PRACTICAL APPLICATION OF MACHINE LEARNING TO CYBER SECURITY

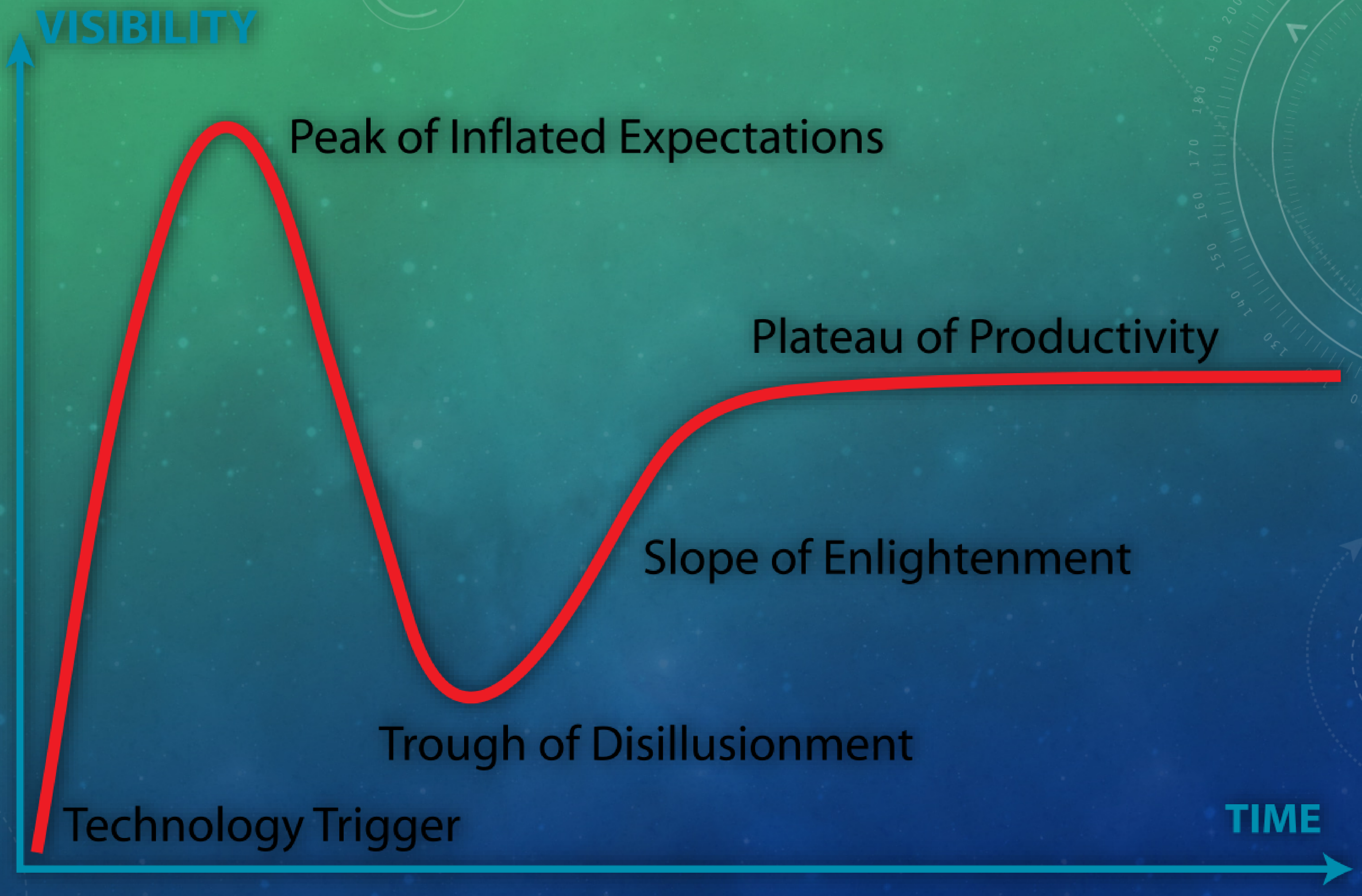
TIM CROTHERS

WHO AM I?

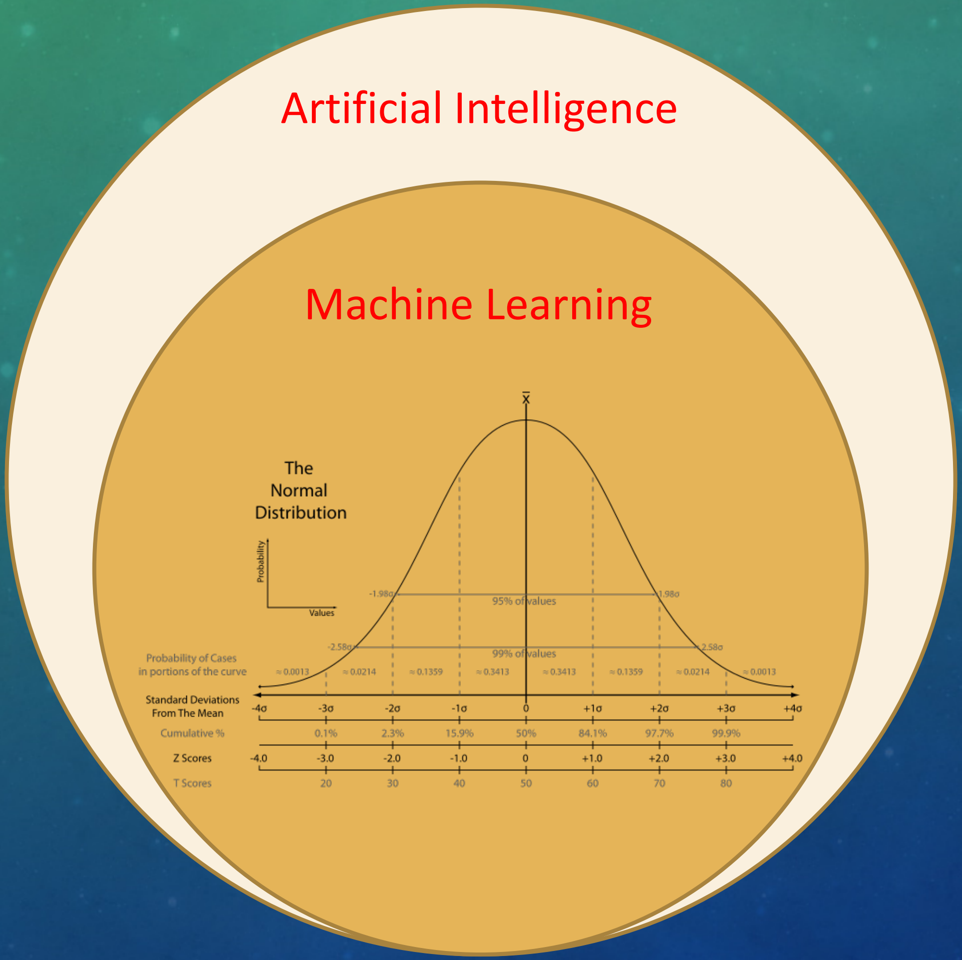
- >30 years in Information Technology and >20 years in Infosec
- Authored/Co-Authored 17 books to date
- Engineer and Maker
- Unabashed Math & Computer Science Geek
- Spent several years “on the ground at some of the largest breaches”

OPPORTUNITY

- Breaches continue to grow in number and severity year after year
- Severe shortage in Cyber Security Subject Matter Expertise
- Venture capital funds and research opportunities are readily available


VISIBILITY
Peak of Inflated Expectations
Plateau of Productivity
Slope of Enlightenment
Trough of Disillusionment
Technology Trigger
TIME

MACHINE LEARNING?



By Heds 1 at English Wikipedia - Transferred from en.wikipedia to Commons by Abdull., Public Domain, https://commons.wikimedia.org/w/index.php?curid=2799839


Machine Learning
Supervised
Unsupervised
Support Vector
Decision Trees
Neural Networks
Recurrent
Convolutional
Dense
Linear
Clustering
Associative Rule Learning
Dimensionality Reduction
Density Based Spacial
K-Means
Principal Component Analysis
Isomap

COMMON FAILURE #1

- Pure anomaly detection
  - Real world networks are messy
  - Real world systems are inconsistently configured
  - Real world vendor applications are usually abnormal
  - Real world hosts are all unique within a few minutes of the end user taking possession

COMMON FAILURE #2

- Trying to be all security things to all security people
  - Determining optimal parameters and features for a tightly scoped use case is pretty easy
  - As the width of the use case increases the difficulty increases exponentially

COMMON FAILURE #3

- Failing to leverage deep cyber security subject matter expertise
  - It’s hard to solve a problem you don’t understand well
  - Interesting != security problem
  - Security problem != something that will improve security
  - Success in a lab is much easier than success in a real world environment

COMMON FAILURE #4

- Failing to leverage deep ML subject matter expertise
  - Proper parameter and feature selection is critical
  - Proper algorithm selection is really important
  - Proper testing and refinement is critical

HOW DO YOU DO IT?

- Problem to solve
- Decide on approach
- Appropriate data
- Determine proper features
- Build your tool
- Test & tune
- Win!

KEYS TO SUCCESS

- Tightly scoped problem statement or use case
- Decide on approach
- Appropriate data
- Determine proper parameters and features
- Test & tune

TIGHTLY SCOPED PROBLEM

- Find the malicious activity in my DNS that my signature based detection isn't finding
- Find malicious PowerShell activity in Windows event logs that isn't being detected otherwise
- Find unknown malicious traffic posing as legitimate applications

DECIDE ON APPROACH

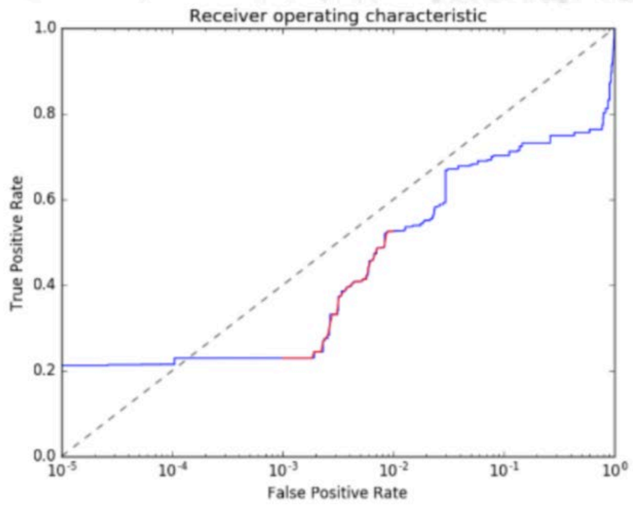
- Supervised
  - Generally best for solving specific problems
  - Needs ’labeled’ data
- Unsupervised
  - Essentially anomaly detection
  - Needs large piles of data

APPROPRIATE DATA

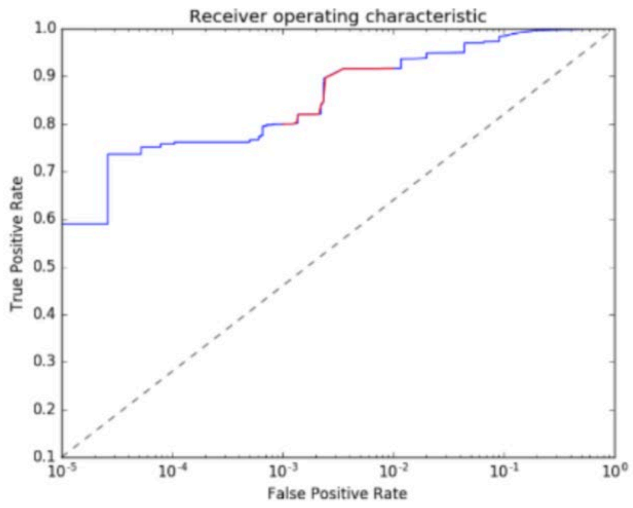
- Data appropriate to the problem you selected
- Known good
- Known bad
- Use 80% of each so you can use the other 20% for testing & tuning

DETERMINE PROPER FEATURES

- Features == Specific data points
  - URL in HTTP headers
  - User Agent in HTTP headers
  - Event ID in Windows event logs
  - DNS Query & Response in DNS traffic
- More != better necessarily
- If in doubt you can read in all the possible features and compute the amount of variance in each. High variance features will usually be your best option.



(a) The initial parameters.



(b) The modified parameters.

Figure 6: The ROC Curve Produced by the Model under Different Settings of Parameters.

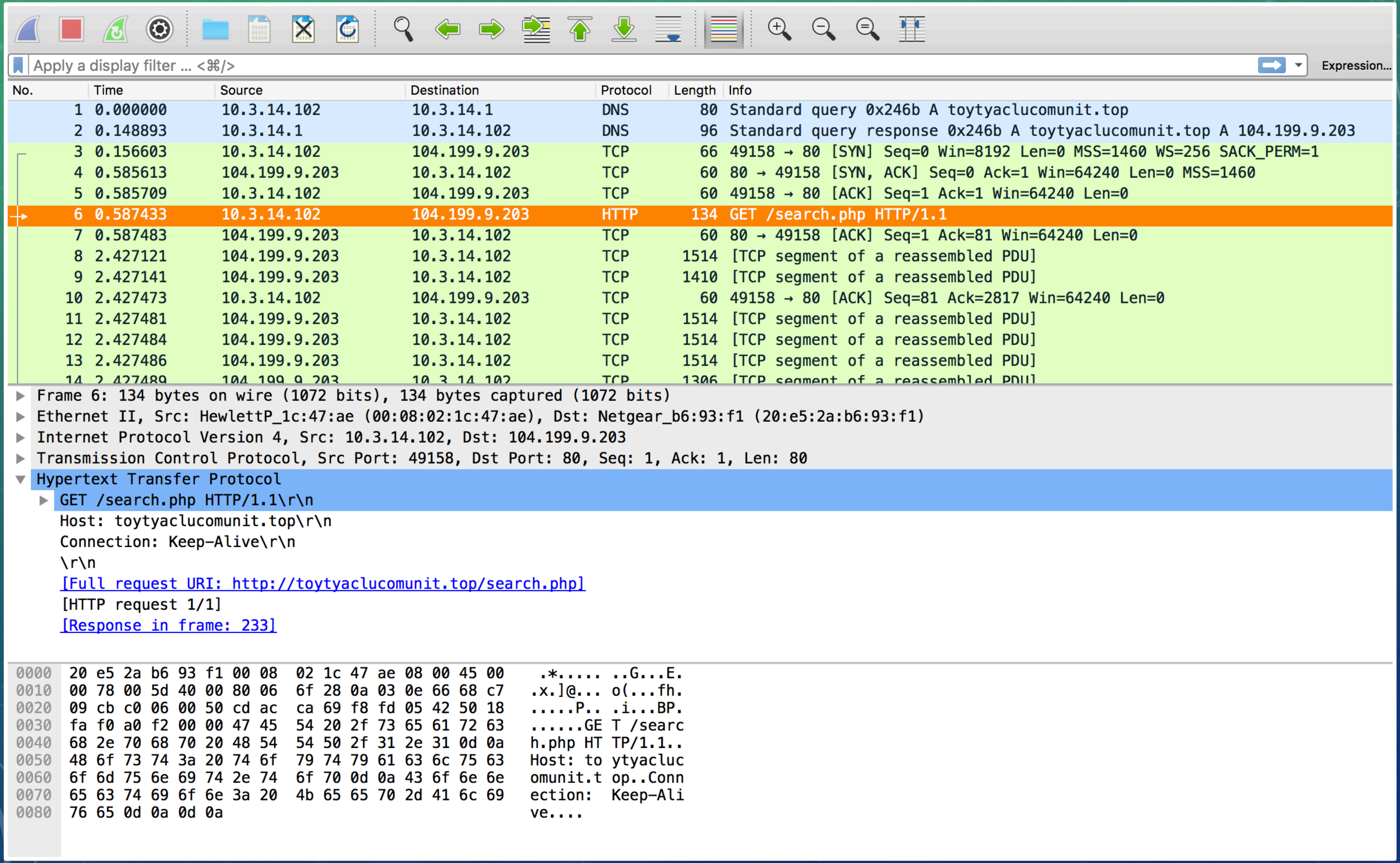
Excerpt from “Practical Cyborgism” by David J. Bianco and Chris McCubbin :
https://speakerdeck.com/davidjbianco/practical-cyborgism-getting-started-with-machine-learning-for-incident-detection

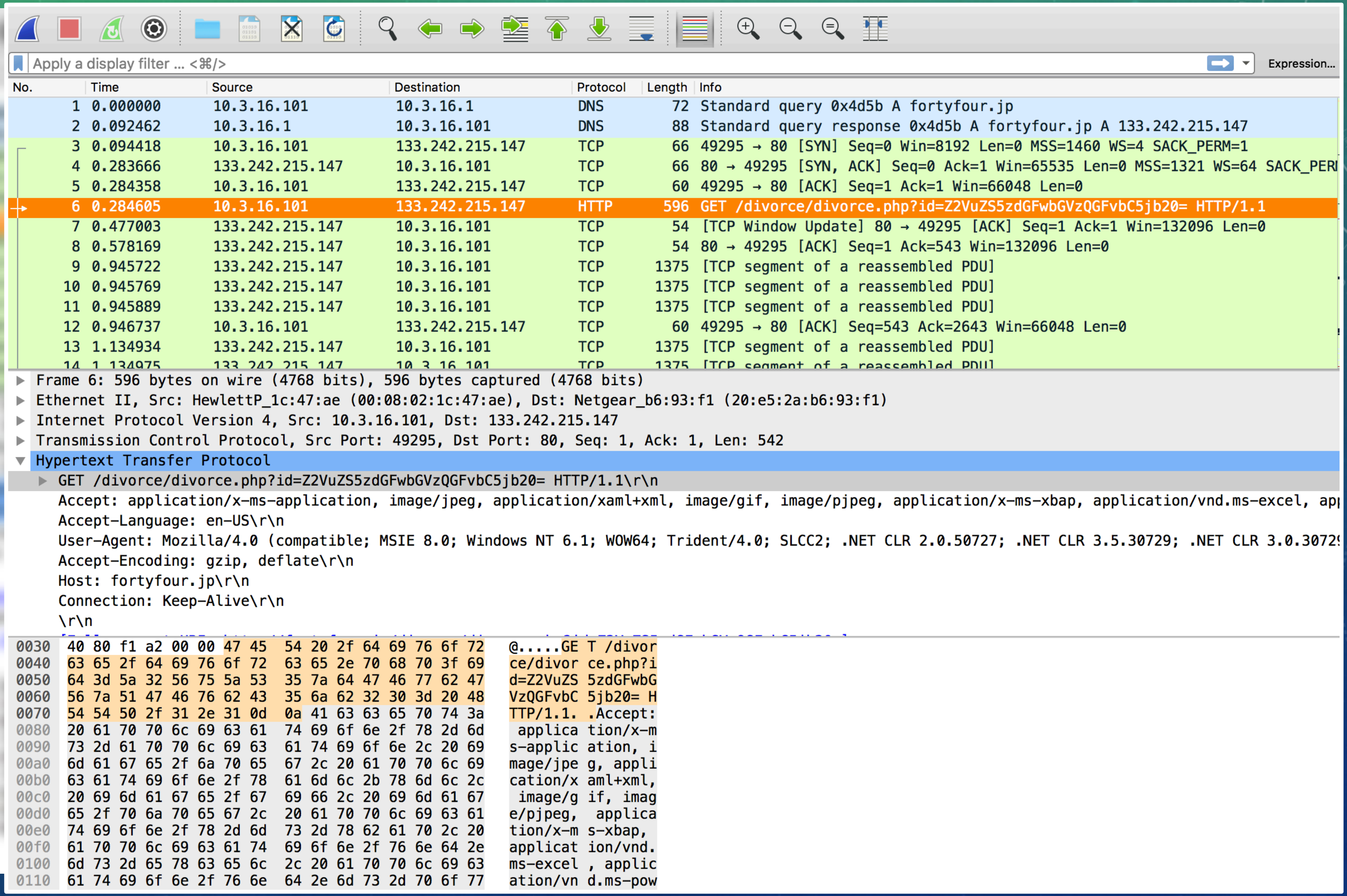
BUILD YOUR TOOL

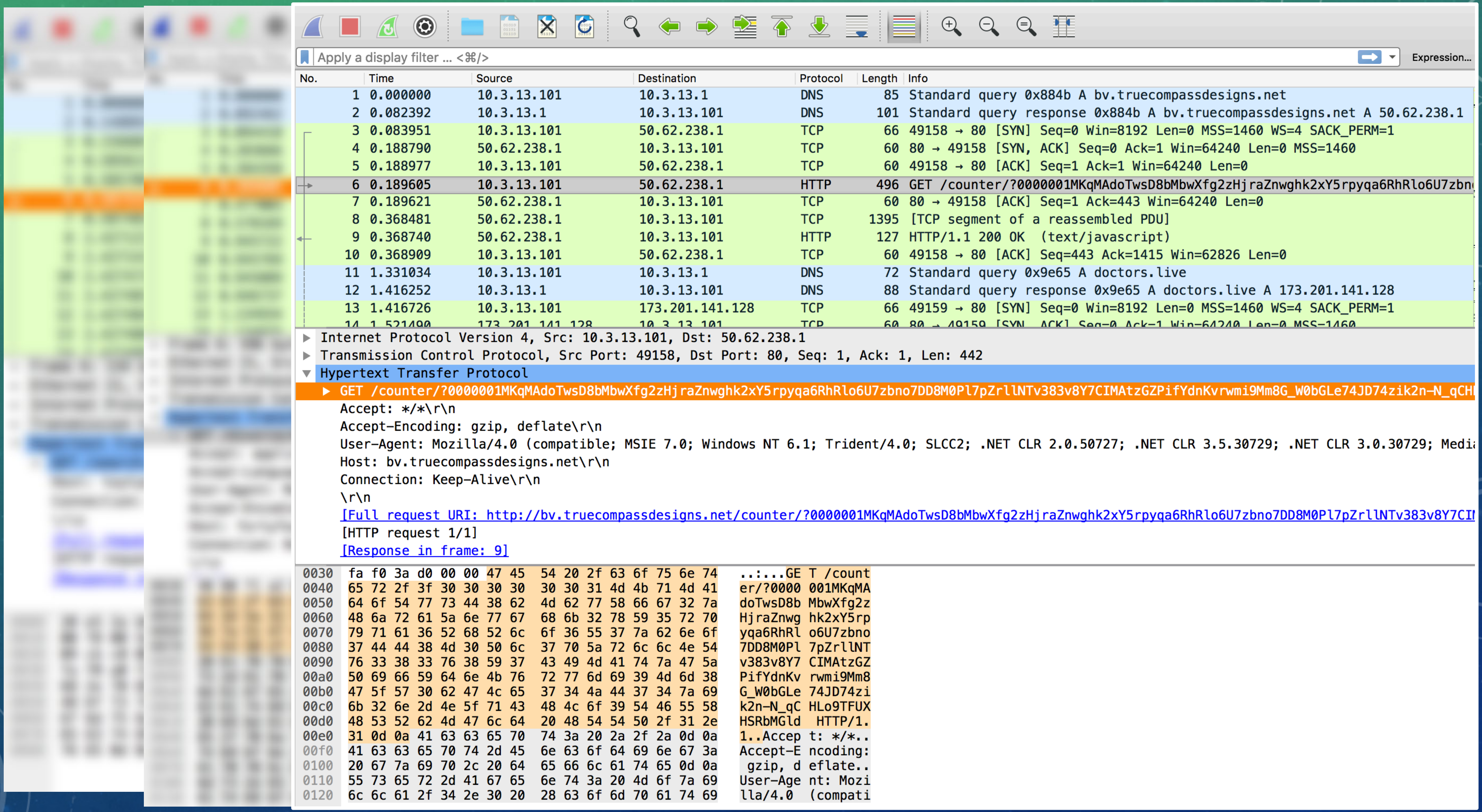
- Python is a great option
  - Pandas, NumPy, and Sci-Kit Learn do almost all of the heavy lifting
- You'll need a way to parse the data so it can be analyzed
- You'll need to select an algorithm
  - When in doubt start with Random Forest
- You'll need a training mode and an analysis mode
- Trainer will need to parse the data, run it through the algorithm and save out the model
- Analyzer will need to load the mode, parse the data to be evaluated, and call out any items flagged by the model

TEST & TUNE

- Use the trainer to build the model off of the 80% known good and known bad data
- Run the resulting model against your 20% known good and known bad to see if the model predicts them properly
- Try changing your feature selection
- Try substituting different algorithms
- If you’re models are still performing badly then you most likely have problems in your data


Apply a display filter ... <⌘/>
Expression...
No. Time Source Destination Protocol Length Info
1 0.000000 10.3.14.102 10.3.14.1 DNS 80 Standard query 0x246b A toytyaclucomunit.top
2 0.148893 10.3.14.1 10.3.14.102 DNS 96 Standard query response 0x246b A toytyaclucomunit.top A 104.199.9.203
3 0.156603 10.3.14.102 104.199.9.203 TCP 66 49158 → 80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
4 0.585613 104.199.9.203 10.3.14.102 TCP 60 80 → 49158 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
5 0.585709 10.3.14.102 104.199.9.203 TCP 60 49158 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
6 0.587433 10.3.14.102 104.199.9.203 HTTP 134 GET /search.php HTTP/1.1
7 0.587483 104.199.9.203 10.3.14.102 TCP 60 80 → 49158 [ACK] Seq=1 Ack=81 Win=64240 Len=0
8 2.427121 104.199.9.203 10.3.14.102 TCP 1514 [TCP segment of a reassembled PDU]
9 2.427141 104.199.9.203 10.3.14.102 TCP 1410 [TCP segment of a reassembled PDU]
10 2.427473 10.3.14.102 104.199.9.203 TCP 60 49158 → 80 [ACK] Seq=81 Ack=2817 Win=64240 Len=0
11 2.427481 104.199.9.203 10.3.14.102 TCP 1514 [TCP segment of a reassembled PDU]
12 2.427484 104.199.9.203 10.3.14.102 TCP 1514 [TCP segment of a reassembled PDU]
13 2.427486 104.199.9.203 10.3.14.102 TCP 1514 [TCP segment of a reassembled PDU]
Frame 6: 134 bytes on wire (1072 bits), 134 bytes captured (1072 bits)
Ethernet II, Src: HewlettP_1c:47:ae (00:08:02:1c:47:ae), Dst: Netgear_b6:93:f1 (20:e5:2a:b6:93:f1)
Internet Protocol Version 4, Src: 10.3.14.102, Dst: 104.199.9.203
Transmission Control Protocol, Src Port: 49158, Dst Port: 80, Seq: 1, Ack: 1, Len: 80
Hypertext Transfer Protocol
GET /search.php HTTP/1.1\r\n
Host: toytyaclucomunit.top\r\n
Connection: Keep-Alive\r\n
\r\n
[Full request URI: http://toytyaclucomunit.top/search.php]
[HTTP request 1/1]
[Response in frame: 233]
0000 20 e5 2a b6 93 f1 00 08 02 1c 47 ae 08 00 45 00 .*..... ..G...E.
0010 00 78 00 5d 40 00 80 06 6f 28 0a 03 0e 66 68 c7 .x.]@... o(...fh.
0020 09 cb c0 06 00 50 cd ac ca 69 f8 fd 05 42 50 18 .....P.. .i...BP.
0030 fa f0 a0 f2 00 00 47 45 54 20 2f 73 65 61 72 63 ......GE T /searc
0040 68 2e 70 68 70 20 48 54 54 50 2f 31 2e 31 0d 0a h.php HT TP/1.1..
0050 48 6f 73 74 3a 20 74 6f 79 74 79 61 63 6c 75 63 Host: to ytyacluc
0060 6f 6d 75 6e 69 74 2e 74 6f 70 0d 0a 43 6f 6e 6e omunit.t op..Conn
0070 65 63 74 69 6f 6e 3a 20 4b 65 65 70 2d 41 6c 69 ection: Keep-Ali
0080 76 65 0d 0a 0d 0a ve....


Apply a display filter ... <⌘/>
Expression...
No. Time Source Destination Protocol Length Info
1 0.000000 10.3.16.101 10.3.16.1 DNS 72 Standard query 0x4d5b A fortyfour.jp
2 0.092462 10.3.16.1 10.3.16.101 DNS 88 Standard query response 0x4d5b A fortyfour.jp A 133.242.215.147
3 0.094418 10.3.16.101 133.242.215.147 TCP 66 49295 → 80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM=1
4 0.283666 133.242.215.147 10.3.16.101 TCP 66 80 → 49295 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1321 WS=64 SACK_PER
5 0.284358 10.3.16.101 133.242.215.147 TCP 60 49295 → 80 [ACK] Seq=1 Ack=1 Win=66048 Len=0
6 0.284605 10.3.16.101 133.242.215.147 HTTP 596 GET /divorce/divorce.php?id=Z2VuZS5zdGFwbGVzQGFvbC5jb20= HTTP/1.1
7 0.477003 133.242.215.147 10.3.16.101 TCP 54 [TCP Window Update] 80 → 49295 [ACK] Seq=1 Ack=1 Win=132096 Len=0
8 0.578169 133.242.215.147 10.3.16.101 TCP 54 80 → 49295 [ACK] Seq=1 Ack=543 Win=132096 Len=0
9 0.945722 133.242.215.147 10.3.16.101 TCP 1375 [TCP segment of a reassembled PDU]
10 0.945769 133.242.215.147 10.3.16.101 TCP 1375 [TCP segment of a reassembled PDU]
11 0.945889 133.242.215.147 10.3.16.101 TCP 1375 [TCP segment of a reassembled PDU]
12 0.946737 10.3.16.101 133.242.215.147 TCP 60 49295 → 80 [ACK] Seq=543 Ack=2643 Win=66048 Len=0
13 1.134934 133.242.215.147 10.3.16.101 TCP 1375 [TCP segment of a reassembled PDU]
Frame 6: 596 bytes on wire (4768 bits), 596 bytes captured (4768 bits)
Ethernet II, Src: HewlettP_1c:47:ae (00:08:02:1c:47:ae), Dst: Netgear_b6:93:f1 (20:e5:2a:b6:93:f1)
Internet Protocol Version 4, Src: 10.3.16.101, Dst: 133.242.215.147
Transmission Control Protocol, Src Port: 49295, Dst Port: 80, Seq: 1, Ack: 1, Len: 542
Hypertext Transfer Protocol
GET /divorce/divorce.php?id=Z2VuZS5zdGFwbGVzQGFvbC5jb20= HTTP/1.1\r\n
Accept: application/x-ms-application, image/jpeg, application/xaml+xml, image/gif, image/pjpeg, application/x-ms-xbap, application/vnd.ms-excel, ap
Accept-Language: en-US\r\n
User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; WOW64; Trident/4.0; SLCC2; .NET CLR 2.0.50727; .NET CLR 3.5.30729; .NET CLR 3.0.3072
Accept-Encoding: gzip, deflate\r\n
Host: fortyfour.jp\r\n
Connection: Keep-Alive\r\n
\r\n
0030 40 80 f1 a2 00 00 47 45 54 20 2f 64 69 76 6f 72 @.....GE T /divor
0040 63 65 2f 64 69 76 6f 72 63 65 2e 70 68 70 3f 69 ce/divor ce.php?i
0050 64 3d 5a 32 56 75 5a 53 35 7a 64 47 46 77 62 47 d=Z2VuZS 5zdGFwbG
0060 56 7a 51 47 46 76 62 43 35 6a 62 32 30 3d 20 48 VzQGFvbC 5jb20= H
0070 54 54 50 2f 31 2e 31 0d 0a 41 63 63 65 70 74 3a TTP/1.1. .Accept:
0080 20 61 70 70 6c 69 63 61 74 69 6f 6e 2f 78 2d 6d applica tion/x-m
0090 73 2d 61 70 70 6c 69 63 61 74 69 6f 6e 2c 20 69 s-applic ation, i
00a0 6d 61 67 65 2f 6a 70 65 67 2c 20 61 70 70 6c 69 mage/jpe g, appli
00b0 63 61 74 69 6f 6e 2f 78 61 6d 6c 2b 78 6d 6c 2c cation/x aml+xml,
00c0 20 69 6d 61 67 65 2f 67 69 66 2c 20 69 6d 61 67 image/g if, imag
00d0 65 2f 70 6a 70 65 67 2c 20 61 70 70 6c 69 63 61 e/pjpeg, applica
00e0 74 69 6f 6e 2f 78 2d 6d 73 2d 78 62 61 70 2c 20 tion/x-m s-xbap,
00f0 61 70 70 6c 69 63 61 74 69 6f 6e 2f 76 6e 64 2e applicat ion/vnd.
0100 6d 73 2d 65 78 63 65 6c 2c 20 61 70 70 6c 69 63 ms-excel , applic
0110 61 74 69 6f 6e 2f 76 6e 64 2e 6d 73 2d 70 6f 77 ation/vn d.ms-pow



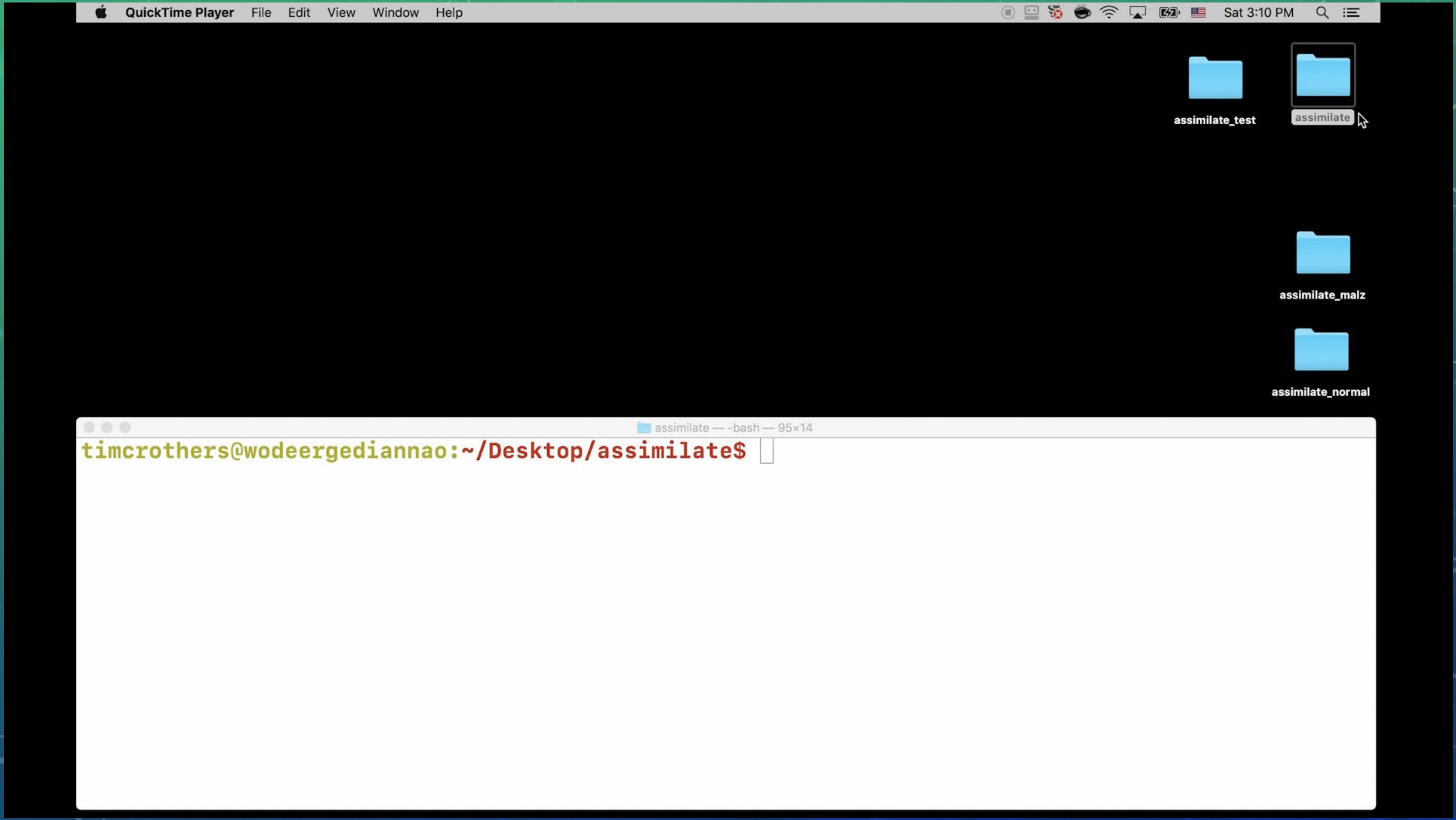
Apply a display filter … <⌘/>
Expression…
No. Time Source Destination Protocol Length Info
1 0.000000 10.3.13.101 10.3.13.1 DNS 85 Standard query 0x884b A bv.truecompassdesigns.net
2 0.082392 10.3.13.1 10.3.13.101 DNS 101 Standard query response 0x884b A bv.truecompassdesigns.net A 50.62.238.1
3 0.083951 10.3.13.101 50.62.238.1 TCP 66 49158 → 80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM=1
4 0.188790 50.62.238.1 10.3.13.101 TCP 60 80 → 49158 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
5 0.188977 10.3.13.101 50.62.238.1 TCP 60 49158 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
6 0.189605 10.3.13.101 50.62.238.1 HTTP 496 GET /counter/?0000001MKqMAdoTwsD8bMbwXfg2zHjraZnwghk2xY5rpyqa6RhRlo6U7zbn
7 0.189621 50.62.238.1 10.3.13.101 TCP 60 80 → 49158 [ACK] Seq=1 Ack=443 Win=64240 Len=0
8 0.368481 50.62.238.1 10.3.13.101 TCP 1395 [TCP segment of a reassembled PDU]
9 0.368740 50.62.238.1 10.3.13.101 HTTP 127 HTTP/1.1 200 OK (text/javascript)
10 0.368909 10.3.13.101 50.62.238.1 TCP 60 49158 → 80 [ACK] Seq=443 Ack=1415 Win=62826 Len=0
11 1.331034 10.3.13.101 10.3.13.1 DNS 72 Standard query 0x9e65 A doctors.live
12 1.416252 10.3.13.1 10.3.13.101 DNS 88 Standard query response 0x9e65 A doctors.live A 173.201.141.128
13 1.416726 10.3.13.101 173.201.141.128 TCP 66 49159 → 80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM=1
Internet Protocol Version 4, Src: 10.3.13.101, Dst: 50.62.238.1
Transmission Control Protocol, Src Port: 49158, Dst Port: 80, Seq: 1, Ack: 1, Len: 442
Hypertext Transfer Protocol
GET /counter/?0000001MKqMAdoTwsD8bMbwXfg2zHjraZnwghk2xY5rpyqa6RhRlo6U7zbno7DD8M0Pl7pZrllNTv383v8Y7CIMAtzGZPifYdnKvrwmi9Mm8G_W0bGLe74JD74zik2n-N_qCH
Accept: */*\r\n
Accept-Encoding: gzip, deflate\r\n
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.1; Trident/4.0; SLCC2; .NET CLR 2.0.50727; .NET CLR 3.5.30729; .NET CLR 3.0.30729; Medi
Host: bv.truecompassdesigns.net\r\n
Connection: Keep-Alive\r\n
\r\n
[Full request URI: http://bv.truecompassdesigns.net/counter/?0000001MKqMAdoTwsD8bMbwXfg2zHjraZnwghk2xY5rpyqa6RhRlo6U7zbno7DD8M0Pl7pZrllNTv383v8Y7CI
[HTTP request 1/1]
[Response in frame: 9]
0030 fa f0 3a d0 00 00 47 45 54 20 2f 63 6f 75 6e 74 ..:...GE T /count
0040 65 72 2f 3f 30 30 30 30 30 30 31 4d 4b 71 4d 41 er/?0000 001MKqMA
0050 64 6f 54 77 73 44 38 62 4d 62 77 58 66 67 32 7a doTwsD8b MbwXfg2z
0060 48 6a 72 61 5a 6e 77 67 68 6b 32 78 59 35 72 70 HjraZnwg hk2xY5rp
0070 79 71 61 36 52 68 52 6c 6f 36 55 37 7a 62 6e 6f yqa6RhRl o6U7zbno
0080 37 44 44 38 4d 30 50 6c 37 70 5a 72 6c 6c 4e 54 7DD8M0Pl 7pZrllNT
0090 76 33 38 33 76 38 59 37 43 49 4d 41 74 7a 47 5a v383v8Y7 CIMAtzGZ
00a0 50 69 66 59 64 6e 4b 76 72 77 6d 69 39 4d 6d 38 PifYdnKv rwmi9Mm8
00b0 47 5f 57 30 62 47 4c 65 37 34 4a 44 37 34 7a 69 G_W0bGLe 74JD74zi
00c0 6b 32 6e 2d 4e 5f 71 43 48 4c 6f 39 54 46 55 58 k2n-N_qC HLo9TFUX
00d0 48 53 52 62 4d 47 6c 64 20 48 54 54 50 2f 31 2e HSRbMGld HTTP/1.
00e0 31 0d 0a 41 63 63 65 70 74 3a 20 2a 2f 2a 0d 0a 1..Accep t: */*..
00f0 41 63 63 65 70 74 2d 45 6e 63 6f 64 69 6e 67 3a Accept-E ncoding:
0100 20 67 7a 69 70 2c 20 64 65 66 6c 61 74 65 0d 0a gzip, d eflate..
0110 55 73 65 72 2d 41 67 65 6e 74 3a 20 4d 6f 7a 69 User-Age nt: Mozi
0120 6c 6c 61 2f 34 2e 30 20 28 63 6f 6d 70 61 74 69 lla/4.0 (compati

CYBER HUNTING CHALLENGES

- Too few experienced practitioners
- Takes too long to develop experienced practitioners
- Too much data to look through
- Hunts are periodic

ASSIMILATE BUILD STEP-BY-STEP

- Gathered the real world network data (one week > 10TB)
- Used Bro (Zeek) to convert the packet captures into metadata (HTTP)
- Compiled over a years worth of packet captures from malware and converted with Bro similarly
- Cleaned the Malicious Bro metadata of the non-malware activity
- Used the malicious data to clean the real world network data
- Tested for algorithm, parameters and features
- Coded trainer & model application, tested, iterated
- 🍺



QuickTime Player
File
Edit
View
Window
Help
Sat 3:10 PM
assimilate_test
assimilate
assimilate_malz
assimilate_normal
assimilate — -bash — 95×14
timcrothers@wodeergediannao:~/Desktop/assimilate$

SO WHAT DID YOU JUST SEE?

- Python script using a trained Naïve Bayes algorithm based model against 37,440 HTTP headers
- Found 46 things that looked suspicious
- 0.12% suspicious

WHAT'S NEEDED TO DO THIS?

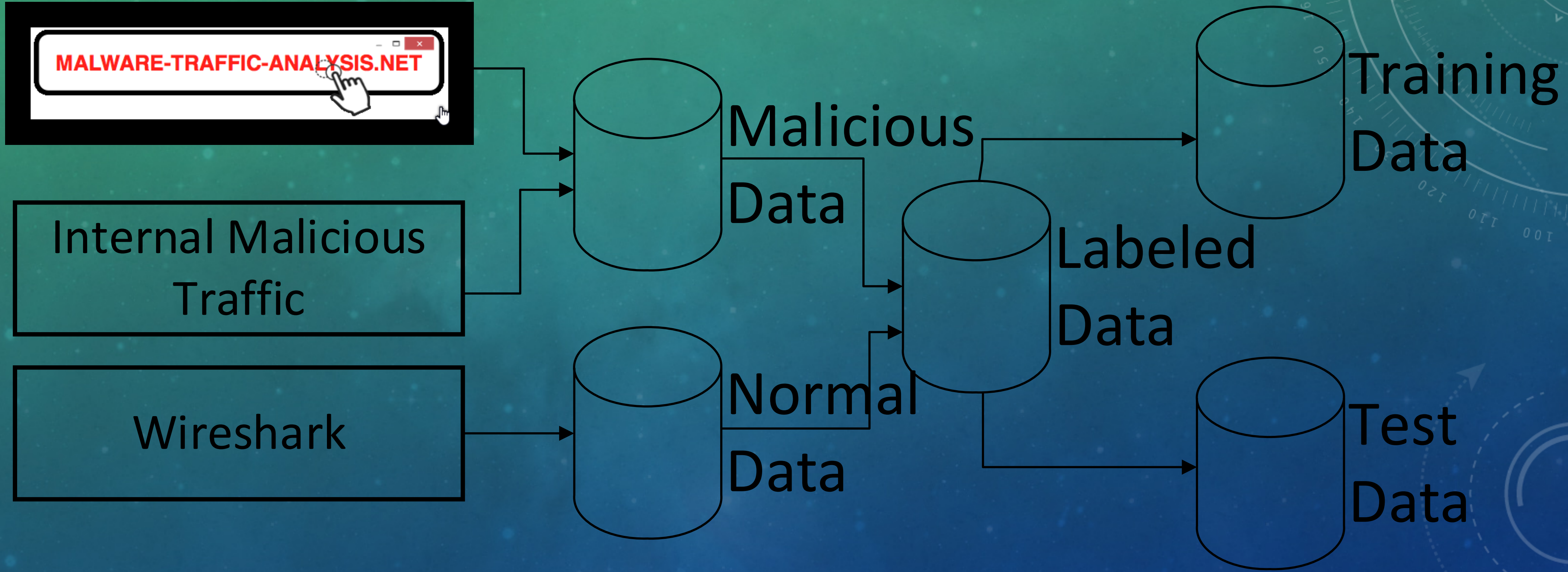
- Python
- Sci-kit Learn & Pandas (python modules)
- Packet captures of non-malicious activity
- Packet captures of malicious activity
- Bro
- Bro HTTP_Header script
- Assimilate python scripts

Customized code at: **https://github.com/Soinull/assimilate**

STEP BY STEP

1. Collect and process training data
2. Train model
3. Assess actual data files
4. Validate suspicious entries
5. Retrain as needed to improve accuracy
6. 

TRAINING DATA



PROCESSING PACKET CAPTURES

- Install customized HTTP_Headers Bro module
- Process all packet captures with “Bro –r”

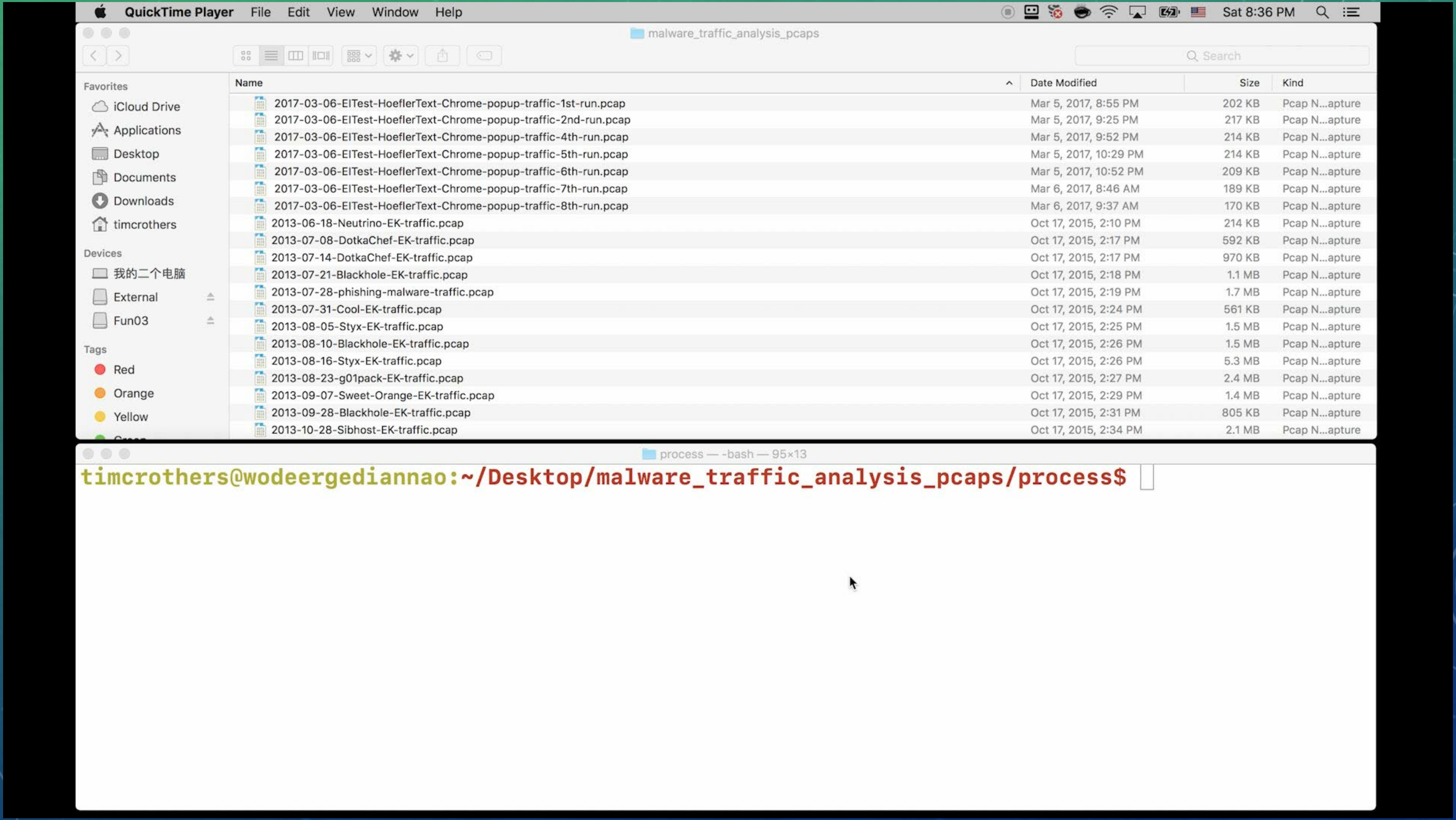
CUSTOMIZED BRO HTTP_HEADERS

```
event http_all_headers(c: connection, is_orig: bool, hlist: mime_header_list)
{
    local my_log: Info;
    local origin: string;
    local identifier: string;
    # local event_json_string: string;
    local event_kv_string: string;

    # Is the header from a client request or server response
    if ( is_orig )
        origin = "client";
    else
        origin = "server";

    # If we don't have a header_info_vector than punt
    if ( ! c?$http || ! c$http?$header_info_vector )
        return;

    print c$http$header_info_vector;
```

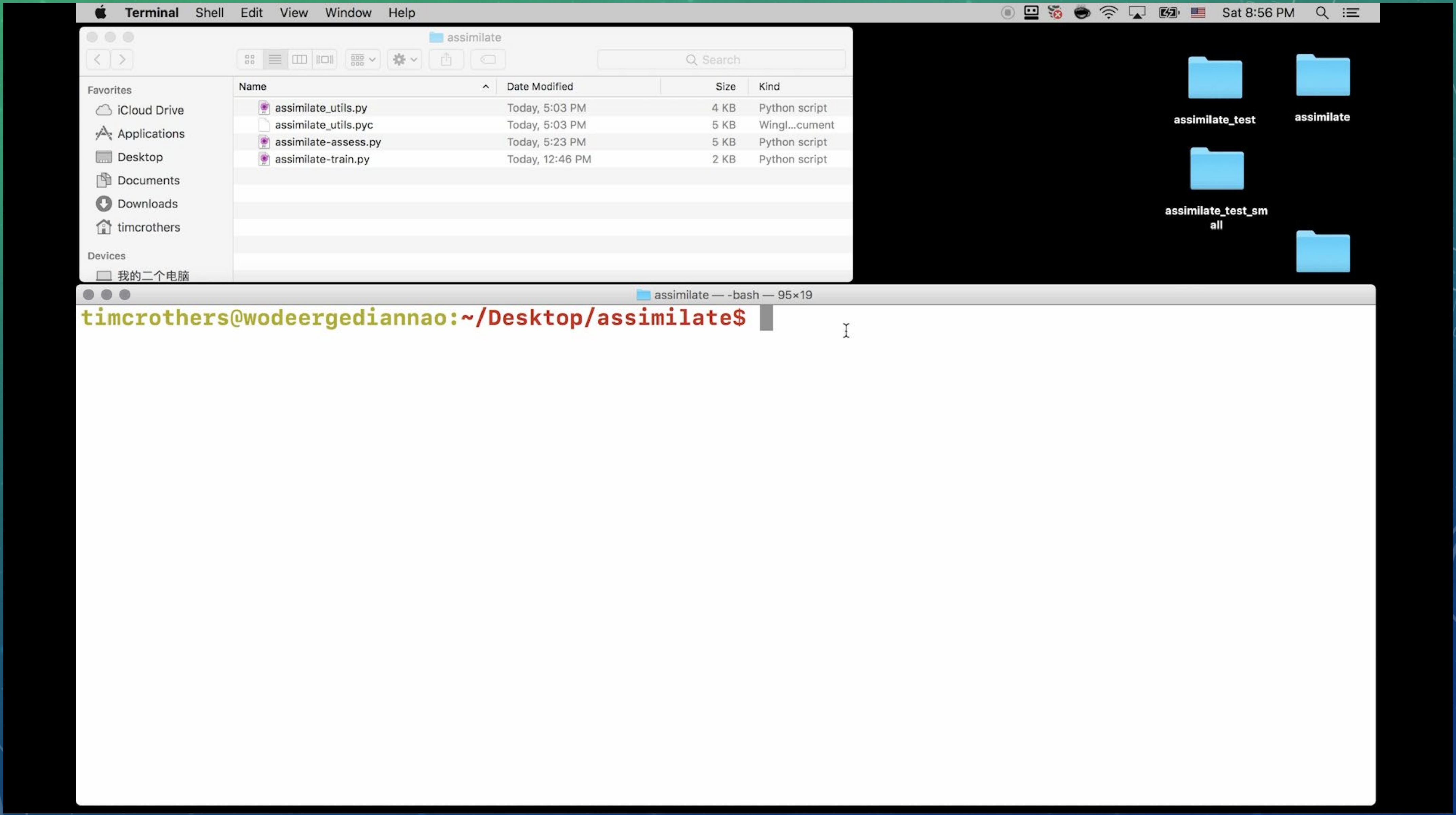
PROCESS SHELL SCRIPT

```
# Example script to iterate over pcap files to get corresponding http.log and httpheader.log files
for file in ../*.pcap
do
    name=${file##*/}
    echo $name
    base=${name%.pcap}
    echo $base
    cp ../"$file" .
    bro -r "$file" custom/BrowserFingerprinting/http-headers.bro
    mv http.log ../"$base"_http.log
    mv httpheaders.log ../"$base"_httpheaders.log
    rm -f *.log *.pcap
done
```

PROCESSING PCAPS



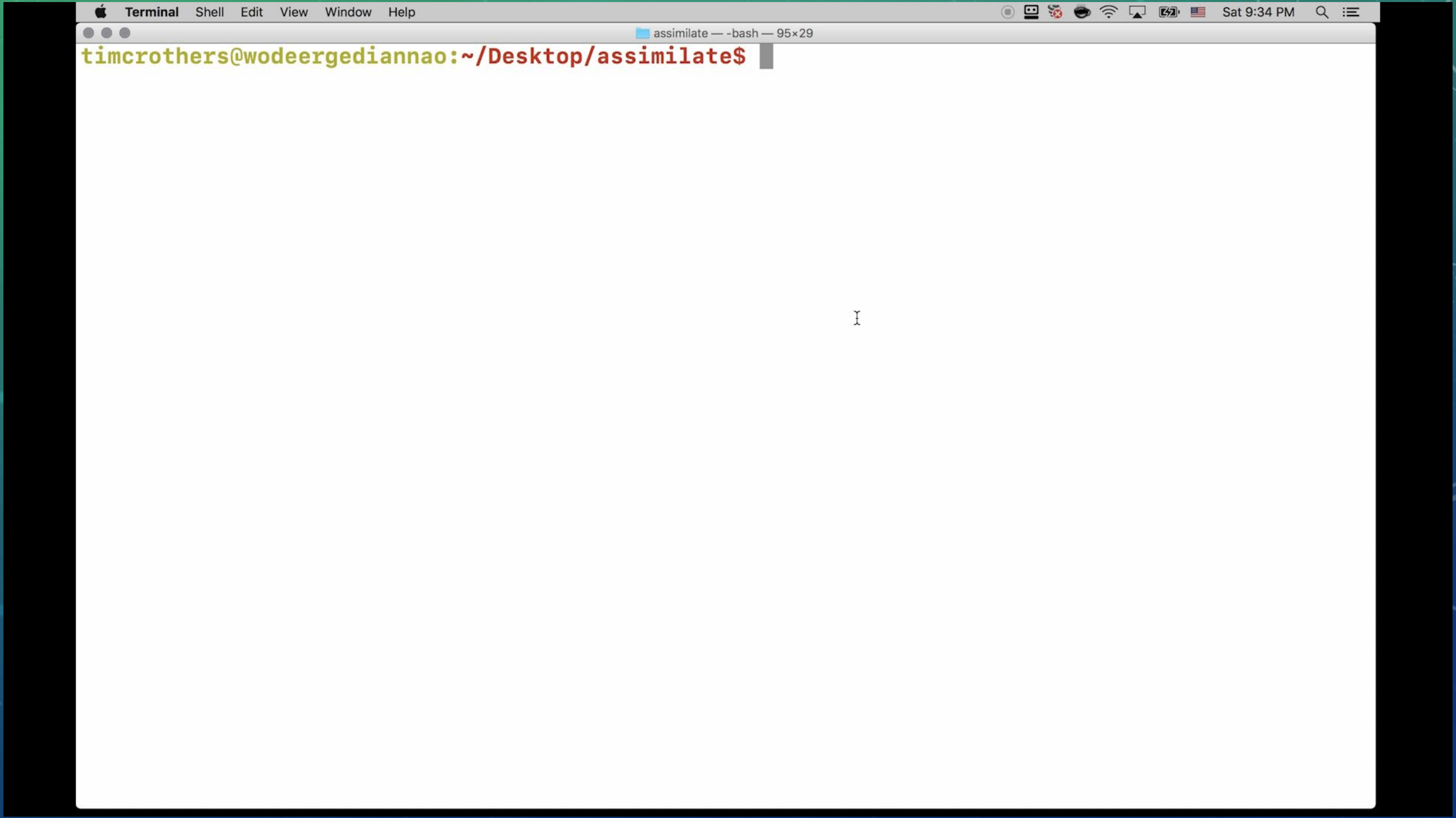
ASSIMILATE-TRAIN.PY



BUILDING ML MODELS FOR HUNTING

- More data == More accuracy
- More data == Slower speed
- Bro Header Normalization == Lower Accuracy
- Tighter Scoping == More Accuracy

ASSIMILATE-ASSESS.PY



DIFFICULT?

```
data = DataFrame({'header': [], 'class': []})
blr = BroLogReader()
print('Reading normal data...')
data = data.append(blr.dataFrameFromDirectory(opts.normaldata, 'good'))
print('Reading malicious data...')
data = data.append(blr.dataFrameFromDirectory(opts.maliciousdata, 'bad'))
print('Vectorizing data...')
vectorizer = CountVectorizer()
counts = vectorizer.fit_transform(data['header'].values)
classifier = MultinomialNB()
targets = data['class'].values
classifier.fit(counts, targets)
print('Writing out models...')
joblib.dump(vectorizer, opts.vectorizerfile)
joblib.dump(classifier,opts.bayesianfile)
```

EXAMPLES

Clearcut – https://github.com/DavidJBianco/Clearcut

Assimilate – https://github.com/Soinull/assimilate

Malicious Macro Bot – https://github.com/egaus/MaliciousMacroBot

RECOMMENDED RESOURCES

Real world bad traffic – https://www.malware-traffic-analysis.net/

Basics - https://speakerdeck.com/davidjbianco/introduction-to-data-analysis-with-security-onion-and-other-open-source-tools

Mid-level - https://speakerdeck.com/davidjbianco/practical-cyborgism-getting-started-with-machine-learning-for-incident-detection

THANK YOU!

badsecurity@gmail.com

https://github.com/soinull/PracticalApplicationofML

linkedin.com/in/timcrothers/