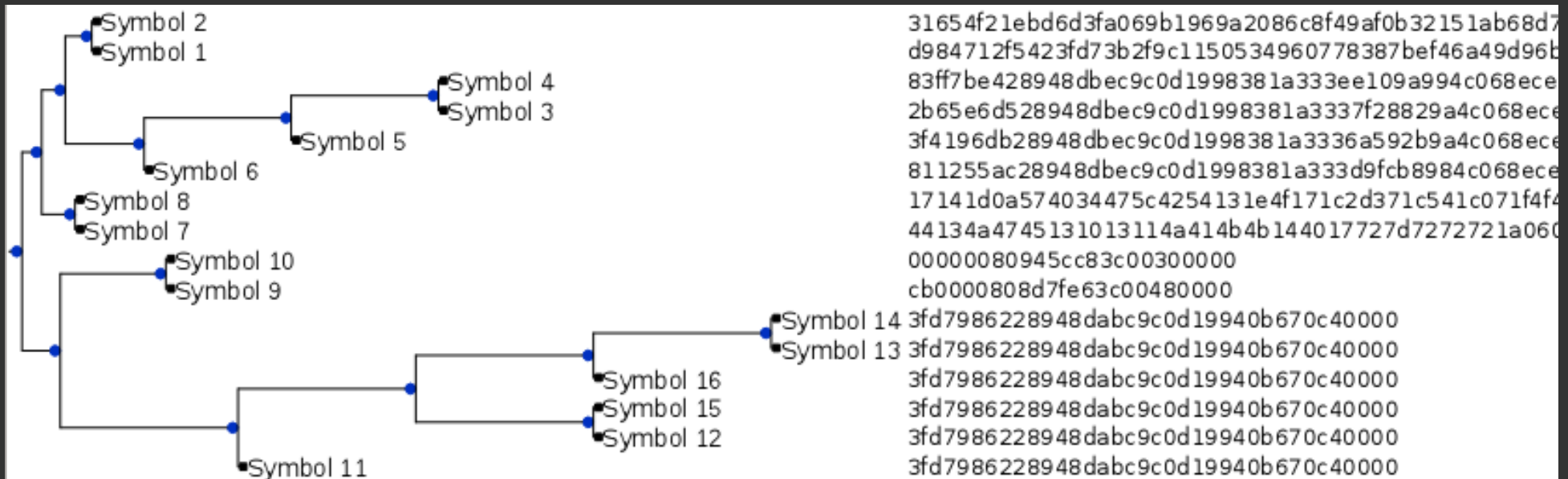


Fill of a similarity matrix

Iteratively merge the 2 most similar messages

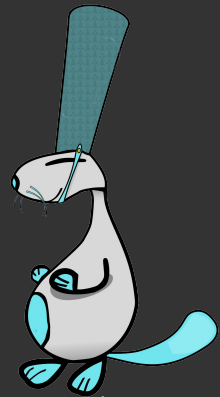
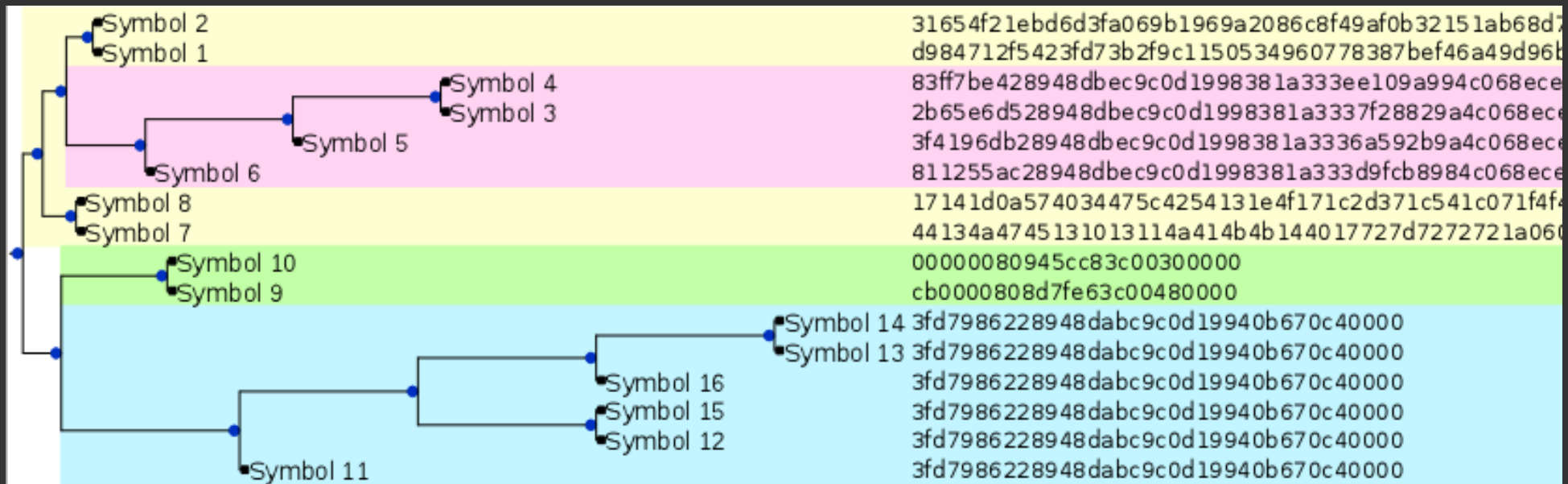


UPGMA creates a similarity tree



UPGMA creates a similarity tree

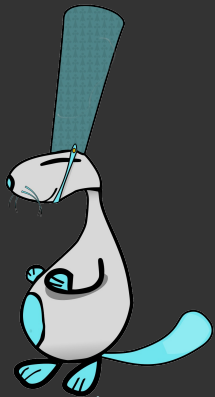
and facilitates clustering



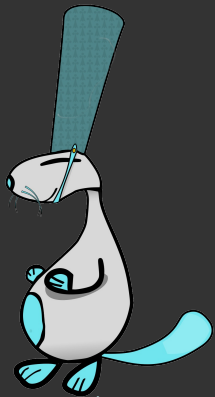
ZAccess Example

Results of Clustering and Sequence Alignment

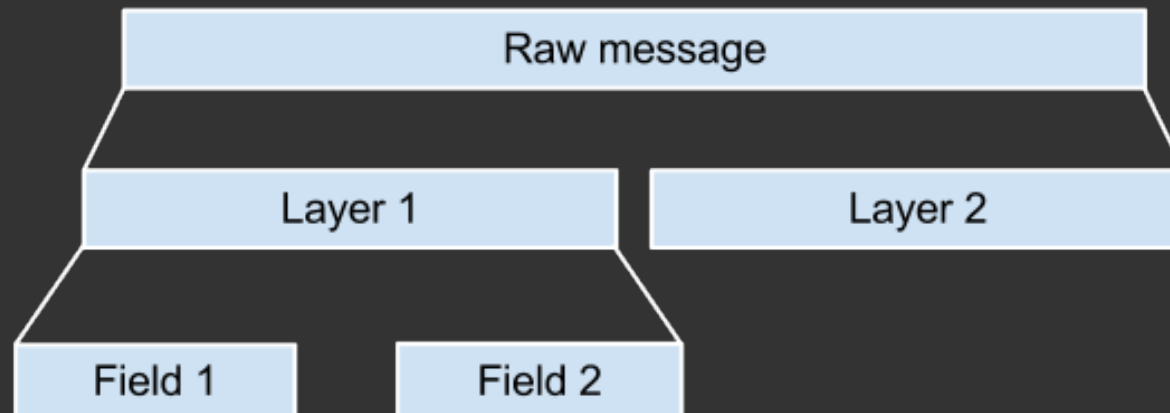
Symboles			Symbol 5						
Nom	Message	Champ	Field 0 (.{8}) hex	Field 1 (4c74657200000000) hex	Field 2 (.{2}) hex	Field 3 (000000) hex	Field 4 (.{8}) hex	Field 5 (00000000) hex	Field 6 (.{8}) hex
Symbol 10	5	3	01a76c9f	4c74657200000000	03	000000	bd584ae8	00000000	d2f1e4c5
Symbol 25	8	3	9782e4fb	4c74657200000000	06	000000	c725c5ca	00000000	b7f1e4c5
Symbol 23	10	5	46b9c0a9	4c74657200000000	05	000000	d0b4d6d1	00000000	be7f1e4c
Symbol 5	8	12	b7dafb61	4c74657200000000	0e	000000	deed851e	00000000	aa7f1e4c
			db34786e	4c74657200000000	0c	000000	e029f3c8	00000000	bd7f1e4c
			738a2cf6	4c74657200000000	07	000000	b6d28e36	00000000	8f7f1e4c
			badfa0e7	4c74657200000000	05	000000	b85b8fda	00000000	cd7f1e4c
			5d2676f0	4c74657200000000	04	000000	954cfe6f	00000000	6f7f1e4c



Abstract fields



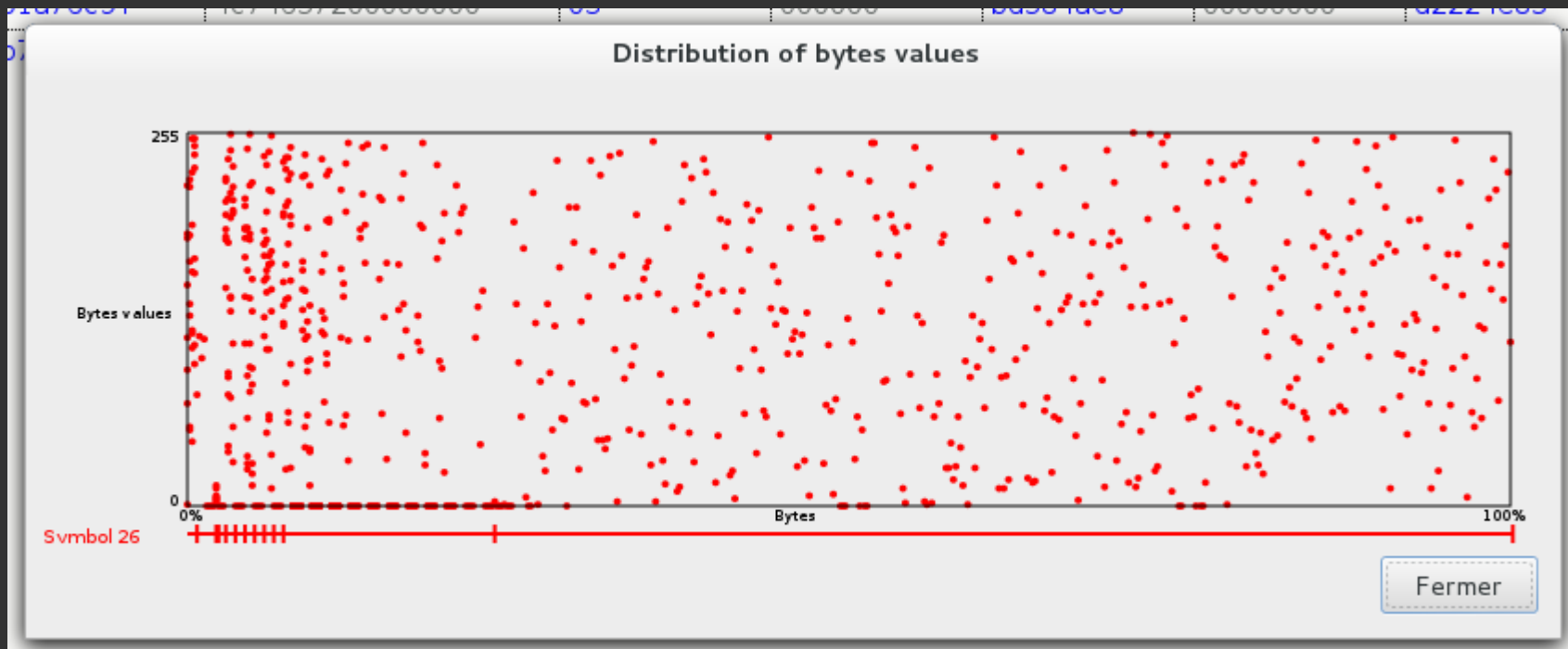
Message format model



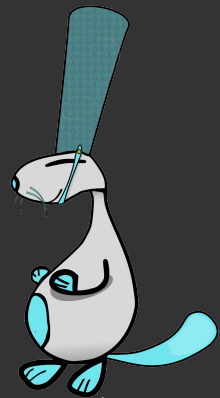
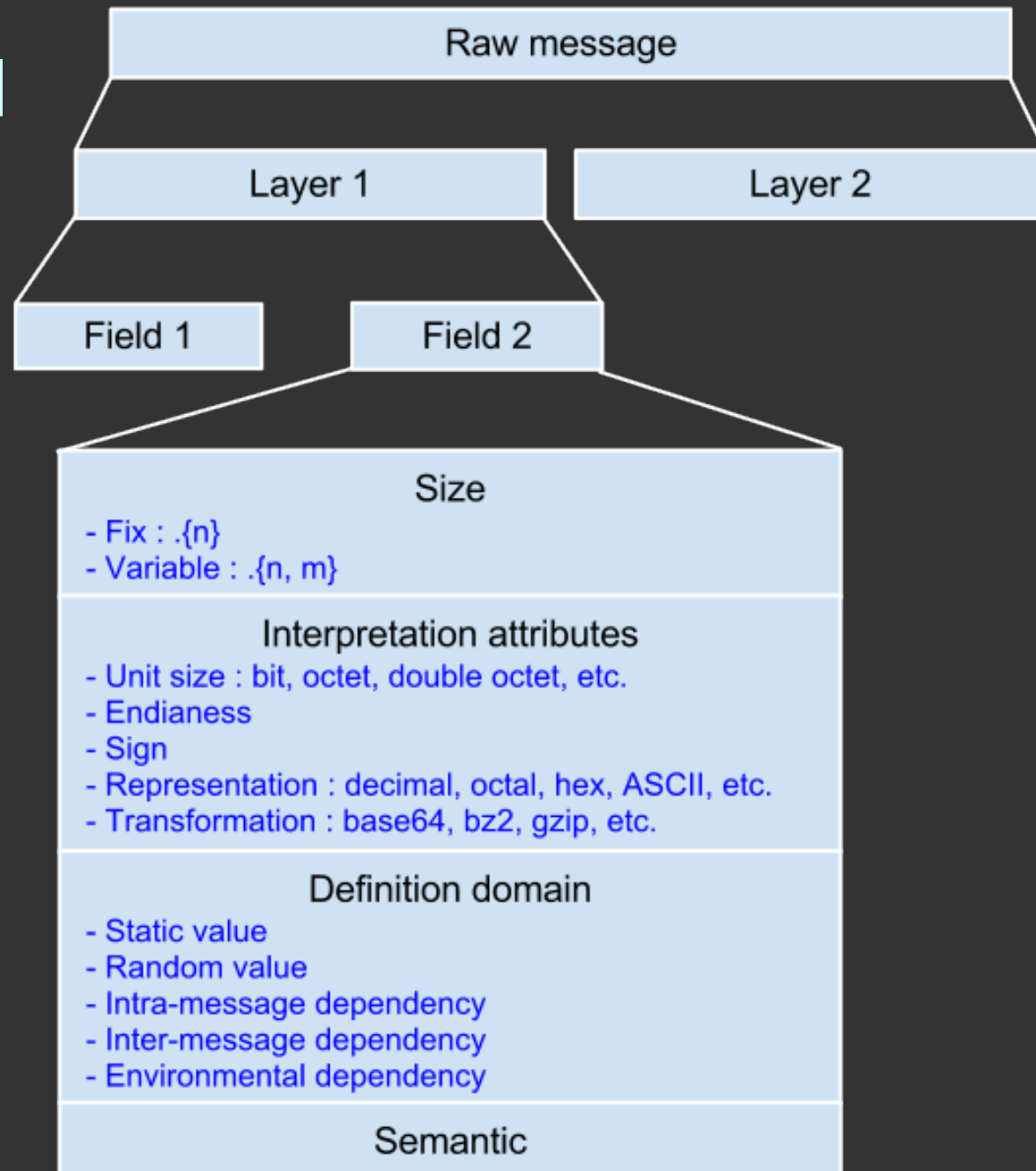
Allows multiple partitionment strategies per symbols

ZAccess Example

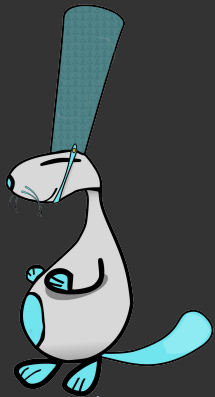
The bytes distribution helps to identify multiple partitionment strategies



Full message format model

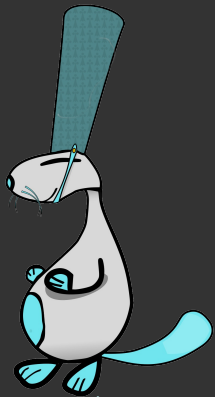


Transformations



The idea: transform raw bytes into application-level bytes

- Applied either on messages, layers or fields
- Examples of provided functions: base64, gzip, bz2



Adding a custom transformation function

Ex: ZeroAccess XOR-based obfuscation

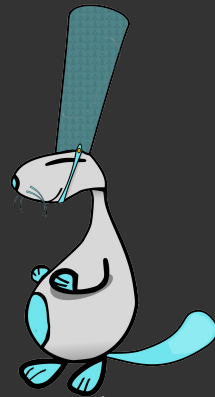
Create a Custom Tra



Custom Transformation Function

Name of the function

```
key=0x66747032
result = []
binMessage = binascii.a2b_hex(message)
for i in range(0,len(binMessage), 4):
    if len(binMessage[i:])>=5 :
        subData = struct.unpack("<I", binMessage[i:i+4])[0]
        xoredSubData = subData ^ key
        result.append(struct.pack("<I", xoredSubData))
        key = ((key << 1) & 0xffffffffL | key >> 31)
strMessage = "".join(result)
message = binascii.b2a_hex(strMessage)
```



Adding a custom transformation function

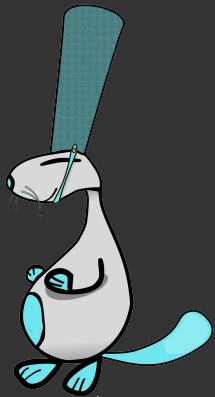
Ex: ZeroAccess XOR-based obfuscation

b59e6155	28948dbec9c0d1998381a333	ad4d699b	4c068ece
4adb017b	28948dbec9c0d1998381a333	9b72a59a	4c068ece
8c43c72c	28948dbec9c0d1998381a333	d9fcb898	4c068ece



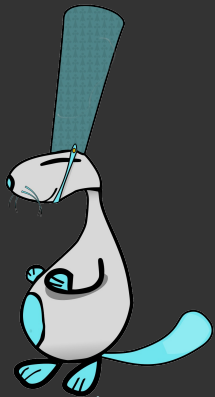
87ee1533	4c746572000000000100000000	8b4e2efc	00000000
78ab751d	4c746572000000000100000000	bd71e2fd	00000000
be33b34a	4c746572000000000100000000	ffffffff	00000000

Search for relations



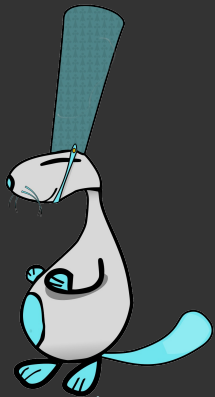
Binary ID, Affiliate ID,
Filenames, *etc.*

« **Inter-Symbol** » and « Intra-Symbol » relations

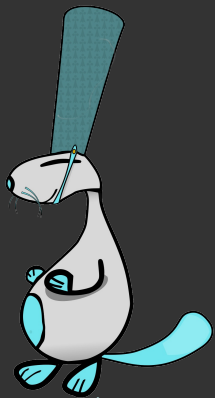


« Inter-Symbol » and « **Intra-Symbol** » relations

└─ Size Fields, CRCs, *etc.*



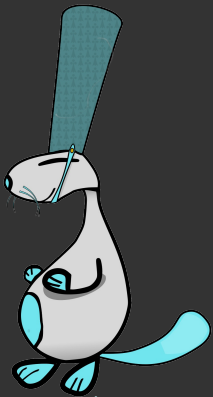
Correlate field's size, values (CRCs,...) with the
« Maximal Information Coefficient » (M.I.N.E. by M.I.T.)



Correlate field's size, values (CRCs,...) with the
« Maximal Information Coefficient » (M.I.N.E. by M.I.T.)



Qualify correlated fields

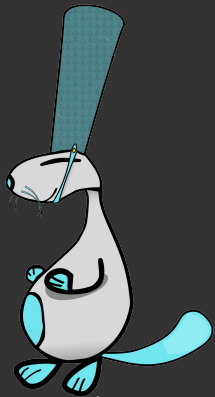


Search for closest pairs :

- Measure **dependence** between each pairs
- rank pairs by their score

Good Points :

- Fast
- Search for dependances
 - Detect linear, non-linear, periodic relations, ...
- Support Noisy datasets

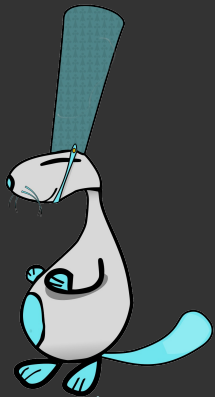


Search for closest pairs :

- Measure **dependence** between each pairs
- rank pairs by their score

Good Points :

- Fast
- Search for dependances
 - Detect linear, non-linear, periodic relations, ...
- Support noisy datasets



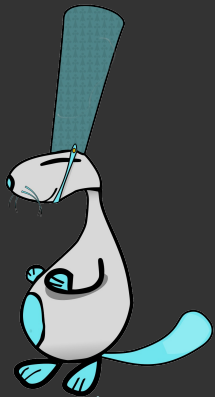
Generate Pairs of data :

Simple way :

- **Value** of each field
- **Size** of each dynamic field

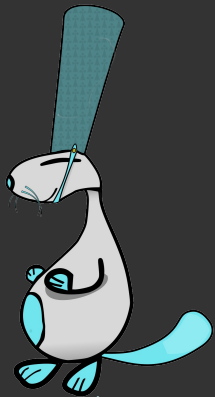
Add more :

- Concat fields
- Create n-grams (4bits, 8bits, ...)
- Consider CRCs, MD5, SHA1,

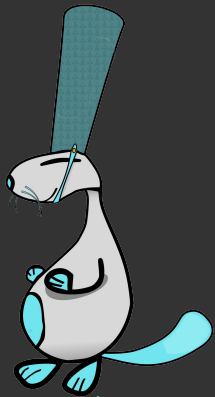


« **Environmental** » dependencies

Search for messages' meta-informations in data

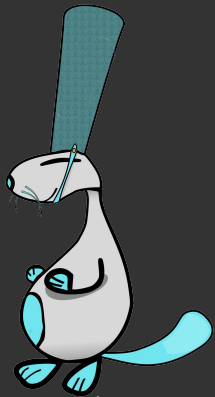


Step 3 : RE grammar



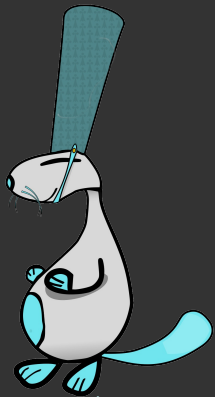
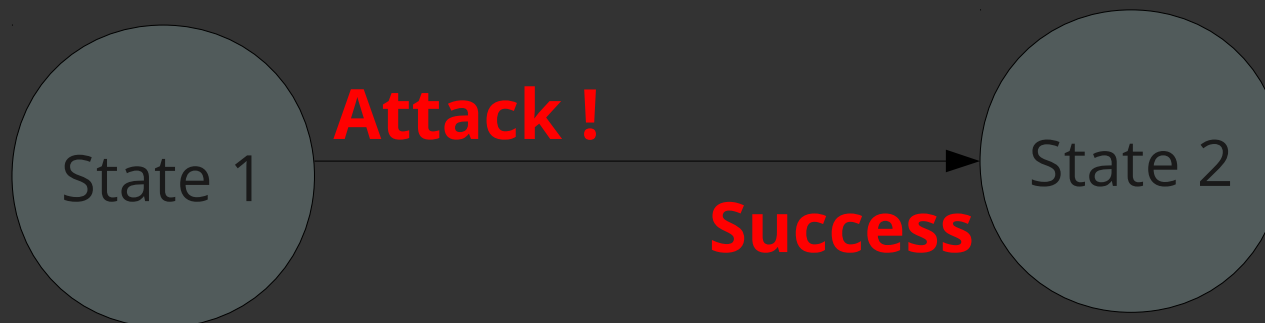
Sequence of valid exchanged symbols.

→ IO Automata

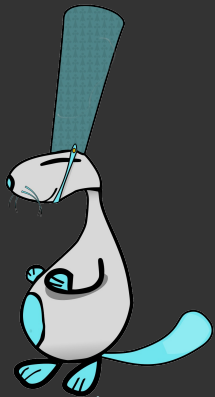
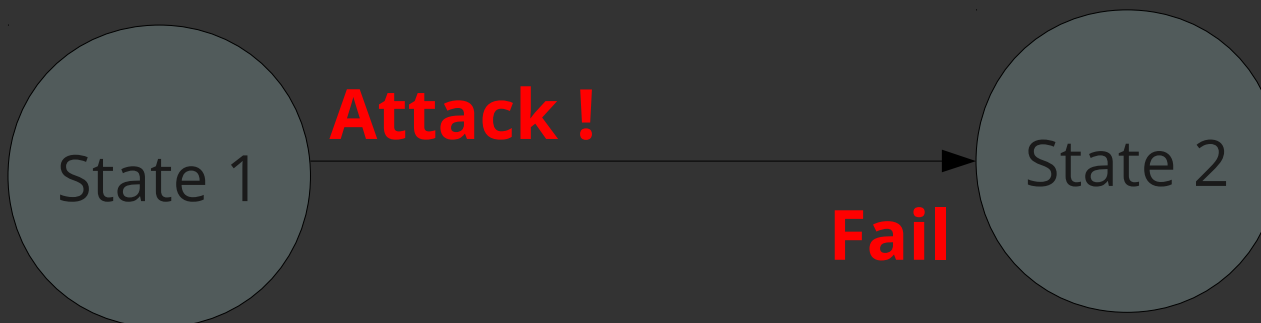


Sequence of valid exchanged symbols.

→ **IO Automata**

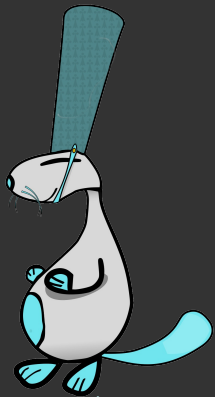
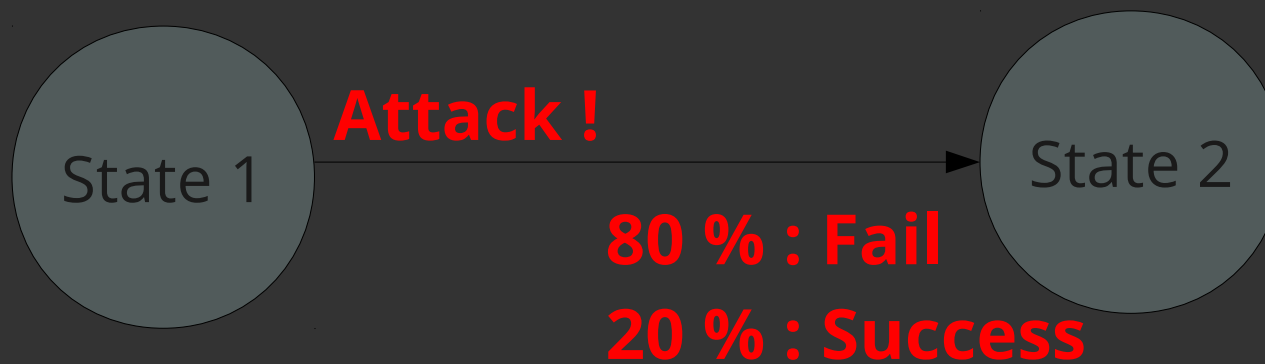


But answers depends on the environment

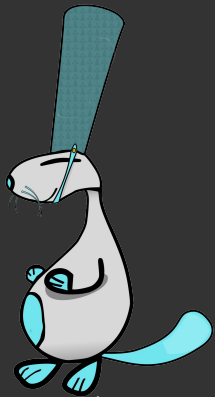
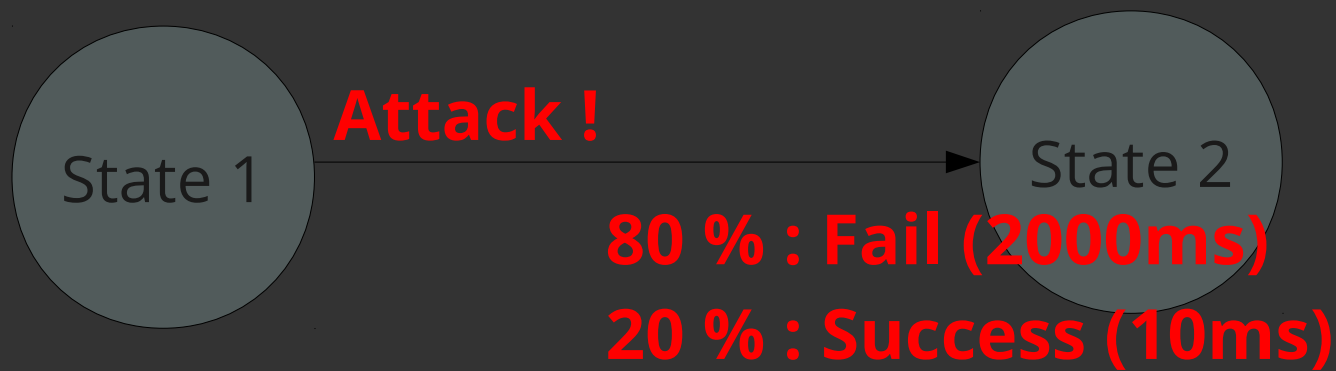


Our model (SMMDT)

→ Add probabilities on output messages

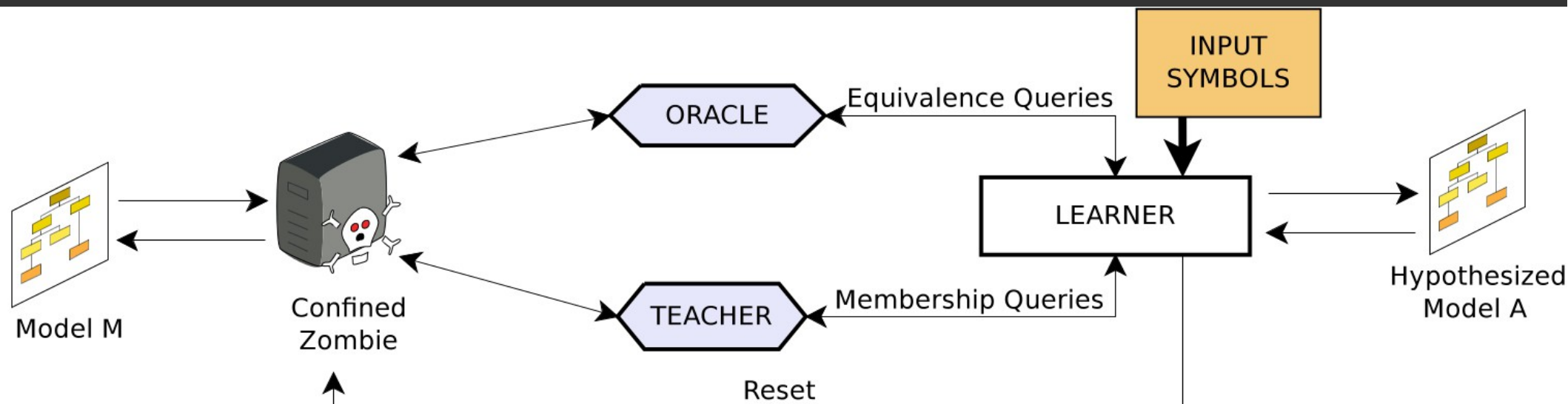


Our model (SMMDT)
→ Add the « reaction time »



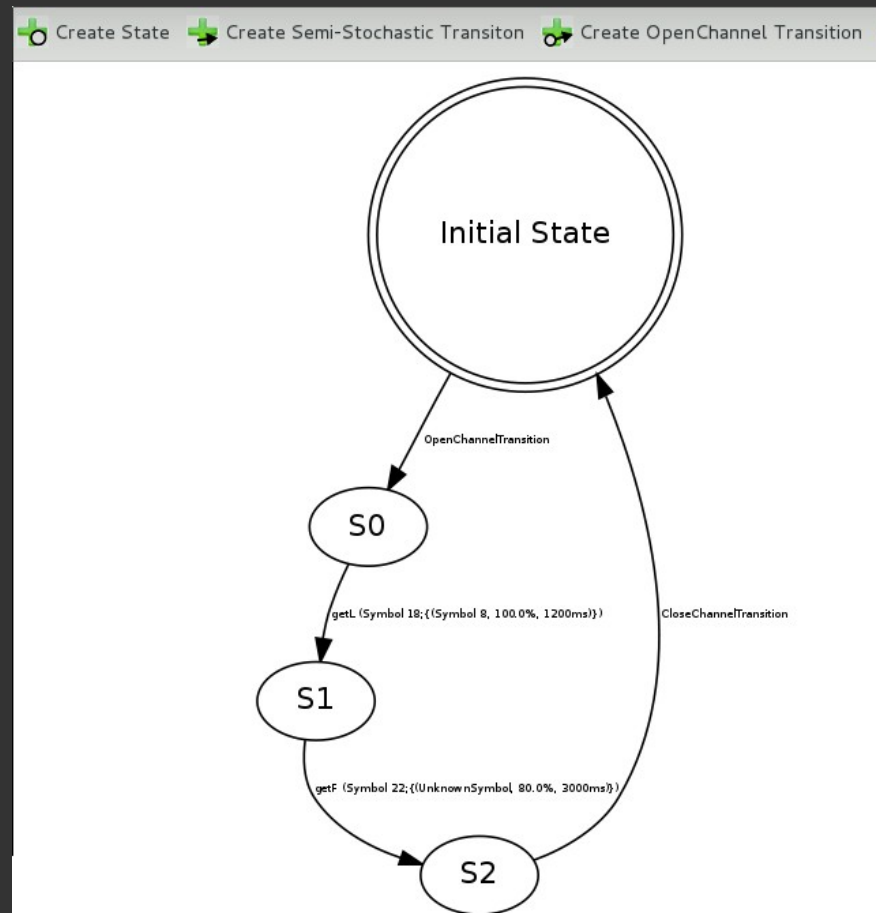
Active Grammatical Inference Process

→ **Angluin L*a Algorithm**

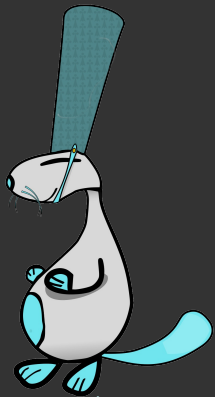


Active Grammatical Inference Process

→ **Angluin L*a Algorithm**

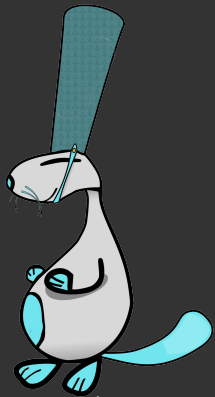


Recaps on ZeroAccess protocol



Recaps on ZA protocol

- At least 2 versions
- Multiple P2P management messages
 - « GetL », « RetL », « GetF », ...
 - Share common format
- **Low-encryption**
- UDP & TCP connections
 - UDP for messages (port 16464)
 - TCP for data
 - Hard coded Bootstrap Peers
 - Ex : 92.47.102.2, (...), 216.211.181.226

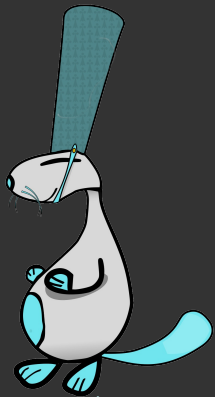




Generating traffic...

Netzob can generate traffic that:

- Follows the inferred message format
- Respects the state machine



Emulation of different kind of actors and flows

- Client ↔ Real server implementation
- Server ↔ Real client implementation
- Both client(s) and server

Create Network Actor

Create a Network Actor

General Informations

ID: d544669c-b30e-46b8-a67c-4d8b15b09f

Name: zeroAccessBot

☒ Initiator

Network Configuration

Type: CLIENT

Predefined Values

L4_PROTOCOL: UDP

BIND_IP: 192.168.42.41

BIND_PORT:

TARGET_IP: 115.22.87.69

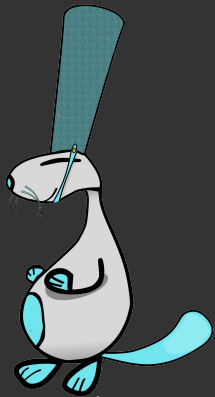
TARGET_PORT: 16464

Annuler  Appliquer 

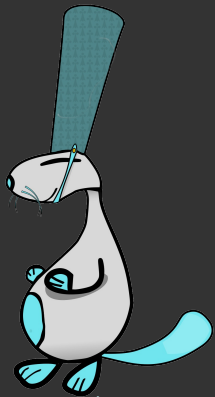
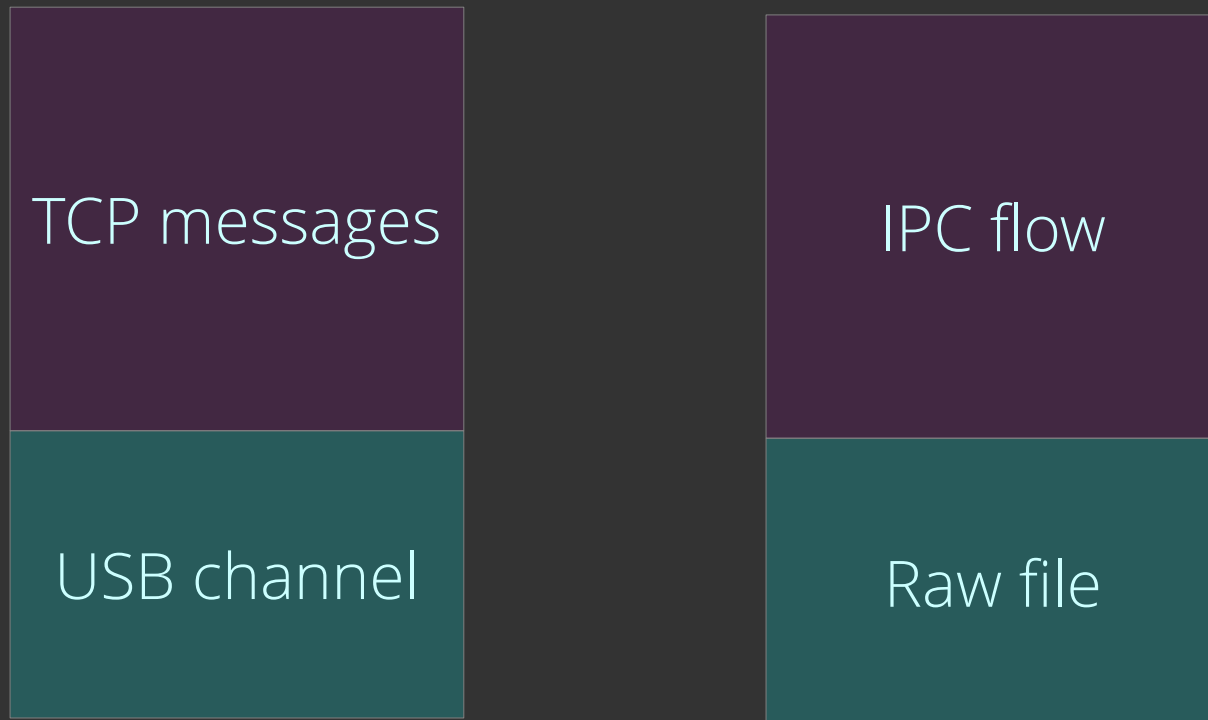
Distinction between

- Client / server
- Initiator / responder of the opening channel

*Ex : TLS with TCP session initiated
by the server*

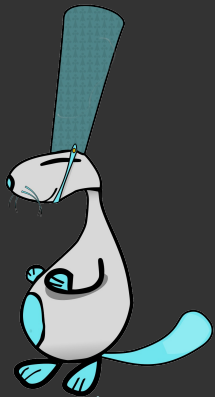


Abstraction from the communication channel



Memory mechanism

- Some received values are **memorized**...
- ...and **reinjected** in future messages
- Also handles contextual values (IP, time, etc.)



Abstraction and contextualization principles

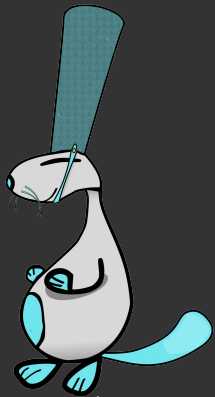
Input device

Input
flow

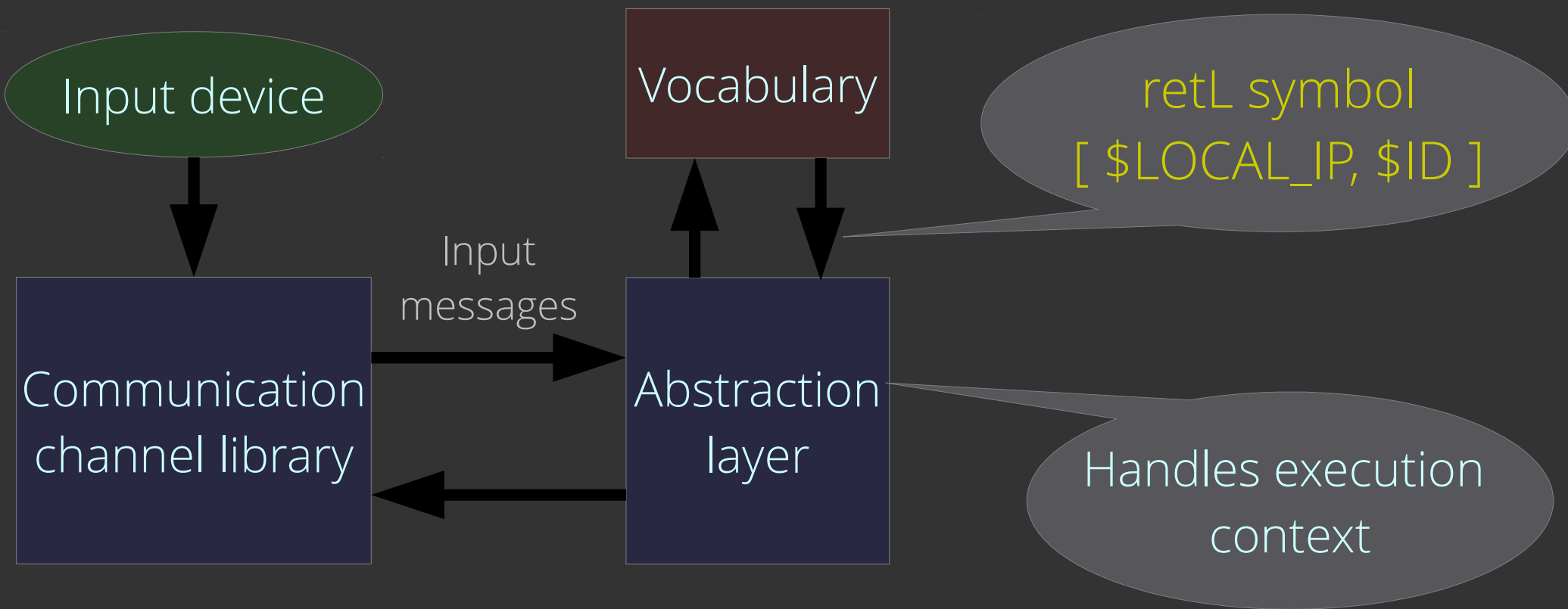


Communication
channel library

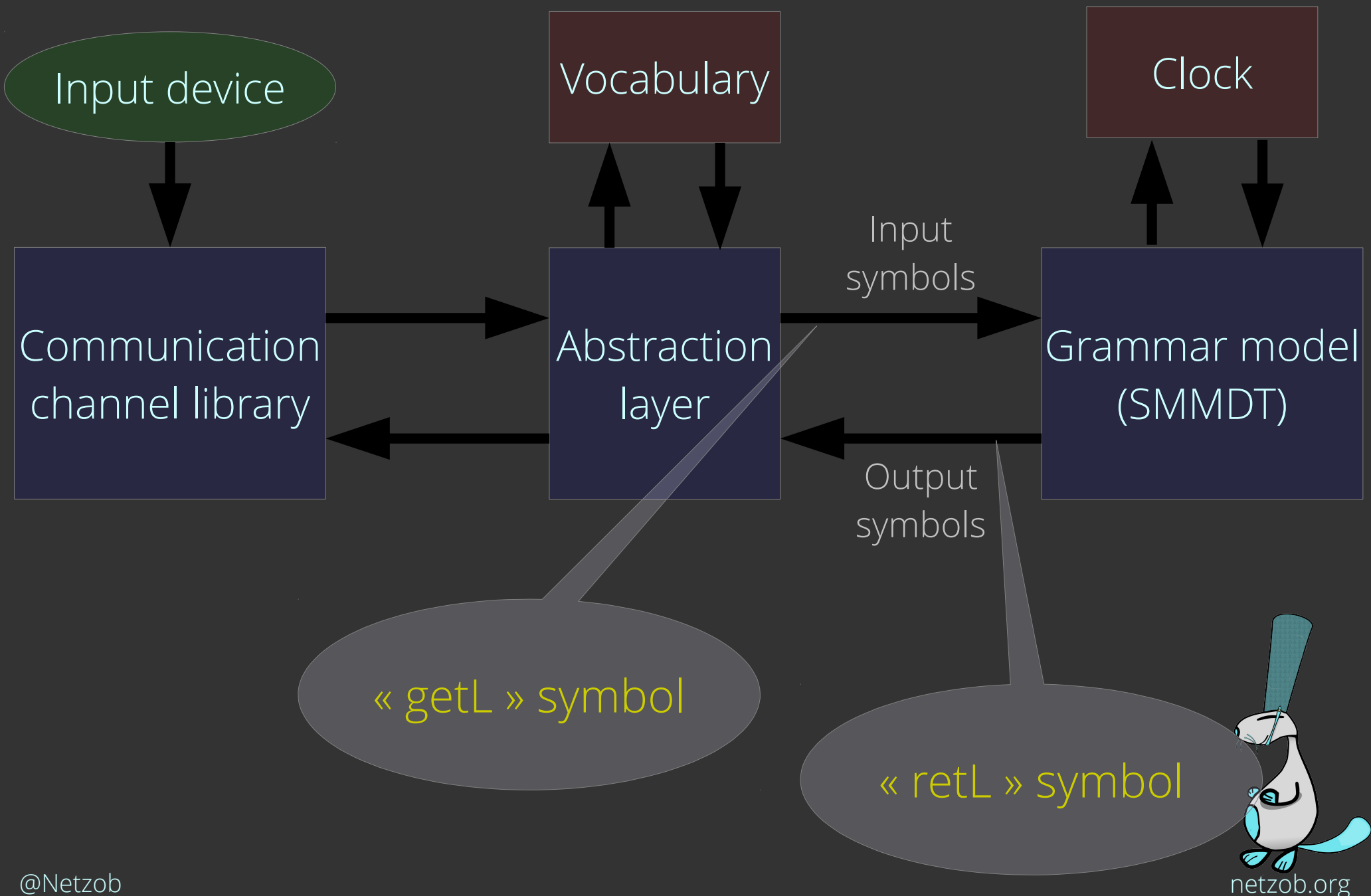
« 70dde8fc000000003 »



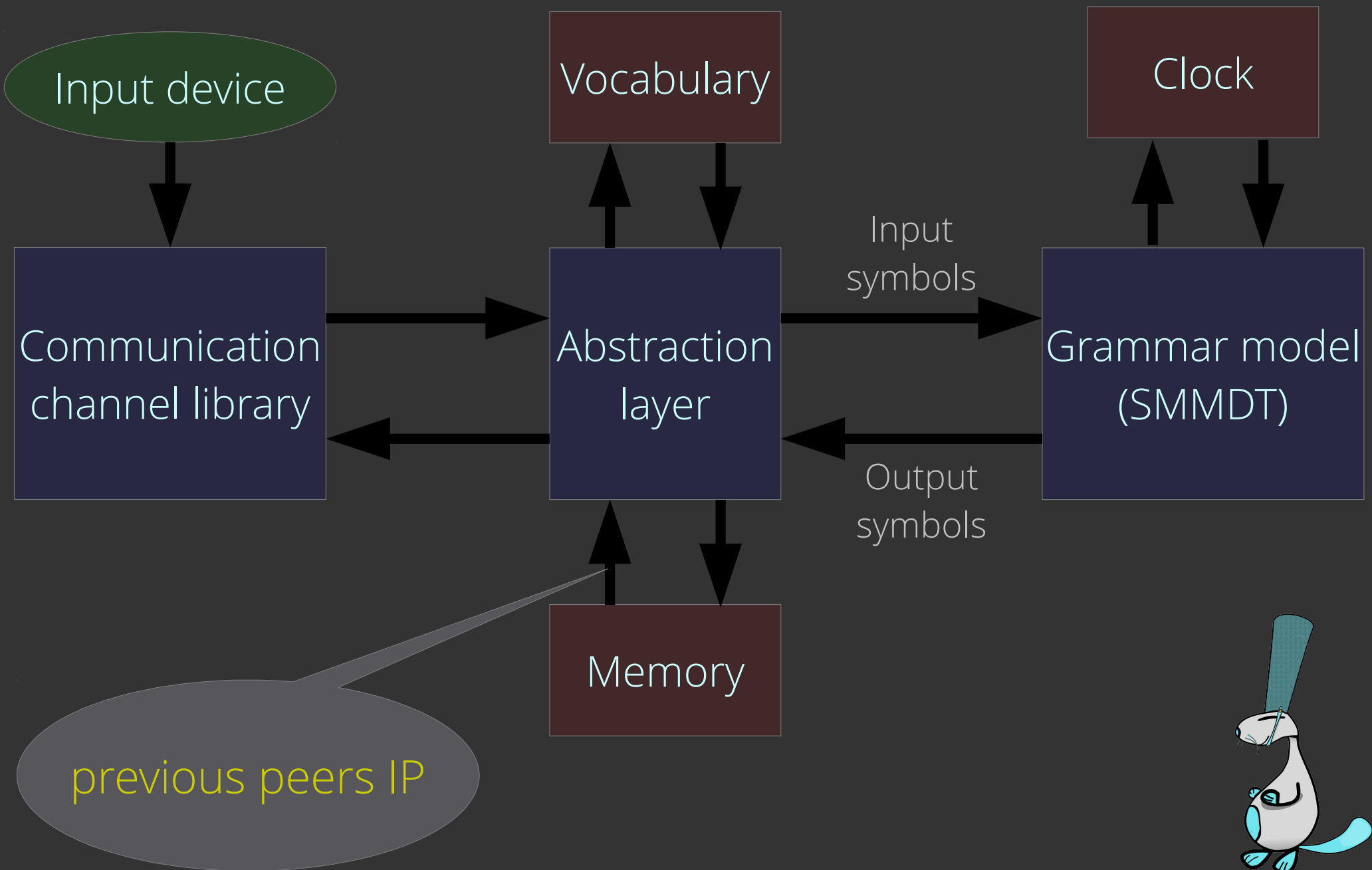
Abstraction and contextualization principles



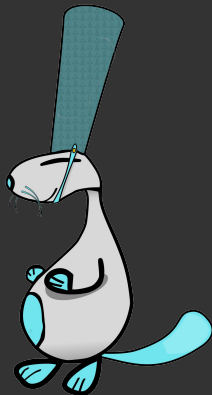
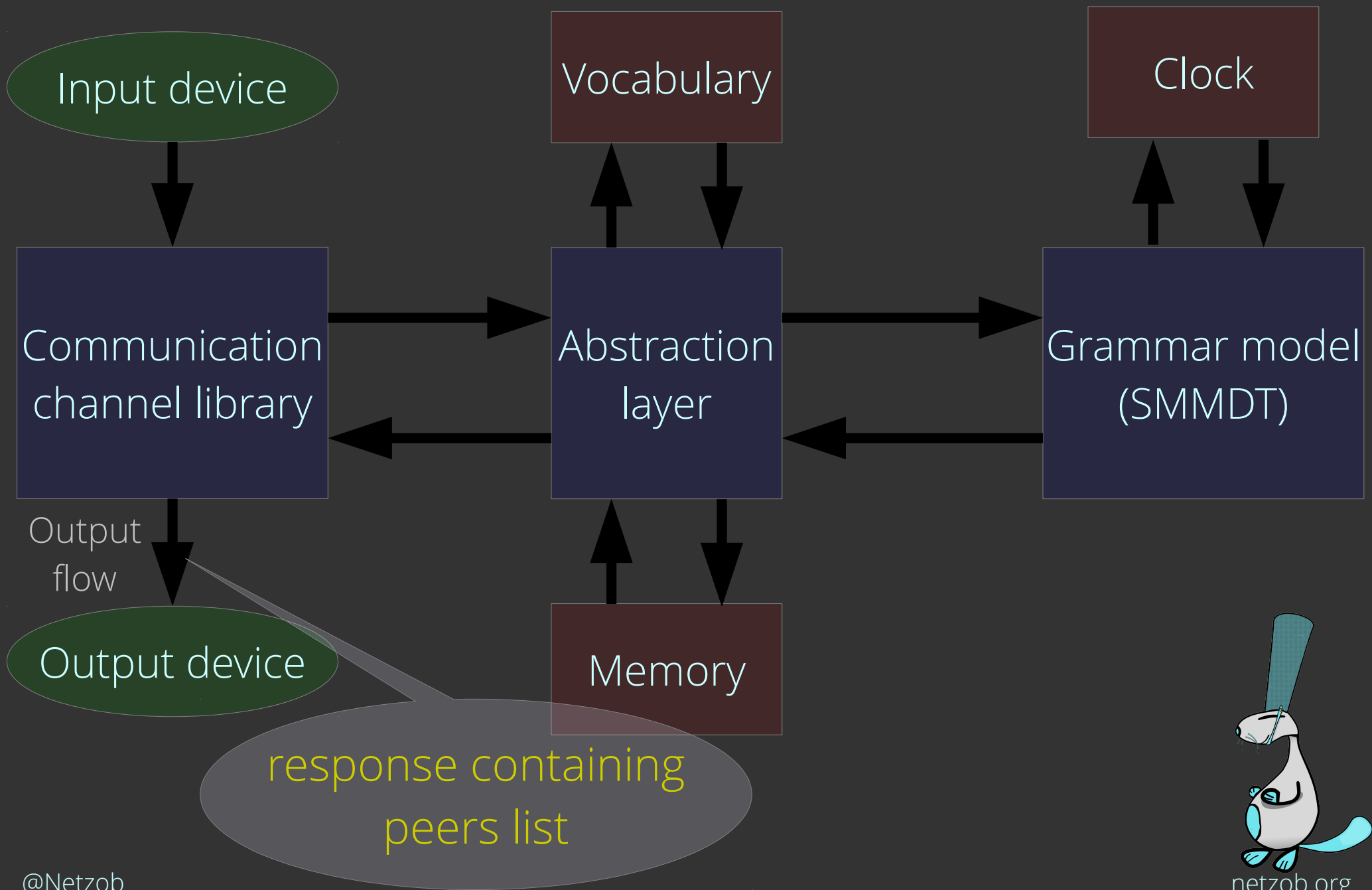
Abstraction and contextualization principles



Abstraction and contextualization principles



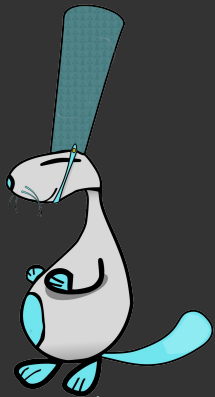
Abstraction and contextualization principles



Use cases of traffic simulation

Use case 1: simulation of a P2P botnet

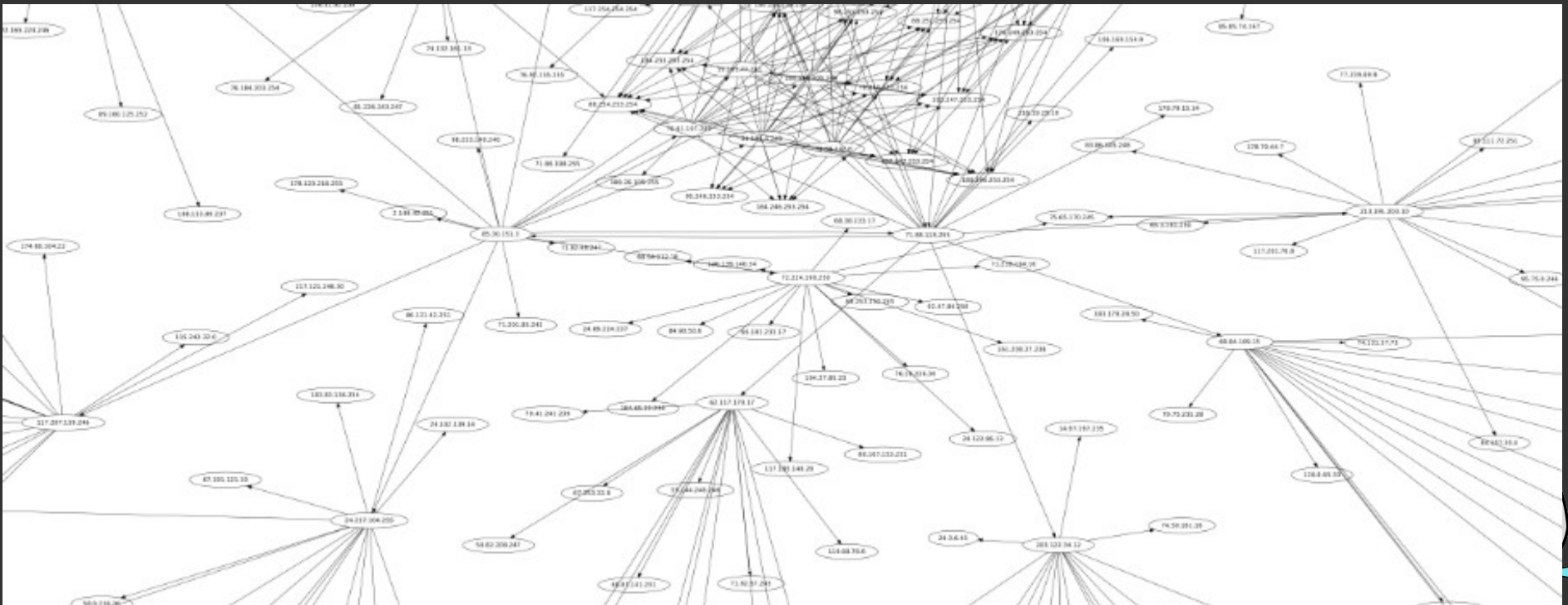
- Analyzing botnet scaling
- Studying botnet behavior at the network level
- Testing 3rd party products (IDS, ...)



Use cases of traffic simulation

Use case 2: retrieve the P2P zombie directory

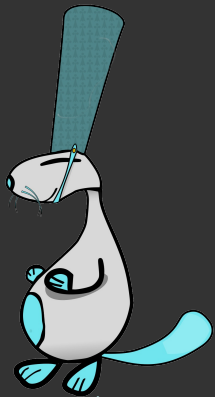
- Zero Access P2P map visualization
- Mapping the peers neighbours relations



Future improvements
of Netzob...

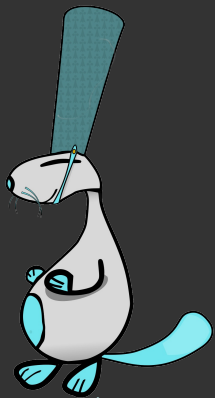
Integrated smart fuzzing, by leveraging the simulator engine

→ Allows to fuzz undocumented and propriatery protocols



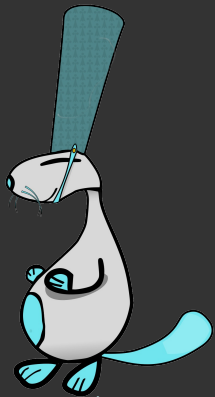
Integrated smart fuzzing, by leveraging the simulator engine

→ Allows to fuzz undocumented and propriatery protocols



Support of more communication channels

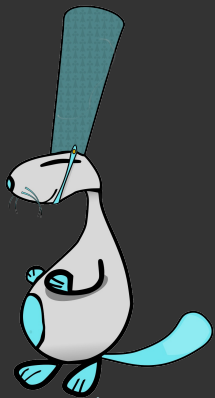
- USB
- IOCTL
- API (ssl_read, ssl_write, etc.)



Export protocol model in more 3rd party products

- Wireshark
- Scapy
- Peach Fuzzer

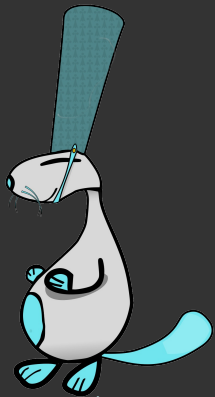
Allows protocol dissection with established tools



Export protocol model in more 3rd party products

- Wireshark
- Scapy
- Peach Fuzzer

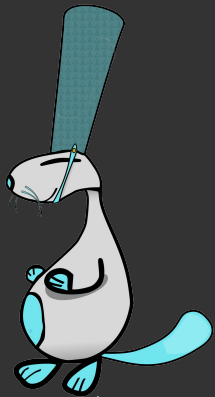
Allows fuzzing of unknown protocols with well-known tools



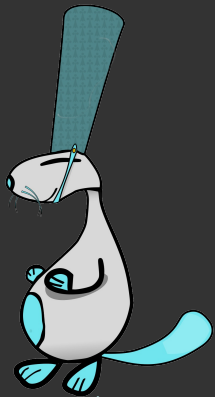
A black and white photograph of a long, empty, modern subway station. The floor is made of a grid of light-colored tiles, and the ceiling is dark with several long, horizontal fluorescent light fixtures. The walls are composed of large, rectangular panels. The perspective is from one end of the station, looking down its length. A small, dark object, possibly a pigeon, is visible on the floor in the lower right foreground.

Conclusion...

- ▶ Protocol RE automation domain is quite active at the academic level
- ▶ But **no real tool available...**
- ▶ Netzob tries to fill this lack by
 - ▶ Supporting academic researches
 - ▶ Being **usable in operational context**



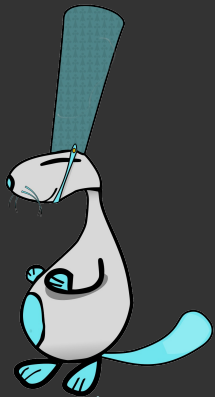
- ▶ Main current **sponsors**: AMOSSYS & Supélec
- ▶ **Open** to all kind of **contributions**
 - ▶ Feedback
 - ▶ Bug fix
 - ▶ Feature proposal / implementation
 - ▶ Translation
 - ▶ ...



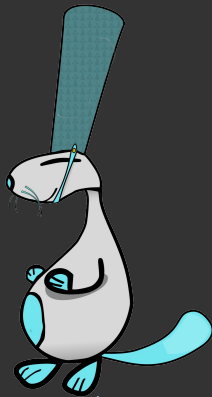
Netzob 0.4

« JumpingRhino »

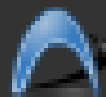
« It's alive ! It's alive ! »
Henry Frankenstein



- Towards a much more stable tool
- « Feature Release »
 - New User-Friendly Graphical Interface (GTK-3)
 - Upgrade the Vocabulary Inference process
 - Per-Layer RE Strategy
 - Add Transformation Functions ...
 - Extend Netzob with your own plugins
 - Upgraded Importers
 - Per-Layer Network importer
 - Ospy Importer ...



- Towards a much more stable tool
- « Feature Release »
 - New User-Friendly Graphical Interface (GTK-3)
 - Upgrade the Vocabulary Inference process
 - Per-Layer RE Strategy
 - Add Transformation Functions ...
 - Extend Netzob with your own plugins
 - Upgraded Importers
 - Per-Layer Network importer
 - Ospy Importer ...



Please download it and tell us
what you think of it !



Thanks for you attention !
Any questions ?

Please complete the Speaker Feedback Surveys.

www.netzob.org @netzob

Image licences

<http://www.flickr.com/photos/arne-halvorsen/3346841686/in/pool-1093738@N24>

CC-BY-NC / aha42 | tehaha

<http://www.flickr.com/photos/torek/3280152297/sizes/l/in/pool-1093738@N24/>

CC-BY-ND ar kirainet

http://www.flickr.com/photos/jdawg/295956572/in/pool-security_theater/

CC-BY-NC r lawgeek

<http://www.flickr.com/photos/tjblackwell/7324060440/sizes/l/in/pool-36004471@N00/>

CC-BY-NC r tj.blackwell

<http://www.flickr.com/photos/sterlingely/1418364/sizes/z/in/pool-865293@N20/>

CC-BY-NC-SA DogFromSPACE

<http://www.flickr.com/photos/massalim/8110616773/in/pool-83823859@N00/>

CC-BY-SA И. Максим

<http://www.flickr.com/photos/davidjunyent/8170407965/sizes/l/in/pool-83823859@N00/>

CC-BY-NC-ND davidjunyent

<http://www.flickr.com/photos/omarparada/8165303605/sizes/l/in/pool-83823859@N00/>

CC-BY-NC-ND Omar Parada

<http://www.flickr.com/photos/tonirodrigo/8114182589/sizes/l/in/pool-83823859@N00/>

CC-BY ar Toni Rodrigo

<http://opte.org/maps/>

CC-BY-NC-SA The optet project

<http://www.flickr.com/photos/niznoz/63732753/lightbox/>

CC BY-NC-SA 2.0 by niznoz

