Domain Specific Honeytokens Based on Natural Language Processing – A Conceptual Model

Tomislav Turek, Tonimir Kišasondi
Laboratory for Open Systems and Security
Faculty of Organization and Informatics
Pavlinska 2, 42000 Varaždin, Croatia
{tomislav.turek, tonimir.kisasondi}@foi.hr

Markus Schatten
Artificial Intelligence Laboratory
Faculty of Organization and Informatics
Pavlinska 2, 42000 Varaždin, Croatia
markus.schatten@foi.hr

Abstract. *This paper presents the idea and conceptual model for keyword modelling by using Natural Language Processing (NLP) in a specific domain. The paper shows that keywords for Honeytokens (HTs) derived from a specific website domain can be generated automatically by extracting concepts of interest from security related or other domain specific texts. A conceptual model for generating such tokens is presented and guidelines for implementation are given. It is argued that such domain specific HTs are a better form of deception technology, that provides a harder challenge for detection from automated attacks and thus improves early detection and incident response procedures in modern complex systems.*

Keywords. honeytokens, honeynets, information security, natural language processing, domain modelling, keyword generation

1 Introduction

Deception technology is designed to detect and prevent an attack on a particular network or an application by deceiving the attacker and notifying the defenders that the attack has happened or is currently taking place. Deception technologies are useful when we want to detect that an attack or active reconnaissance is happening against a system. This helps incident responders and local CERT staff to screen a possible incident or to detect that a future incident might happen. A basic concept in deception technologies is a honeypot system which is a system that doesn't contain any real business data and whose main purpose is to collect intelligence and detect intrusions (Spitzner, 2003a). At the same time similar concepts such as honeytokens, honeynets and honeyfarms emerged that used the honeypot concept as a foundation. After (Spitzner, 2003b) defined HTs as a honeypot that is not a computer, there were a lot of different use cases for HTs in production systems and the problem of their generation was also researched by different scientists.

Published work by (Bercovitch et al., 2011) on an automated HTs generator called *HoneyGen* conducts data mining and extraction of characteristics and properties from real data items in order to generate an artificial relational database based on extracted rules. This method can generate artificial data items that can be planted into production resources in order to detect the unauthorized use of information. In a more specific use case, (Erguler, 2016) proposed a honeyword generation method by using existing user passwords as a solution for detecting password disclosures which was later improved by (Akshaya and Dhanabal, 2017) where graphical passwords were used to mitigate the ethic issues on using users real credentials. The research on generating honeywords is specifically used in securing passwords against disclosure and as an alarm system in case the honeyword is used as a users password.

Shown related work regarding HTs is used in very specific use cases. This means that generated HTs cannot be used in different situations other than for which they were designed. Main issue in HT generation is that HTs can be used in multiple places and, at this moment, to fully utilize them in the whole system, we need to use different methods of generation for different parts of the system.

In order to automate the implementations of HTs into applications and systems, we are introducing a conceptual model for generating HTs via NLP methods by using domain related content to enable the generation of HTs of any type with high probability of luring attackers and automated scanners but low probability of detecting which of the elements is an actual HT. By solving the problem of generating HTs that cannot be distinguished between real elements of the system, we would be one step closer to automate the process of implementing intrusion detection and prevention into an application or a system and raise the security level of every system by default along with escaping the cumbersome actions of manually implementing the same security defenses for each system separately.

The rest of this paper is organized as follows: firstly in section 2 we provide an introduction into HTs. Afterwards in section 3 we give an overview of NLP techniques related to the research at hand. In section 4 we present the developed conceptual model that can enable

us to use NLP techniques for generating HTs. Then we discuss the approach in section 5 and draw our conclusions in 6.

2 Honeytokens

The first usage of deception technology was described by (Stoll, 1988) where Clifford Stoll created an alert system and decoy files to gather information and identify a persistent hacker that accessed Lawrence Berkeley Laboratory systems. Afterwards, the concept of a honeypot emerged, which are specialized computers in a network that are not related to production systems. Honeypots are security components in a system designed to serve multiple purposes such as (1) luring attackers in order to alarm the administrators, (2) learning more about attackers behaviour and tactics, (3) stopping an attack or (4) recognizing new threats that are in the wild but still unknown to the public (Spitzner, 2003a). From a defensive perspective, honeypots can extend the knowledge of methods attackers use, and serve intelligence as a foundation for setting up defenses on production systems. Additionally, their implementation allows for automating the production system breach recovery process in case one of the production system gets attacked and infected prior to gathering intelligence via honeypots (I. Kim and M. Kim, 2012).

Spitzner (Spitzner, 2003b) also described a HT concept where he defined HTs as an implementation of a honeypot that is not a computer system. For example HTs can be anything from whole documents, URL parameters, database records, a sentence or a word in a legitimate business document. Basic usage for HTs is the same as for honeypots. You can use HTs as a decoy URL parameter to recognize that a service is being attacked and raise an alarm in the system, you can use them to analyze attackers tactics and methods, list a unique fake database record along with legitimate records to be able to recognize data leaks or disclosures, insert a unique fake sentence that does not change the meaning of a text to be able to recognize if a user has disclosed or leaked information or trace which profile was breached to steal data.

The definition of HTs for a specific system, can be hard and cumbersome manual work, since for every given application domain (AD) a new set of tokens has to be established and positioned in adequate places to serve the outlined purpose. In the following chapters we will try to address the first part of this process: how to generate honeytokens by using NLP techniques.

3 Natural Language Processing

NLP is a field of research that deals with various computational methods with the main objective how to make sense out of human language in order to be able to use it for various applications (Chowdhury, 2003). In particular, NLP uses various approaches for synonym and concept extraction, taxonomic or non-taxonomic relationship extraction, or even ontology construction, by building upon well established symbolic and statistical methods including but not limited to Lexico-Syntactic Patterns (LSPs), Hidden Markov Models (HMMs), Neural Networks (NNs), Support Vector Models (SVMs), Conditional Random Fields (CRFs), Compound Noun Information (CNI), various clustering techniques, co-occuring information, association rule mining, dependency triples, and of course machine– and more recently deep–learning (Liu et al., 2011).

One special technique we would like to outline here is the technique of concept/keyword extraction. In this technique, most relevant keywords from a given text are extracted, usually using statistical techniques with a few additions from lexical analysis (e.g. stop word, common word elimination, text vectorization, stemming and similar) as has been described in (Schatten, Seva, et al., 2015; Voinea and Schatten, 2014).

Another important technique is synonym extraction which we deem especially useful for the research at hand. Wordnet (Miller, 1995) is a lexical database for the English language that formally models various semantic and conceptual relationships between English words. Some of these relationships include synonyms and can be used to extract synonymous concepts (as has been done in (Schatten, Magdalenic, et al., 2011; Schatten, Ševa, et al., 2015), albeit for a different purpose).

By combining these two techniques, keyword extraction (function ke) and synonim extraction (relation se), one is able to construct a set of relevant keywords K from a given domain document D, as well as a superset K^+ of concepts synonymous to concepts in K by applying:

$$\begin{aligned} K &= ke(D) \\ K^+ &= \{s \mid \forall k \in K \wedge (k,s) \in se\} \end{aligned}$$

4 A Conceptual Model

Most important part in HT generation is to achieve several characteristics in order for the HT to serve its purpose. We propose generation of HTs which are related to the security part of the system but at the same time are mixed with the domain of the system where it is applied. If HT is a generated keyword related to the information security domain, they would be much more attractive to exploit than the standard generated domain keyword. On the other hand, if the generated HT is related only to information security but not to the domain, it could be exposed as a HT.

Therefore, every HT needs to have several properties:

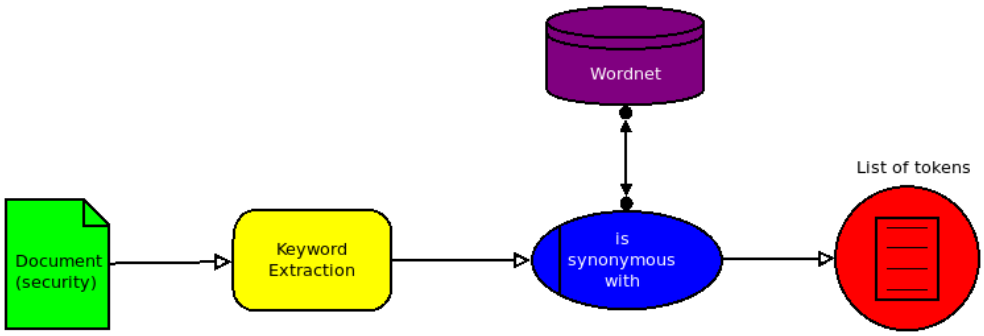

Figure 1: Flavor 1: Extraction of security related tokens

- HT should be attractive to exploit
- HT should be related to its domain
- HT should be adequate to its placement
- HT should not be easy to discover

In a HT environment, the established deception technology establishes one or more parameters (tokens) which in their nature are keywords related to the information security domain and/or the domain of the actual web system that is being protected with the HT. We propose two flavors of a method for generating such keyword parameters through the application of NLP techniques introduced in the previous section.

The first flavor is the straightforward extraction of tokens from the security AD, and generation of a synonymous token set. The next phase is usage of this set for establishing a HT. In particular, let D^{sec} be a document (or set of documents) dealing with a information security domain related to web systems. Further, let P^u be a set of usual parameters of the system being secured. The token set P^{sec} for the HT is then defined as:

$$P^{sec} = \{s \mid \forall k \in ke(D^{sec}) \wedge (k, s) \in se\}$$

Since P^{sec} and P^u might overlap, we define a mapping δ, that will exchange all keywords from P^u that are in P^{sec} with other, arbitrary words not in P^{sec}, therefore:

$$\begin{aligned} P^u_\delta &= \{\delta(k) \mid \forall k \in P^u \cap P^{sec}\} \\ &\cup \{k \mid \forall k \notin P^u \cap P^{sec}\} \end{aligned}$$

Figure 1 shows a graphical workflow of the proposed flavor.

The second flavor would use documents related to the actual AD of the web system being protected (these could be the actual web pages for this particular case), and try to find tokens that are either equal or synonymous to tokens extracted from the information security domain. This way, the set of tokens to be used would reflect the actual domain of the system. Let D^{dom} be a domain specific document (or set of documents), then the token set P^{dom} for the HT is defined as:

$$\begin{aligned} P^{dom} = \{s \mid & \\ & \forall \quad s \in ke(D^{dom}), \\ & \forall \quad s \in ke(D^{sec}) \\ & \wedge \quad (k, s) \in se\} \end{aligned}$$

Figure 2 shows the graphical work-flow for the second flavor.

5 Discussion

The proposed mechanism of automatically generating HTs via NLP would be the first step in automating the implementation of security defenses into an application or a system. In order for those security defenses to actually serve a purpose, they need to be generated in a way that the attacker or a scanner will not be able to distinguish between real data items or HTs and at the same time increase the chances of luring attackers into a trap. If we can generate HTs that meet these requirements, we will be able to increase the efficacy of intrusion detection and thus prevent possible attacks. Furthermore, we can use any system or application to gather intelligence on various tactics and methods that attackers use by implementing HTs as a security mechanism by default via automating the implementation of security defenses into systems.

To measure the effects of generated HTs, we can use different methods. For example, vulnerability scanners could be used as a tool to recognize vulnerabilities in the system by using different heuristics. Scanners can scan different elements of an application such as cookies and cookie flags, URL paths and parameters, HTTP headers, SSL certificates and similar. By using these elements as a placement for HTs, we could measure the scanners affinity towards certain parameters or data items and the probability of luring the scanners into actively exploiting the HT or recommending the HT as a potential attack vector. On the other hand, to measure the affinity of a human attacker towards certain elements of an application, we could conduct a case study in a Capture the Flag (CTF) competition where

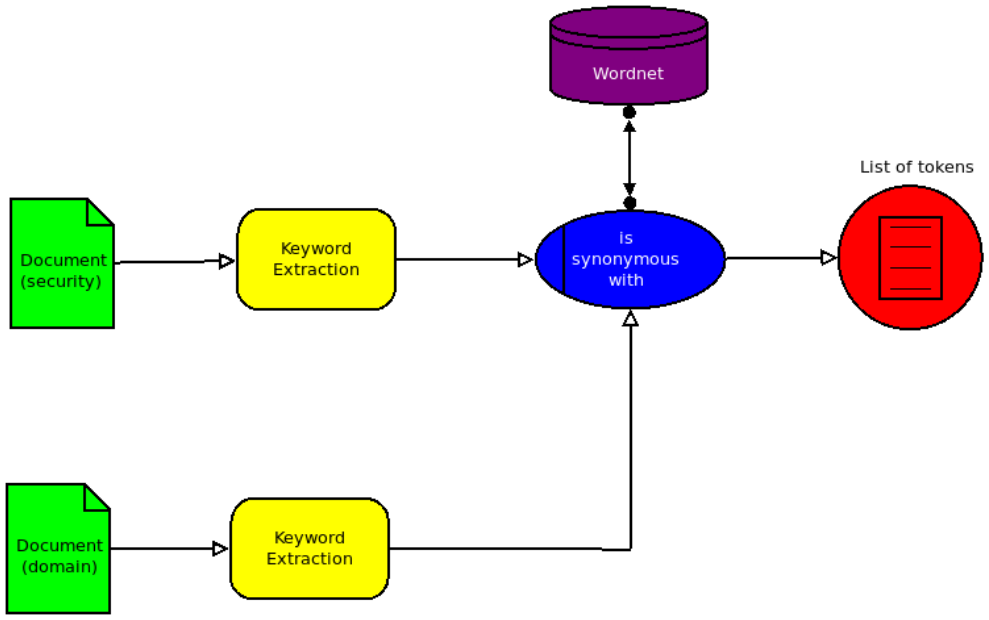

Figure 2: Flavor 2: Mapping of domain-specific to security related tokens

we could measure the affinity of the competitors towards certain elements in an application when trying to solve the challenge by analyzing their initial attack vector or how many vectors they tried to exploit before they started exploiting a HT. Additionally, we could measure if humans can recognize the difference between real data items and HTs by conducting a Turing test in order to see if there is a possibility to distinguish between items generated by a computer and items that are placed in the system for legitimate purposes.

6 Conclusion

The easiest way to increase the security of a system, without spending a lot of time in designing and implementing security defenses into an existing system, is to automate the process. One of the most convenient ways to implement intrusion detection is to use HTs which need to be generated in a way that look relevant to the domain and look attractive to exploit but at the same time do not raise suspicion that they are actually placed to lure the attacker. To solve this problem we propose two flavors for generating HTs by implementing extraction of security related tokens and mapping of domain-specific to security related token. To analyze the effectiveness of generated HTs we can use vulnerability scanners to analyze its affinity towards certain parameters or, to include the human element, conduct a case study by analyzing competitors behaviour during CTF competitions when solving challenges.

Our future research will be aimed at identifying possible interfaces to integrate the proposed approach into existing HT systems especially the placement of HTs in a given system. This task is non-trivial and depends on the system architecture, use-cases as well as best practices.

On the other hand, we will be testing different NLP techniques for generating tokens by using some of the outlined methods in order to find the most suitable ones for a wide range of systems. An open question is, would it be possible to generate phrases instead of, or in conjunction with, keywords?

In the end, we will try to implement an automated system described in the conceptual model that will make use of both placement and automated HTs generation techniques.

References

Akshaya, K. & Dhanabal, S. (2017). Achieving flatness from non-realistic honeywords. In *Innovations in information, embedded and communication systems (iciiecs), 2017 international conference on* (pp. 1–3). IEEE.

Bercovitch, M., Renford, M., Hasson, L., Shabtai, A., Rokach, L., & Elovici, Y. (2011). Honeygen: An automated honeytokens generator. In *Intelligence and security informatics (isi), 2011 ieee international conference on* (pp. 131–136). IEEE.

Chowdhury, G. G. (2003). Natural language processing. *Annual review of information science and technology*, *37*(1), 51–89.

Erguler, I. (2016). Achieving flatness: Selecting the honeywords from existing user passwords. *IEEE Transactions on Dependable and Secure Computing*, *13*(2), 284–295.

Kim, I. & Kim, M. (2012). Agent-based honeynet framework for protecting servers in campus networks. *IET Information Security*, *6*(3), 202–211.

Liu, K., Hogan, W. R., & Crowley, R. S. (2011). Natural language processing methods and systems for biomedical ontology learning. *Journal of biomedical informatics*, *44*(1), 163–179.

Miller, G. A. (1995). Wordnet: A lexical database for english. *Communications of the ACM*, *38*(11), 39–41.

Schatten, M., Magdalenic, I., & Vrdoljak, B. (2011). Towards ontology alignment of e-business standards using owl and f-logic. *International Journal of Metadata, Semantics and Ontologies*, *6*(3-4), 207–218.

Schatten, M., Seva, J., & Đuric, B. O. (2015). An introduction to social semantic web mining & big data analytics for political attitudes and mentalities research. *European Quarterly of Political Attitudes and Mentalities*, *4*(1), 40.

Schatten, M., Ševa, J., & Okreša-Đurić, B. (2015). Big data analytics and the social web: A tutorial for the social scientist. *European Quarterly of Political Attitudes and Mentalities*, *4*(3), 30–81.

Spitzner, L. (2003a). Honeypots: Catching the insider threat. In *Computer security applications conference, 2003. proceedings. 19th annual* (pp. 170–179). IEEE.

Spitzner, L. (2003b). *Honeytokens: The other honeypot*. SecurityFocus.

Stoll, C. (1988). Stalking the wily hacker. *Communications of the ACM*, *31*(5), 484–497.

Voinea, C. F. & Schatten, M. (2014). Recovering the past. eastern european web mining platforms for reconstructing political attitudes. In *European conference on political attitudes and mentalities ecpam*.