Botnets Behavioral Patterns in the Network

Garcia Sebastian

@eldracote

Hack.Lu 2014

CTU University, Czech Republic. UNICEN University, Argentina.

October 23, 2014

How are we detecting malware and botnets?

- Analyze the binary files.
- Analyze the network traffic.

Analyze the binary files

- Static, Dynamic or Hybrid.
- What the malware is capable of doing. Even if it is not doing it, or can't do it now.
- How dangerous it is, how complex.
- Which techniques it uses.
- Behavior inside the host.
- Intentions through the capabilities.

Analyze the network traffic

- Static or Behavioral.
- The actions and how they change.
  - The actions of all the binaries and modules together.
- Real-time updates of binaries and C&C servers.
- You can see the intentions through the actions.

- People doing dynamic analysis of the binary files to read the network data *may* have this information.

How are we analyzing the Network Traffic?

From 39 products/companies in the market

- 47% use fingerprints or rules.
- 34% use reputation (TA).
- 50% use Anomaly Detection.
- Only 2 Machine Learning algorithms where not AD.

What is working?

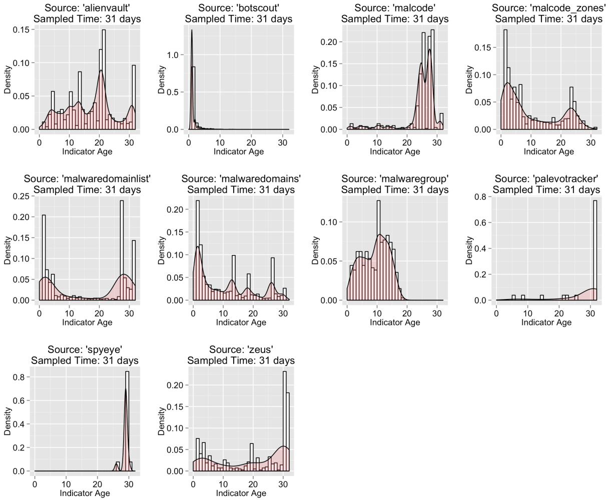
- Fingerprints are fast. Some are dynamic (e.g. Port Scan)
- Fingerprints have few False Positives and can be tuned for your network.
- Reputation if fast. Don't need to be tuned so much.
- Anomaly Detection... may work for very specific contexts.

What is not working?

- Fingerprints may take days to be created. Most attacks are not covered.
- Reputation is case-by-case. Maliciousness is hard to assess. A lot of effort, rules may be short lived.
- Anomaly Detection needs to build the normality of each network and adapt. Also, anomaly != maliciousness.

What is not working?

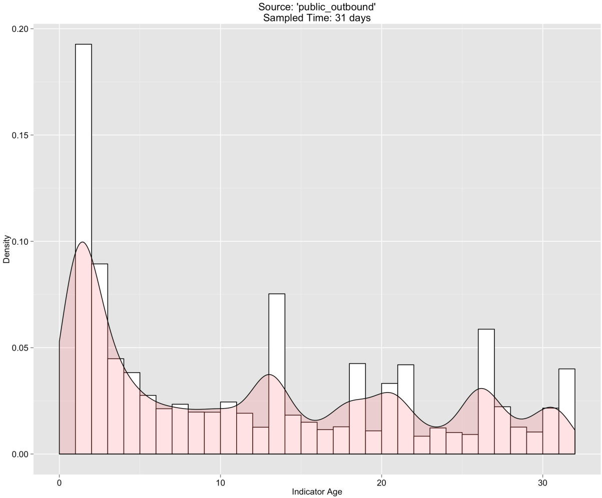
How long does an indicator sit in a Threat Intel feed?



Thanks Alex Pinto for the work and image! @alexcpsec.
www.mlsecproject.org

What is not working?

How long does an indicator sit in a Threat Intel feed?



Thanks Alex Pinto for the work and image! @alexcpsec.
www.mlsecproject.org

What is not working in Machine Learning?

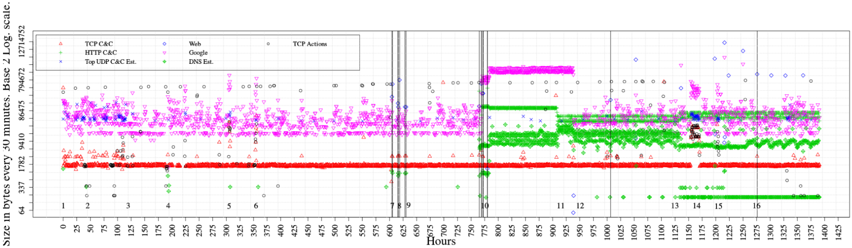
- Lack of complete description of the algorithms.
- Lack of good, common and labeled datasets.
- Lack of good evaluations in real environments.
- Lack of good comparisons with other methods.
- Results highly depend on the dataset.
- Results highly depend on the metrics!
- Generalization is very difficult.
- "There is no algorithm that can perfectly detect all possible virus" (Fred Cohen, "Computer Viruses: Theory and Experiments", Computers and Security 6 (1987)).

A different approach: Network Behaviors

- Instead of anomalies, it tries to model how does a specific traffic behaves.
- Behavior means to analyze features over time.
- But what should be modeled?
  - Networks?
  - Hosts?
  - Servers?
  - A bot?
  - A botnet?

The complexity of network traffic is high

One bot, 57 days. 3 C&C protocols simultaneously (UDP, TCP and HTTP).



A long-term analysis show the decisions by the botmaster.

Our Proposal

To deal with the complexity by modeling and finding the behavior of individual connections.

But what is a connection?

All the packets related with certain type of action.

- The traffic to a DNS server (Not all the DNS traffic).
- The access to https://www.google.com .
- The SPAM sent to a specific SMTP server.
- The traffic to a C&C service (server and port).
- Etc.

- How can we capture these?

The need for aggregation: 4-tuples

If a bot connects every 1 day to a TCP C&C server...

- TCP-style connections.
  - One TCP connection is not enough.
- At least one NetFlow every day.
  - One NetFlow does not capture everything.
- To get all the connections we need to aggregate NetFlows.
- The aggregation structure is called 4-tuple. IT simply aggregates NetFlows by ignoring the source port:
  - Source IP, destination IP, destination port and protocol.

The creation of our state-based behavioral model

"All models are wrong, but some are useful."

- We analyze the behavior of each 4-tuple by extracting 3 features of each NetFlow.
- Each NetFlow is assigned a state based on these 3 features.

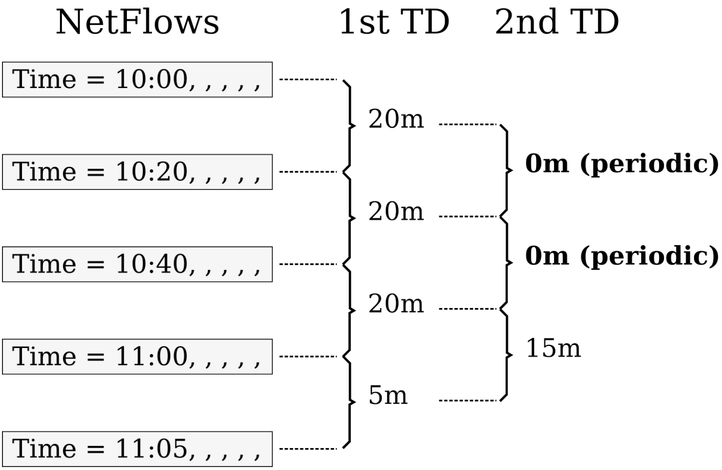
Features of each State. Keep it Simple.

Based on the analysis of long-term C&C channels...

- The size of the flow.
- The duration of the flow.
- The periodicity of the flow.

But how is periodicity defined?

Periodicity: 2nd Order Time Difference (TD)



This definition of periodicity allow us accurately analyze connections.

Behavioral Model: State Assignment to NetFlows

- The range of values for each feature is separated with 2 thresholds.
- Each NetFlow can be assigned one of 36 states.
- The special letter 0 is used for timeout.

	Small Size			Medium Size			Big Size		
Duration	Short	Medium	Long	Short	Medium	Long	Short	Medium	Long
Not enough data	1	2	3	4	5	6	7	8	9
Strongly periodic	a	b	c	d	e	f	g	h	i
Weakly periodic	A	B	C	D	E	F	G	H	I
Not periodic	r	s	t	u	v	w	x	y	z

Behavioral Model: Chain of States

- Each 4-tuple receives multiple NetFlows.
- Each NetFlow is assigned one state (one letter).
- The 4-tuple has a chain of states that models its behavior over time.

For example the 4-tuple *147.32.84.165-212.117.171.138-65500-tcp* has the following chain of states:

> *96iIIiFfiiIiiIIIfiIIIiiiiiiiiIfIiIIIiiiiIFiFIiwzwwzzIIiF0wzwzzfi0www wwzFzwFw0wwwwfiiw0wwzwww0wwwwwwfwww0wwzwiww wziwzwF0wwwfwwwwwwwwwzzwzziifiiiiifdffwwz0wzwiidfF IFFdiDFIIwzziiiiwwzfwwweiFFwwFFwwFEfi0wwwwFf(...)*

This connection is a TCP-based plain text botnet C&C channel.

Visualization of Behavior. 1st Botnet Connections

An example of a botnet with DNS/TCP access for DGA, HTTP, and HTTPS.
An example of an HTTP C&C channel.

Visualization of Behavior: 2nd Normal Connections

Normal HTTP and DNS.

Summary of the visualizations

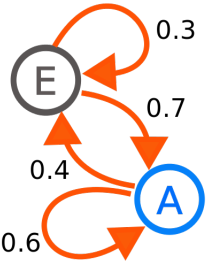
- It is important to be able to see and verify the behaviors. Helps evaluating the detection later.
- No connection has a perfect frequency periodicity.
- The most periodic connections are automatic by the OS by retrying.
- More important than the states are the transitions between states.

What can be done whit this? Our Botnet Detection Model

- Based on the behaviors we created a detection model that:
  1. Training phase: Trains a Markov Chain from the known and labeled behaviors.
  2. Testing phase: Generalizes the trained Markov Chains to detect similarities in unknown traffic.

Botnet Detection Model: Training Phase

- Created a labeled dataset.
  - Manually verified.
  - Botnet, Normal and Background labels.
  - 600GB of data.
  - 1,471 different unique labels (to, from).
  - Publicly available. (NetFlows all. Pcap only botnet)
- Use a Markov Chain to represent the probabilities of the transitions on each chain of states.



	a	b	c
a	0.1	0.6	0.3
b	0.25	0.05	0.7
c	0.7	0.3	0

Botnet Detection Model: Training Phase

The training model that we store includes:

- The Markov Chain Matrix
- The probability of generating the original chain of states that generated the matrix (POriginal).

Botnet Detection Model: Testing phase

Use the stored and trained models to detect similar behavior.

- For each 4-tuple in the unknown traffic:
  - Generate the chain of states of the unknown 4-tuple (letters).
  - For each previously trained model:
    - Compute the probability that the current model generated the unknown chain of states (PUnknown).
    - Compute the difference between POriginal and PUnknown.
    - If this difference is larger than a certain threshold, discard the model.
    - If not, retain this model as a candidate.
  - Select the candidate model with the smallest difference.
  - Assign the labels to the NetFlows.

Botnet Detection Model: Results

- We ran the algorithm in the labeled dataset (separated in training/cross-validation/testing).
- You need labeled data to obtain metrics.
- Results so far:
  - Average F-Measure: 78% (Best 93%)
  - Average FPR: 10% (Best 0.2%)
- Which were the errors? Some experiments did not have aa good trained model that represented the testing botnet traffic.

Comparison with other methods

- The detection model was compared with other 3 detection methods.
- Using the same dataset and error metrics.
- CAMNEP system, BotHunter system and BClus system.

Example Results

Name	tTP	tTN	tFP	tFN	TPR	TNR	FPR	FNR	Prec	Acc	ErrR	FM1
CCD	**87.6**	**254**	**14**	**0**	**1**	**0.94**	**0.05**	**0**	**0.86**	**0.96**	**0.03**	**0.92**
AllPo	65.5	0	69	0	1	0	1	0	0.4	0.4	0.5	0.65
BClus	**30.2**	**41.3**	**27.6**	**35.3**	**0.4**	**0.5**	**0.4**	**0.5**	**0.5**	**0.5**	**0.4**	**0.48**
Fs1	7.8	66.4	2.5	57.5	0.1	0.9	<.0	0.8	0.7	0.5	0.4	0.20
Fs1.5	6.3	67.2	1.7	59.1	<	0.9	<	0.9	0.7	0.5	0.4	0.17
Fd1	6.8	54.2	14.6	58.6	0.1	0.7	0.2	0.8	0.3	0.4	0.5	0.15
Fs2	4	67.6	1.3	61.4	<	0.9	<	0.9	0.7	0.5	0.4	0.11
Fd1.5	4.6	57.5	11.4	60.8	<	0.8	0.1	0.9	0.2	0.4	0.5	0.11
Fd2	2.2	59.8	9.1	63.2	<	0.8	0.1	0.9	0.1	0.4	0.5	0.05
Mi1	2.3	52.3	16.6	63.1	<	0.7	0.2	0.9	0.	0.4	0.5	0.05
X1	1.7	68.6	0.3	63.6	<	0.9	<	0.9	0.8	0.5	0.4	0.05
X1.5	1.5	68.6	0.3	63.9	<	0.9	<	0.9	0.8	0.5	0.4	0.04
BH	**1.59**	**73.8**	**0.18**	**109**	**0.01**	**0.9**	<	**0.9**	**0.8**	**0.4**	**0.5**	**0.02**
Mi1.5	1	56.9	12	64.4	<	0.8	0.1	0.9	<	0.4	0.5	0.02
Mi2	0.6	63.1	5.8	64.8	<	0.9	<	0.9	<	0.4	0.5	0.01
Le1	0.2	68.1	0.8	65.2	<	0.9	0.01	0.9	0.2	0.5	0.4	0.007
Ko1	0.1	68.7	0.1	65.3	<	0.9	<	0.9	0.4	0.5	0.4	0.004
Ko1.5	0.08	68.9	0.02	65.3	<	1	0	0.9	0.7	0.5	0.4	0.002
CA1	**0.005**	**68.7**	**0.2**	**65.4**	**0**	**0.9**	<	**1**	<	**0.5**	**0.4**	**<0**
T1.5	0.005	68.9	0	65.4	0	1	0	1	1	0.5	0.4	<0
T1	0.005	68.9	0	65.4	0	1	0	1	1	0.5	0.4	<0

Future Work: Behavioral IPS

- Use verified behavioral models in real time traffic.
- The actions are taken depending on the matching model and independently of the IP addresses, ports, domains or payloads:
  - Block C&C behaviors.
  - Block DoS attacks.
  - Block certain type of SPAM.
  - Block malicious P2P, while allowing normal P2P.
  - Block brute-force attacks while allowing normal logins.
- Behavioral models can be as specific as a signature. . They can generalize to similar behaviors.

Conclusions

- How many flows do we need?
- Attacking with one flow?
- Future research
  - The analysis of several behaviors together may improve the detection of the IP.
  - Verify the classification of more labels.
- Behavioral models captures the dynamism.
- Behavior is key to long-run detection.

Thanks!

Thanks for staying!
@eldracote
eldraco@gmail.com
Malware Capture Facility Project: http://mcfp.weebly.com/

Be careful with the metrics...

- Metrics highly depend on the dataset and network.
- The utility of a model depends on how it is used.
- Metrics highly depend on how errors are considered. We use time windows, IP addresses and an aging function.
- We consider a TP when an IP address is correctly detected as botnet at least once in the time window.
- We consider a TN when an IP address is correctly detected as Normal during the whole time window.

Resources

- An empirical comparison of botnet detection methods. S. García, M. Grill, J. Stiborek, A. Zunino. Computers & Security Journal. Elsevier.