DETECTING RANDOM STRINGS; A LANGUAGE BASED APPROACH

Mahdi Namazifar, PhD

Cisco Talos

PROBLEM DEFINITION

- Given an arbitrary string, decide whether the string is a ***random sequence of characters***
- *Disclaimer 1:* This work does not address strings that are random sequences of dictionary words
- *Disclaimer 2:* The current parameters of the code are tuned for strings with length 8 or more

MOTIVATION AND BACKGROUND

- Detecting domain names that are generated by Domain Generation Algorithms (DGA)

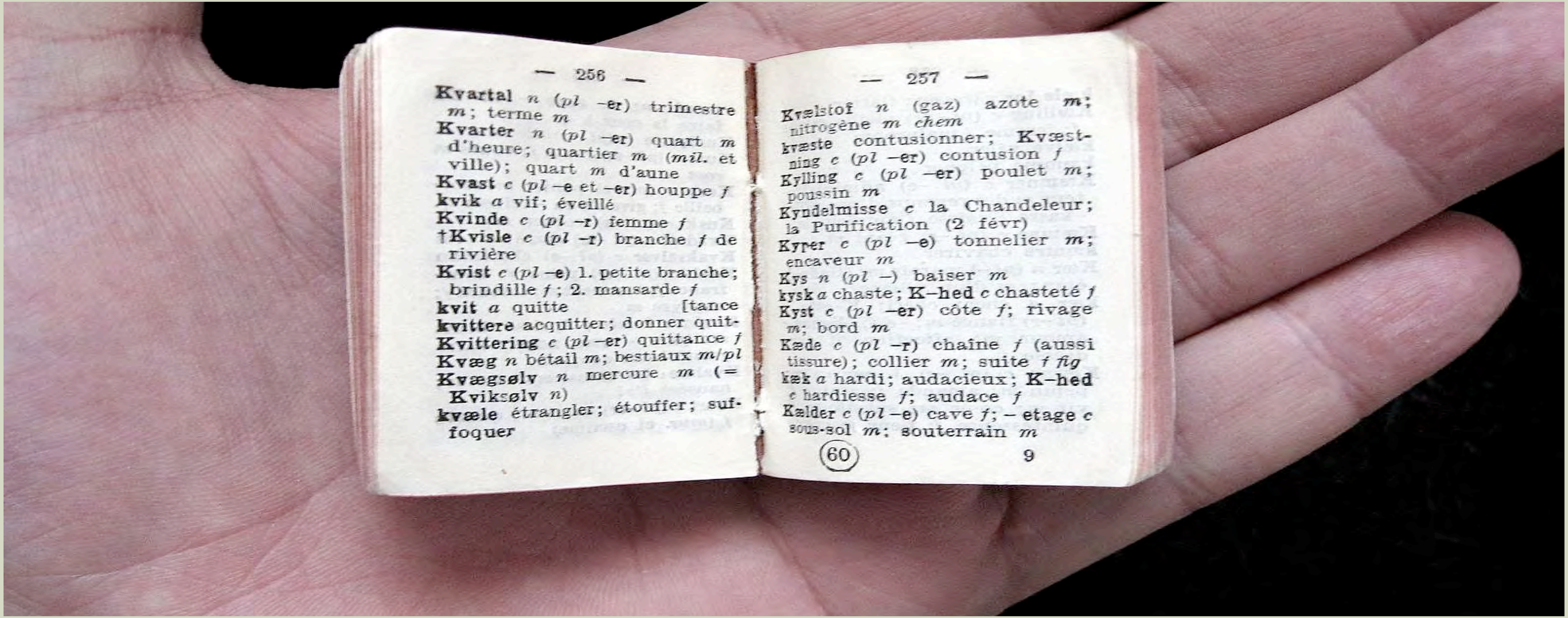
- Many have studied this problem:
  - Papers such as:
    - S. Yadav, A. Reddy, A.L.N. Reddy, and S. Ranjan, *"Detecting Algorithmically Generated Malicious Domain Names"*, IMC'10, November 1–3, 2010, Melbourne, Australia.
    - J. Raghurama, D.J. Millera, and G. Kesidis, *"Unsupervised, low latency anomaly detection of algorithmically generated domain names by generative probabilistic modeling"*, Journal of Advanced Research, Vol. 5, Issue 4, pp. 423–433.
    - …
  - Bayesian network approaches
  - Random Forrest classifiers
  - …

OUR APPROACH; THE BIG PICTURE

- Gather as many dictionaries as you can
- Look up substrings of a given string in the dictionaries
- Based on
  - number of dictionary hits
  - length of substrings that were in a dictionary
  - number of different languages needed to cover the substrings

  define a randomness score.
- Used the score to determine whether the string is random

“MEGA” DICTIONARY



— 256 —

Kvartal *n* (*pl* -er) trimestre *m*; terme *m*

Kvarter *n* (*pl* –er) quart *m* d'heure; quartier *m* (*mil.* et ville); quart *m* d'aune

Kvast *c* (*pl* –e et –er) houppe *f*

kvik *a* vif; éveillé

Kvinde *c* (*pl* –r) femme *f*

†**Kvisle** *c* (*pl* –r) branche *f* de rivière

Kvist *c* (*pl* –e) 1. petite branche; brindille *f*; 2. mansarde *f*

kvit *a* quitte

kvittere acquitter; donner quittance

Kvittering *c* (*pl* –er) quittance *f*

Kvæg *n* bétail *m*; bestiaux *m/pl*

Kvægsølv *n* mercure *m* (= **Kviksølv** *n*)

kvæle étrangler; étouffer; suffoquer

— 257 —

Kvælstof *n* (gaz) azote *m*; nitrogène *m* *chem*

kvæste contusionner; **Kvæstning** *c* (*pl* –er) contusion *f*

Kylling *c* (*pl* –er) poulet *m*; poussin *m*

Kyndelmisse *c* la Chandeleur; la Purification (2 févr)

Kyper *c* (*pl* –e) tonnelier *m*; encaveur *m*

Kys *n* (*pl* –) baiser *m*

kysk *a* chaste; **K–hed** *c* chasteté *f*

Kyst *c* (*pl* –er) côte *f*; rivage *m*; bord *m*

Kæde *c* (*pl* –r) chaîne *f* (aussi tissure); collier *m*; suite *f* *fig*

kæk *a* hardi; audacieux; **K–hed** *c* hardiesse *f*; audace *f*

Kælder *c* (*pl* –e) cave *f*; – **etage** *c* sous-sol *m*; souterrain *m*

(60) 9

“*MEGA*” DICTIONARY – LANGUAGES❖

Afrikaans
Akan
Albanian
Bulgarian
Catalan*
Chichewa
Croatian
Czech
Danish
Dutch
English*
Esperanto**
Estonian
Faroese
French*
Frisian
Gaeilge
Galician
German*
Greek
Hungarian
Indonesian
Interlingua**
Italian
Kinyarwanda
Kurdish
Latin
Latvian
Lithuanian
Malagasy
Malay
Mandarin
Māori
Norwegian*
Occitan
Polish
Portuguese*
Romanian
Russian*
Saraiki
Scottish Gaelic
Slovene
Southern Ndebele
Southern Sotho
Spanish*
Swahili
Swati
Swedish
Tagalog
Tetum
Tsonga
Tswana
Turkish
Ukrainian
Venda
Vietnamese
Welsh
Xhosa
Zulu

❖ Source: OpenOffice and others
* Different versions of the language
** Constructed language

“*MEGA*” DICTIONARY – OTHER

- US 1990 census data:
  - Female names
  - Male names
  - Surnames
- Dictionary of Scrabble words
- Alexa 1000 domain names
- Numbers
- Dictionary of texting acronyms
  - “yolo”, “wyd”, “ttyt”

SPECIAL TREATMENT

- Slugify to deal with accents, special characters, etc.

- Mandarin, Japanese, …
  - 狴犬
  - Pinyin: “geng3 quan3”
  - The following words are added to the dictionary:
    - “geng3quan3”
    - “gengquan”

- Russian and Ukrainian
  - Use “koi8-r” decoding
  - *“i”* and *“y”* are used interchangeably

- …

SAME WORD MULTIPLE DICTIONARIES

- The word “book” appears in multiple different dictionaries
  - English, Polish, Dutch
- Run Map-Reduce to find all the dictionaries that a word appears in
- As a result every entry of the “mega” dictionary looks like
  - “suis”, ['ad', 'nl', 'af', 'ms', 'ca', 'fr’]
  - Each element of the list is a 2-letter code indicating a dictionary
- Some special dictionaries:
  - ‘ee’: English dictionary with ~360K words (simple English)
  - ‘ad’: English dictionary (including Scrabble words) with over 1.5M words (elaborate English)

MEGA DICTIONARY

- A Python dictionary of str to list of str
  - “suis”: ['ad', 'nl', 'af', 'ms', 'ca', 'fr’]
- Lookup time complexity *O(1)* for average case
- Currently contains over 11.7M entries

LOOKING UP SUBSTRINGS

- Traversing the string
  - From left:
    - “mystring” → “mystring”
    - “mystring” → “ystring”
    - “mystring” → “string”
    - “mystring” → “tring”
    - “mystring” → “ring”
    - “mystring” → “ing”
  - From right:
    - “mystring” → “mystring”
    - “mystring” → “mystrin”
    - “mystring” → “mystri”
    - “mystring” → “mystr”
    - “mystring” → “myst”
    - “mystring” → “mys”

LOOKING UP SUBSTRINGS (SIMPLE ENGLISH)

- Traversing and looking up (simple English)
  - From left:

“goodtobethere”	“goodtobethere”	No
“goodtobethere”	“oodtobethere”	No
“goodtobethere”	“odtobethere”	No
“goodtobethere”	“dtobethere”	No
“goodtobethere”	“tobethere”	No
“goodtobethere”	“obethere”	No
“goodtobethere”	“bethere”	No
“goodtobethere”	“ethere”	Yes!
“goodtob”	“goodtob”	No
“goodtob”	“oodtob”	No
“goodtob”	“odtob”	No
“goodtob”	“dtob”	No
“goodtob”	“tob”	Yes!
“good”	“good”	Yes!

[“ethere”, “tob”, “good”]

LOOKING UP SUBSTRINGS (SIMPLE ENGLISH)

- Traversing and looking up (simple English)
  - From right:

“goodtobethere”	“goodtobethere”	No
“goodtobethere”	“goodtobether”	No
“goodtobethere”	“goodtobethe”	No
“goodtobethere”	“goodtobeth”	No
“goodtobethere”	“goodtobet”	No
“goodtobethere”	“goodtobe”	No
“goodtobethere”	“goodtob”	No
“goodtobethere”	“goodto”	No
“goodtobethere”	“goodt”	No
“goodtobethere”	“good”	Yes!

“tobethere”	“tobethere”	No
“tobethere”	“tobether”	No
“tobethere”	“tobethe”	No
“tobethere”	“tobeth”	No
“tobethere”	“tobet”	No
“tobethere”	“tobe”	Yes!

“there”	“there”	Yes!

[“good”, “tobe”, “there”]

PICKING BETWEEN TWO SETS

- [“ethere”, “tob”, “good”] · > min length: 3
- [“good”, “tobe”, “there”] · > min length: 4

[“good”, “tobe”, “there”]

LOOKING UP FOR MORE LANGUAGES

- floatingbarmalapascua.com
- Registered on: June 23, 2013
- Substrings found:
  - “floating”: ['de', 'ee', 'it', 'ad']
  - “barma”: ['sk', 'sq', 'gs', 'cs', 'pt']
  - “lapas”: ['gs', 'gl', 'oc', 'af', 'hi', 'lt']
  - “cua”: ['vi', 'en', 'id', 'gl', 'ca', 'gs', 'bg', 'sq']
- How to find minimal set of dictionaries that has non-empty intersections with all the dictionary lists above?

MINIMUM HITTING SET PROBLEM

- Collection C of subsets of a finite set S

- A hitting set for C, i.e., a subset $S' \subset S$ such that S' contains at least one element from each subset in C

- Find minimum cardinality hitting set, S'

- **Bad news:** MHS is NP hard
- **Good news:** our sets are small enough that we use a greedy algorithm

MINIMUM HITTING SET; GREEDY ALGORITHM

- From each subset, pick an element and put them together into a set
- Find all possible sets built this way
- Take the ones with minimum cardinality
- Disclaimer: there are more efficient algorithms for this problem, but this one is good enough for us
- Back to our example:
  - Substrings found:
    - “floating”: ['de', 'ee', 'it', 'ad']
    - “barma”: ['sk', 'sq', 'gs', 'cs', 'pt']
    - “lapas”: ['gs', 'gl', 'oc', 'af', 'hi', 'lt']
    - “cua”: ['vi', 'en', 'id', 'gl', 'ca', 'gs', 'bg', 'sq’]
  - Minimum hitting sets:

    ['de', 'gs'], ['ee', 'gs'], ['gs', 'it'], ['gs', 'ad']
  - At least 2 dictionaries are needed to cover the words

NON-RANDOMNESS SCORE

- Factors:
  - Minimum hitting set number
  - Length of the string
  - Sum of length of words found in the string
  - Number of words longer than 3 letter

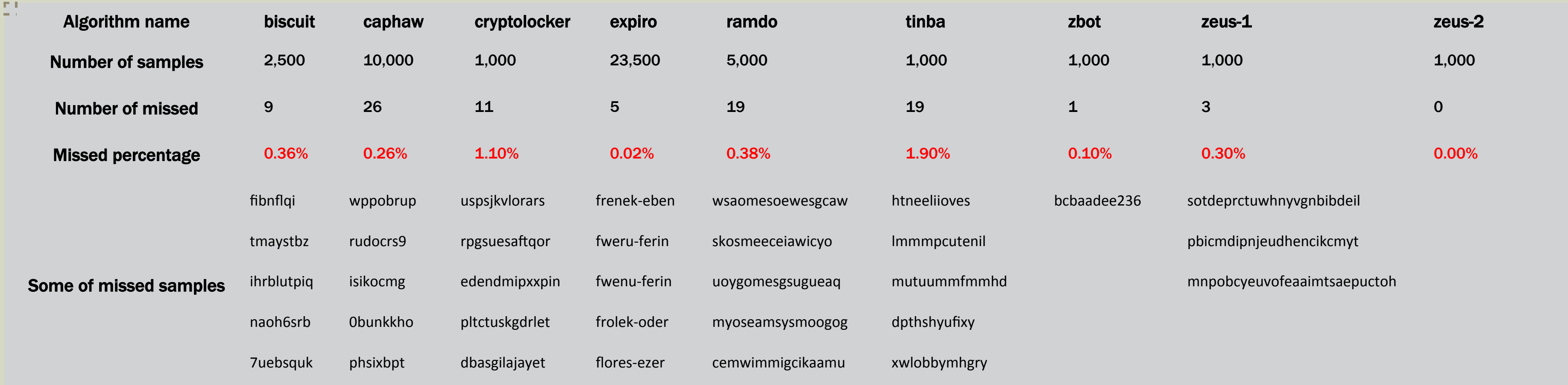
- These factors along with parameters that are tuned are used to give scores for:
  - Randomness with regards to a “simple” English dictionary
  - Randomness with regards to a “comprehensive” English dictionary
  - Randomness with regards to “all” languages

OTHER CONSIDERATIONS

- Sequence of alternating vowels and consonants.
  - Example: “symebitop”, “cusabifik”, “figih-avow”, …
- Is “_” or “-” present in the string?
  - These characters indicate some sort of separation that could be used
  - Example: “ugg-outlet-store-online”, “free-android-claims”
- Punycode:
  - xn--t8j0gd4151ac8betyjq5g
  - お金借りる主婦

RESULT

- **False negative:**
  - We use 9 Domain Generation Algorithms to generate random strings
  - We see how many of them are missed by our algorithm



Algorithm name	biscuit	caphaw	cryptolocker	expiro	ramdo	tinba	zbot	zeus-1	zeus-2
Number of samples	2,500	10,000	1,000	23,500	5,000	1,000	1,000	1,000	1,000
Number of missed	9	26	11	5	19	19	1	3	0
Missed percentage	0.36%	0.26%	1.10%	0.02%	0.38%	1.90%	0.10%	0.30%	0.00%
Some of missed samples	fibnflqi tmaystbz ihrblutpiq naoh6srb 7uebsquk	wppobrup rudocrs9 isikocmg 0bunkkho phsixbpt	uspsjkvlorars rpgsuesaftqor edendmipxxpin pltctuskgdrlet dbasgilajayet	frenek-eben fweru-ferin fwenu-ferin frolek-oder flores-ezer	wsaomesoewesgcaw skosmeeceiawicyo uoygomesgsugueaq myoseamsysmoogog cemwimmigcikaamu	htneeliioves lmmmpcutenil mutuummfmmhd dpthshyufixy xwlobbymhgry	bcbaadee236	sotdeprctuwhnyvgnbibdeil pbicmdipnjeudhencikcmyt mnpobcyeuvofeaaimtsaepuctoh	

RESULTS

- False positive:
  - Take Alexa 10,000 domains
  - Filter out strings shorter than 8 characters
  - Left with 5400 domain names.
  - I run them through my code
  - here are the ones that my code detected as random

lmebxwbsno	bezuzyteczna	thiruttuvcd	123sdfsdfsdfsd	lavoixdunord	3a6aayer
fmdwbsfxf0	plsdrct2	andhrajyothy	canlidizihd1	abckj123	muryouav
nguoiduatin	mazika2day	hosyusokuhou	przegladsportowy	follovvme	masqforo
fullvehdfilmizle	plsdrct1	addic7ed	1c-bitrix	anige-sokuhouvip	xxeronetxx
akb48matomemory	3djuegos	phununet	thqafawe3lom	donya-e-eqtesad	ikih0ofu
thaqafnafsak	srv2trking	vecteezy	turkcealtyazi	adstrckr	avmuryou
nsdfsfi1q8asdasdzz	iiasdomk1m9812m4z3	thiruttuvcd	esrvadspix	isif-life	ig84adp2

TOMORROW starts here.