Weaponizing data science for social engineering: Automated E2E spear phishing on Twitter

John Seymour and Philip Tully
{jseymour, ptully}@zerofox.com

Introduction and Abstract

Historically, machine learning for information security has prioritized defense: think intrusion detection systems, malware classification and botnet traffic identification. Offense can benefit from data just as well. Social networks, especially Twitter with its access to extensive personal data, bot-friendly API, colloquial syntax and prevalence of shortened links, are the perfect venues for spreading machine-generated content.

We present SNAP_R, a recurrent neural network that learns to tweet phishing posts targeting specific users. The model is trained using spear phishing pen-testing data, and in order to make a click-through more likely, it is dynamically seeded with topics extracted from timeline posts of both the target and the users they retweet or follow. We augment the model with clustering to identify high value targets based on their level of social engagement such as their number of followers and retweets, and measure success using click-rates of IP-tracked links. Taken together, these techniques enable the world's first automated end-to-end spear phishing campaign generator for Twitter.

This research and code are for educational research purposes only. Possible uses include internal pen-testing, staff recruitment, or social engagement.

Contents

Introduction and Abstract	1
Background	2
Machine Learning: An Offensive Approach	3
High-level Description of Tool	3
Target Discovery	4
Automated Spear Phishing	5
Conclusion	7
References	7

Background

Historically, the security community has used Machine Learning (ML) in a defensive manner, for example in classifying malicious binaries or finding anomalous network traffic. Even now, startups continue to spring up advertising novel techniques for detecting inbound threats. But much of InfoSec is dedicated to identifying critical weaknesses and vulnerabilities through offense. While tools such as Metasploit [1] exist for automating Red-Team activities, there has been little work toward using ML as a weapon.

Social engineering, particularly phishing, is one of the oldest yet still most effective weapons for exploitation. Early phishing attempts took a shotgun approach: only a few targets out of millions needed to bite in order for an attack to succeed. Spear phishing introduced specificity into this process by narrowing the target base to fewer people, enabling attackers to focus efforts on high value targets. It increased the likelihood of success by preemptively gathering personal information and using it to gain the target's trust.

Today, phishing attacks span across a variety of platforms. A prime example is social media, which exposes this vulnerability with its ever-expanding user base, high usage statistics, and strong incentive to disclose personal data. The challenge is how to exploit these natural weaknesses at scale, combining the shotgun approach of phishing campaigns with the specificity and success of targeted spear phishing.

Enter Machine Learning.

Machines have been successfully fooling humans since the first AI chatbot ELIZA in 1966 [2]. At a high level, ML is a statistical tool: given enough data, it can detect patterns that reveal information about unencountered samples. Natural Language Processing (NLP) is a use case of ML where raw text is the data source from which patterns are extracted. NLP has been successfully used for many applications, a significant application being spam detection [3]. Phishing is particularly amenable to the NLP approach because recurring patterns of text can be utilized to identify topics the target is interested in and generate sentences the target might respond to.

The Social-Engineer Toolkit [4] is the gold standard for automating Social Engineering attack payloads, but users still have to gather their own data and write the "front-end" email delivery service. If we could automatically churn through unorganized personal data and generate a personalized phishing message, we can automate the spear phishing process completely and operate at a much larger scale with higher success rates. This would be invaluable for educational purposes and would make internal penetration testing on social media viable.

Machine Learning: An Offensive Approach

A proof of concept for automatically generating phishing emails, Honey-Phish, was first demonstrated at Shmoocon 2016 [5]. It targeted phishers themselves and attempted to trick them into clicking a link. Since a corpus of phishing emails was unavailable, Honey-Phish used a Markov Model [6] trained on Reddit posts from /r/personalfinance. But the model-generated English was noticeably different than what a human would write, so only 2 phishers responded out of 41 phishing emails generated.

We make this approach viable by switching the social environment: we post on Twitter instead of sending an email. This allows us to scale both on the number of targets and on accuracy by tailoring the posts using the account's profile. Furthermore, on Twitter, the culture readily accepts broken English (we call this Twitterese) because of its 140-character limit.

Twitterese has several interesting side effects that can be utilized for ML. Because short posts are the norm, posts produced by the model have decreased probability of grammatical error. Furthermore, to adapt to this constraint, links on Twitter are almost always shortened. This can be used to obfuscate a phishing domain, increasing the rate of clickthroughs. Other useful quirks of Twitterese include the abundant use of emojis, and the fact that victims disclose an absurd amount of personal data. These factors can all be used to generate more human-like posts and avoid suspicion.

High-level Description of ***SNAP_R***

SNAP_R takes in a list of twitter usernames (e.g. from Twitter's User Streaming API endpoint), then triages the users based on probability of success, which is determined from their individual profiles and post topics. If the user is relatively more vulnerable or has a high value, ***SNAP_R*** selects them as a target, then automatically posts to them content with an embedded "phishing" link.

When a target is selected, ***SNAP_R*** extracts a topic and the timing history for that user's tweets and replies in order to seed the phishing tweet generation. ***SNAP_R*** allows for tuning the tradeoff of throughput and accuracy by providing a parameter which defines the number of timeline calls per user. We found that the most frequently occuring words, excluding stopwords like common prepositions, were the most effective seeds for the tool. ***SNAP_R*** bucketizes the posts by the hour they were posted, and it schedules the phishing tweet to be sent at a random minute within the hour that the user historically posts most frequently.

We pre-train a neural network for generating tweets using a combination of spear phish pen-testing data, Reddit submissions, and tweets. We also allow for Markov models trained using the target's recent posts. We measure two types of successful phishing results: responses from questions in the DEFCON SECTF [7] as a proxy for successful information exfiltration and clicks from an IP-traced shortened link to simulate a successful pwn.

A powerful aspect of our method is that it generalizes across different languages and demographics out-of-the-box, since generated tweets simply reflect content that's publicly available on the target's timeline.

Target Discovery

If we indiscriminately spam everyone on Twitter with phishing links, we would quickly be discovered and have our accounts terminated due to ToS violations. Therefore, we triage users to determine which ones are either more likely to be phished or provide exceptional value.

We first cluster users into groups based on their profiles, using data such as the amount of information revealed in their profile, follower interactions, and engagement metrics. Further, if any information on their profile indicates that they would be a high-value target (such as job titles, particular list membership, and popularity), we use that data as well.

Using the clusters, we collect users from the Twitter Firehose and predict which cluster the user fits into. If they fit into a cluster displaying features identified as likely to lead to a successful phish, they are selected as a target, and recon for automated profiling is performed. Features extracted from their timeline include:

1. How frequently do they post? When are they likely to respond?
2. What topics do they tend to tweet about or respond to?

We have also considered the following features:

3. What is the sentiment on those topics?
4. Can we extract their location from geotagged media?
5. Can we find any patterns of behavior?
6. Can we find any large events they've gone to recently or plan to go to in the future?

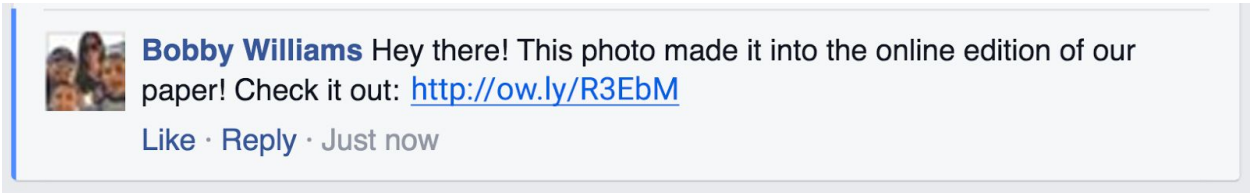
We use these features as parameters to determine when to post the tweet and how to seed the model.

Automated Spear Phishing

After a target is determined, ***SNAP_R*** sends them a machine-generated tweet with either an embedded link or a DEFCON SECTF question. To generate tweets, we use both Markov Models [6] and Long Short-Term Memory (LSTM) recurrent neural networks [8].

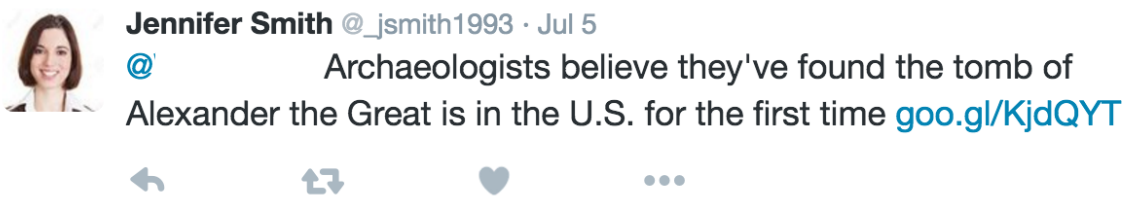
Markov models generate text word by word based on probabilities of word co-occurrences in the training set. For example, if the training data has many instances of the phrase “the cat in the hat”, then if the model generates the word “the”, it will likely generate either “cat” or “hat” as its next word. However, the model does not consider any context prior to the current word when generating the next, so oftentimes the output will be nonsensical.

LSTMs differ from Markov Models by being able to remember context from earlier in the sentence when predicting the next word. LSTMs have been used extensively for NLP because language is naturally sequential and words that are separated by a large distance may still be related to each other. However, the increase in accuracy of LSTMs comes at a cost as they require more time and data to train.



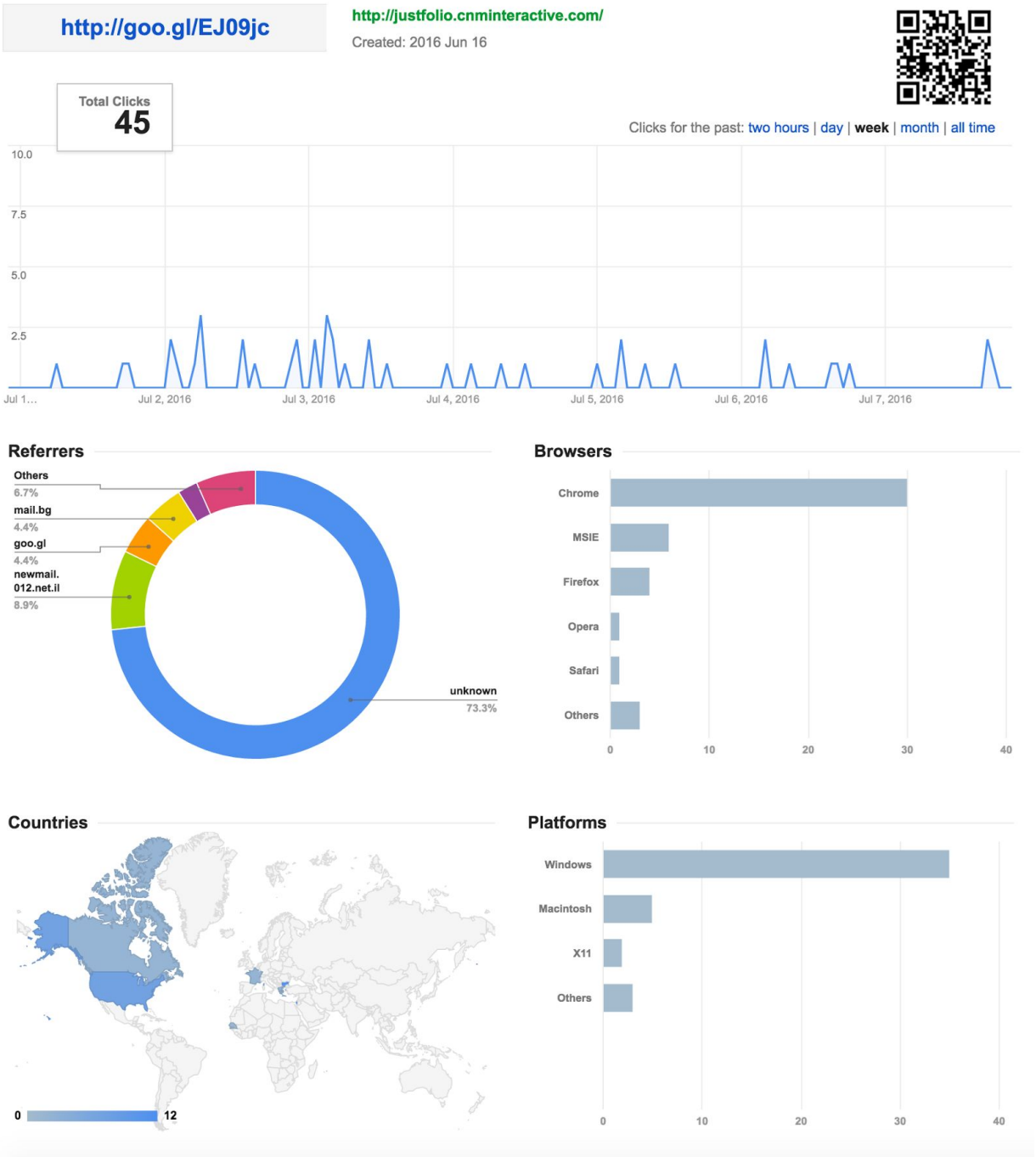
Example of a Facebook phishing post used as training data.

To avoid the computational cost of retraining these models for each user, we pre-train these models on spear phishing data and seed them with a topic generated from automated profiling. There is no standard corpus for phishing emails, much less phishing tweets, so we create our own using spear phishing attacks from numerous sources, e.g. [9]. We supplement this data with streamed tweets from the Twitter API Streaming Endpoint. Finally, since Twitter’s API allows us to post the generated tweet autonomously, we use the target’s own post history to select a time when they are likely to observe and respond to our tweet.



Example of a machine-generated tweet.

To evaluate our model, we place links inside the generated tweets leading to a payload, shortened using goo.gl. If the user clicks through, we record the timestamp, user-agent, and screenname that the tweet was sent to. We prepend generated tweets with a @mention directed at our target in order to decrease the chance that a user other than the target clicks on the link.



Goo.gl analytics from a malicious shortened link.

Though large-scale phishing campaigns tend to have very low compromise rates, they persist because the few examples that do succeed lead to a high return on investment. On tests consisting of 90 users, we found that our automated spear phishing framework had between 30% and 66% success rate. This is more successful than the 5-14% previously reported in large-scale phishing campaigns [10, 11], and comparable to the 45% reported for large-scale manual spear phishing efforts [12]. We attribute our results to the unique risks associated with social media and our ability to leverage data science to target vulnerable users with a highly personalized message.

Conclusion

This work marks an advance in offensive capabilities through automation of a traditionally manual process using ML techniques. Our approach is predicated on the fact that social media is rapidly emerging as an easy target for phishing and social engineering attacks. We use Twitter as our platform because of its low bar for admissible posts, its community tolerance of convenience services like shortened links, its effective API, and its pervasive culture of overexposing personal information.

SNAP_R is entirely data-driven: it employs modeling techniques that learn the relevant textual statistics of successful spear phishing campaigns on social media, and uses those models to generate tailored posts to high risk/high value Twitter users based on their public content. Click-through rates are among the highest ever reported for a large-scale phishing campaign, underscoring the efficacy of coordinated automatic social engineering at scale.

There are existing frameworks such as the Social-Engineer Toolkit that automate the payload of the phishing process, but none that tailor the phishing message to the target. ***SNAP_R*** closes this gap and enables penetration testers to address larger groups of targets while not compromising the quality of the spear phishing post. We present this automated end-to-end spear phishing campaign generator in order to foster greater awareness and understanding of spear phishing and social engineering attacks.

References

[1] Maynor, David. Metasploit toolkit for penetration testing, exploit development, and vulnerability research. Elsevier, 2011.

[2] Weizenbaum, Joseph. "ELIZA—a computer program for the study of natural language communication between man and machine." *Communications of the ACM* 9.1 (1966): 36-45.

[3] Sahami, Mehran, Dumais, Susan, Heckerman, David, and Eric Horvitz. "A Bayesian approach to filtering junk e-mail." *Learning for Text Categorization*: *Papers from the 1998 workshop*. Vol. 62. 1998.

[4] Pavković, Nikola, and Luka Perkov. "Social Engineering Toolkit—A systematic approach to social engineering." *MIPRO, 2011 Proceedings of the 34th International Convention*. IEEE, 2011. https://www.trustedsec.com/social-engineer-toolkit/

[5] Gallagher, Robbie, "Where Do the Phishers Live? Collecting Phishers' Geographic Locations from Automated Honeypots", *2016 ShmooCon*, https://bitbucket.org/rgallagh/honey-phish

[6] Markov, Andrey A. "Extension of the limit theorems of probability theory to a sum of variables connected in a chain". reprinted in Appendix B of: R. Howard. *Dynamic Probabilistic Systems, volume 1: Markov Chains*. John Wiley and Sons, 1971.

[7] http://www.social-engineer.org/wp-content/uploads/2015/11/SECTF-2015_Public.pdf

[8] Gers, Felix A., Jürgen Schmidhuber, and Fred Cummins. "Learning to forget: Continual prediction with LSTM." *Neural Computation* 12.10 (2000): 2451-2471.

[9] https://shkspr.mobi/blog/2015/08/would-you-fall-for-this-twitter-phishing-attack/

[10] Thompson, Steven C. "Phight Phraud." Journal of Accountancy 201.2 (2006): 43.

[11] Jakobsson, Markus, and Jacob Ratkiewicz. "Designing ethical phishing experiments: a study of (ROT13) rOnl query features." *Proceedings of the 15th international conference on World Wide Web*. ACM, 2006.

[12] Bursztein, Elie, et al. "Handcrafted fraud and extortion: Manual account hijacking in the wild." *Proceedings of the 2014 Conference on Internet Measurement Conference*. ACM, 2014.