2014

Design requirements for generating deceptive content to protect document repositories

Ben Whitham
University of New South Wales, Canberra, Australia, b.whitham@student.adfa.edu.au

DOI: 10.4225/75/57a84b5cbefb9
15th Australian Information Warfare Conference, held on the 1-3 December, 2014 at Edith Cowan University, Joondalup Campus, Perth, Western Australia.

DESIGN REQUIREMENTS FOR GENERATING DECEPTIVE CONTENT TO PROTECT DOCUMENT REPOSITORIES

Ben Whitham
University of New South Wales, Canberra, Australia
b.whitham@student.adfa.edu.au

Abstract

For nearly 30 years, fake digital documents have been used to identify external intruders and malicious insider threats. Unfortunately, while fake files hold potential to assist in data theft detection, there is little evidence of their application outside of niche organisations and academic institutions. The barrier to wider adoption appears to be the difficulty in constructing deceptive content. The current generation of solutions principally: (1) use unrealistic random data; (2) output heavily formatted or specialised content, that is difficult to apply to other environments; (3) require users to manually build the content, which is not scalable, or (4) employ an existing production file, which creates a protection paradox. This paper introduces a set of requirements for generating automated fake file content: (1) enticing, (2) realistic, (3) minimise disruption, (4) adaptive, (5) scalable protective coverage, (6) minimise sensitive artefacts and copyright infringement, and (7) contain no distinguishable characteristics. These requirements have been drawn from literature on natural science, magical performances, human deceit, military operations, intrusion detection and previous fake file solutions. These requirements guide the design of an automated fake file content construction system, providing an opportunity for the next generation of solutions to find greater commercial application and widespread adoption.

Keywords
Fake files, decoy documents, honey-files, canary files, cyber deception

INTRODUCTION

Fake files, also referred to as honeytokens (Spitzner, 2003), honeyfiles (Yuill et al., 2004), digital decoys (Kushner, 2003), decoy files (Bowen et al., 2009), and canary files (Whitham, 2013a), is a cyber deception approach that holds potential to assist in data theft detection. There has been work on fake file processes and systems, however, research in the production of deceptive content has been limited to specific circumstances and problems, such as database fields (White and Thompson, 2006; Bercovitch et al., 2011), passwords and account details (Bojinov et al., 2010; Nikiforakis et al., 2011; Liu et al., 2012) and file attributes (Rowe 2004; Yuill et al., 2004; Bowen et al. 2009; Ben-Salem and Stolfo, 2011; Voris et al., 2012). In most cases the researcher hand crafted the deceptive content or reduced the problem to a set of defined fields. Several researchers have commented that automating deceptive content is either impossible, or much harder than it first appears, and migrating these handcrafted solutions to new content types or environments is likely to require significant effort (Rowe, 2004, White and Thompson, 2006).

The problem of automated content generation is very much in its infancy. In the military, deception is a long established art form; specialist staff appointments are supported with thousands of years of advice on how to build and run deceptive campaigns. In the field of cyber security, conversely, fake file researchers and practitioners cannot be confident in designing solutions without guidance on what constitutes success. This paper briefly reviews the area of fake files for deception and then presents seven requirements that fake file content generation techniques must address to be successful.

FAKE FILES

Clifford Stoll (2005) was the first to publish a description on the use of deception in the defence of computer networks. In August 1986, a persistent computer intruder attacked the Lawrence Berkeley Laboratory (LBL). Instead of trying to keep the intruder out, Stoll took the novel approach of allowing the intruder to access the system, baiting them with fake files to successfully capture the attacker's activities and trace the source.

There are a number of advantages of employing fake files. One of the key challenges to traditional approaches is the necessity to collect and track every action on the network. The massive amount of security data generated by network sensors and host-based applications can quickly overwhelm the operators charged with defending the network (Conti, 2006). By monitoring only fake files, intrusion detection teams can reduce the number of

documents that need to be observed, a similar approach to 'focused monitoring' (Roesch et al., 1999). When positioned and configured correctly, fake files can generate a very small number of high interest alerts (Ben-Salem and Stolfo, 2011). Unlike honeypots, fake files do not require additional hardware, or expose further potential software vulnerabilities. Fake files can be placed directly in file repositories, amongst the documents that need protection (Voris et al., 2013), rather than installed on a different IP address, collision domain, or network segment.

Fake files, like other forms of deception, can also create confusion and uncertainty regarding the value and location of critical information systems and resources (Tirenin and Faatz, 1999). The intruder's lack of knowledge of the file system makes it difficult to discern what truly belongs (White and Thompson, 2006). In most instances, if the attacker desires to read the contents of the fake document, their only option is to open the file and risk triggering an alarm (Yuill et al., 2004). This dilemma presents a combinatorial problem of differentiating the false from the real (Jackson and Hart, 2004), creating a deterrent and making the attackers cautious and uncertain.

FAKE FILE CONTENT GENERATION

While the placement and ability to track the fake file are important system design considerations (Bowen et al. 2009; Ben-Salem and Stolfo, 2011), the success of the fake file is largely due to the construct of its deceptive content (filename, file attributes and the actual document text). One of the challenges that Stoll faced was that the intruder's brief modem connections prevented telephone technicians from resolving the geographic location of the source more precisely than to a particular German city. Stoll baited the intruder by generating realistic and attractive content about how LBL was to support research on the controversial US military Strategic Defense Initiative (SDI). The files also contained a mailing list and a request form to obtain additional documents by mail. Not only did the intruder find the documents, they also remained logged into the terminal for several hours reading the text, allowing a successful trace. The content in the deception was so successful that the intruder also lodged the application form requesting the additional documents (Stoll, 2000).

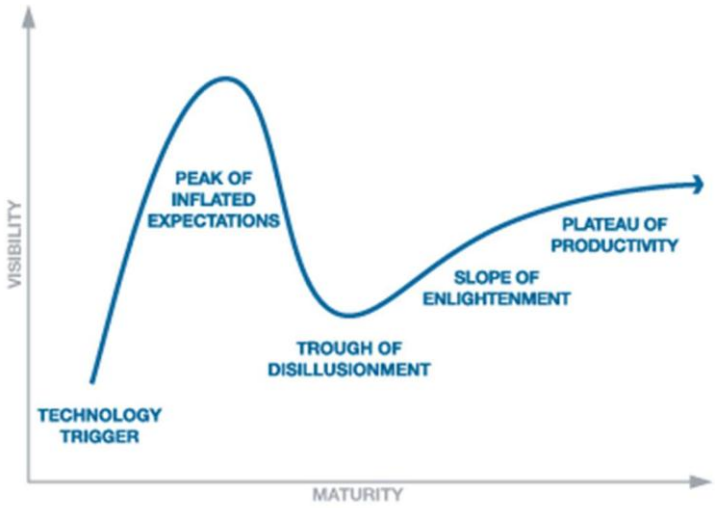

Figure 1: The Linden-Hype Cycle of Capability Maturity

Despite Stoll's success in the 1980s and the acclaimed potential of fake files to detect data theft, there has been little research in the area of deceptive content generation. Comparatively, the concepts of usage control, encryption and content tagging (watermarking), have evolved to commercial products that can allow a relatively non-technical user to specify complex rules to protect a variety of content in a range of environments. Gartner employs the Linden-Hype Cycle (Linden and Fenn, 2003), shown in figure 1, to differentiate the maturity of a product or capability. The early introductory phase of a solution is characterised by: (1) little to no adoption in the marketplace; and (2) the performance of products is poor outside of customised deployments. This description aligns with the current state of fake file systems. Only a single fake file solution is available on the Internet (Carey, 2013). Even this sole solution requires the user to supply the entire fake content; the system just manages distribution and detection. To date there are no public commercial fake file offerings.

The key barrier to fake files delivering on their expectations appears to be the burden of generating deceptive content. White and Thompson (White and Thompson, 2006) noted, "It takes a lot of effort to produce realistic

decoys... It took several months of research to locate all of the data that was needed to produce each field. This information also had to be processed into a usable form in order to incorporate it into the program. This is a very labour intensive task". Even a recent solution proposed by Wang, et al. (2014) requires a new software module to be developed for each topic.

REQUIREMENTS DEVELOPMENT

One of the first steps in designing a system to automate fake file content is to develop requirements. Requirements are a set of aspirational behaviours of the system (Sommerville and Kotonya, 1998). They inform the design of the solution and are essential components of both sequential design processes, such as the waterfall methodology (Benington, 1987; Bell and Thayer, 1976) and agile design, like Scrum (Schwaber, 1997). Requirements can be detailed, like software requirements specifications documents (Jackson 1995), or a list of guiding principles, such as the U.S. military's idioms for military deception: (1) focus, (2) objective, (3) centralised planning and control, (4) security, (5) timeliness, and (6) integration (United States Government, 2012). While ideas can be drawn from related areas, it is rarely possible, however, to enforce a complete set of requirements designed for one task to a related, but different problem (Nuseibeh and Easterbrook, 2000).

The remainder of the paper presents a set of seven requirements drawn from a review of deceptive literature in the fields of natural science, magical performance, human deceit and military operations. Much of the work on cyber deception has been drawn from developments in these fields. These fake file content generation requirements also build on recent work from other researchers who have considered the problem of delivering cyber deception from a system perspective.

REQUIREMENTS TO GOVERN THE GENERATION OF FAKE FILE CONTENT

Enticing

There are many examples of plants and animals successfully employing mimicry to entice and entrap prey. Figure 2 is a picture of the Venus Fly Trap, a carnivorous plant that emits a deceptive scent to attract its insect prey. The Anglerfish is another example. It has a fleshy outgrowth protruding from the fish's head, which acts as a fisherman's rod and lure.

Figure 2: The Venus Fly Trap emits an enticing scent to attract its insect prey

Enticement is also an effective military strategy. The Battle of Kasserine Pass took place during the Tunisia Campaign of World War II in February 1943. During the initial engagements, the U.S. forces were tricked into believing that the enemy was retreating. The German tank retirement was a ploy. When the German panzers reached the ambush point, with U.S. armour in hot pursuit, a screen of German anti-tank guns engaged, destroying nearly all the American tanks deployed to Tunisia. Without armoured support, the U.S. infantry were compelled to withdraw, and most of Tunisia fell into German hands (Watson 2006).

Constructing content that is desirable to data thieves is critical to the utility of a fake file (Salem and Stolfo, 2011). One of the limitations of fake files is that they can only detect data theft if a fake file is accessed by the perpetrator (Joshi and Sardana, 2011). Enticing a malicious user by advertising a list of passwords is likely to be more effective than mimicking a document containing the office social calendar.

The military approach to enticement is to devise a scenario in which the target is inclined to take action (Daniel and Herbig, 1982). In the context of fake files, this might involve matching the fake content to the desires of the attacker (Yuill et al., 2004; Bowen et al., 2009). However, this can be a difficult approach to achieve in practice; what is important for one threat actor, may not be important for another. For instance, an external data thief might seek immediately exploitable financial data, whereas a disgruntled insider might desire intellectual property.

A simpler approach, less dependent on the attacker, might concentrate on matching the fake file content with the sensitive data that requires protection. Sensitive data could be identified as part of an organisational threat and risk assessment to discover and appraise potential disruptions to the operation of the business. Identification of sensitive data might also be achieved dynamically through automated detection algorithms, such as those proposed by White and Panda (2011).

Realistic

Realism is critically important for deception (Haswell, 1985; Dewar, 1989; Latimer, 2003). During World War II, the Allies successfully delayed the reinforcement of German military forces at Normandy, through the use of physical dummies (see figure 2), logistical movements and simulated radio traffic (Perroni, 1984). The Allied forces organised a small team of people to simulate an entire radio network of military units. Great care was taken in ensuring authenticity including, the scripting of mistakes, confusions, unnecessary replications and transmission conflicts to ensure realism (Young and Stamp, 1990).

Figure 3: A realistic, inflatable tank (taken by an unknown photographer)

In 2000, a RAND study (Gerwehr and Rothenberg, 2000) conducted an experiment to test the effectiveness of deception against cyber adversaries. They concluded that realistic deceptions are more effective. That same year, Cohen (2000) independently argued that as a counterintelligence tool, deceptions could be of significant value, particularly if the artificial constructions were made to sufficient authenticity. His follow on study concluded that while simple deceptions are very effective at what they do, content must also be provided that survives scrutiny (Cohen and Thomas, 2001). Bowen, et al. (2009) used the term ‘believability’. Voris, et al (2011), expanded on Bowen’s idea, stating that, “upon inspection, a [fake document] should appear authentic and trustworthy. In the absence of any additional information, it should be impossible to discern a spurious decoy from authentic data” (Voris, et al., 2011, p6)

It takes significant effort to produce realistic deceptive content, even for highly formatted data (White and Thompson, 2006). Creating coherent sentences is even more challenging. The next generation of fake file content generation systems may need to conform with phonetic, morphological, lexical, syntactic, semantic and discourse conventions (Liddy, 1998; Feldman, 1999). They may need to appreciate Natural Language Processing factors, such as: author styles, sentence complexity, paragraph lengths, layout and mimicking embedded artefacts (Goldstein et al., 1999; Callaway and Lester, 2002). Realistic constructions may need to be aware of preventing repetitive prose, managing distance since the previous use against harvested statistical distributions (Elhadad and Robin, 1992; Stede, 1996). These systems may also need to ensure that the synthetic content is no

more or less legible than those documents that it is trying to protect, include grammatical peculiarities and spelling mistakes.

Minimise Disruption

Yuill, Zappe, Denning, and Feer (2004) proposed that fake files increase a network's internal security without adversely affecting normal operations. This is not necessarily true in all circumstances. For instance, realistic and enticing fake documents can pollute authentic data, confusing legitimate users or wasting their time (White and Thompson, 2006; Bowen et al., 2009). The insertion of fake data could skew the statistical properties of the data management system (White and Thompson, 2006). This bias can be very significant in certain applications (Adam and Worthmann, 1989).

Balancing the requirement to minimise the disruption to authentic users against other characteristics, such as believability, is one of the most critical aspects of any practical fake file system (Bowen et al., 2009). Thankfully, legitimate users should be very familiar with the content in the folders that they regularly access and will also utilise their system in fairly predictable ways (Bowen et al., 2009). Masqueraders, or automated processes running as the identity of a legitimate user, will have a limited knowledge of the file system, and may be more likely to conduct general content searches. These approaches need to be balanced against the requirements to entice malicious users, by employing indexable plain text content in the language used predominantly in the file system, and the desire to reduce distinguishable characteristics (discussed later).

Adaptive

Tirenin and Faatz (1999) argued that effective cyber deceptions must be dynamic in their implementation, presenting a continually changing situational picture to the enemy. In nature, deceptive adaptability provides distinct advantages to the user. For instance, figure 4 provides three time-sequenced pictures of the Octopus Vulgaris reacting to a potential predator by altering its colour, opacity, and reflectivity of the epidermis to blend in the background (Hanlon, 2007). Adaptability is also a critical component of a successful military deception, providing the flexibility to deal with unforeseen developments, and adjusting to a dynamic environment (Haswell, 1985; Latimer, 2003; Dickerson, 2003).

Figure 4: Octopus vulgaris reacting to a diver / predator (Hanlon, 2007)

Bowen, et al. (2009) included 'variability' in their set of system properties for decoy documents. They use the term to promote randomness in the initial generation in order to avoid detection. In 2004, Rowe (2004) published a prototype fake file generator (NFDir) that was able to construct fake files using the statistical averages of the target file system. Similarly, Kontaxis, Polychronakis and Keromytis (2014) developed a software prototype that produces fake network activity in response to a set of simple, single-keyword dictionary queries. The fake traffic generation process is conducted prior to the system becoming operational, as the training activity is computationally expensive. Variability, however, is not the same as adaptability. The limitation on both these approaches is that this process appears only to add variability on creation. Adaptability necessitates a continual feedback process to modify one or more aspects of the fake file during the term of its life, and to determine when the fake file is no longer necessary.

Yuill et al. (2004) presented a system to support the creation and management of fake files. They noted that operational systems change over time, and so too must fake files if they are to be believable. They proposed that the files' time attributes of the deceptive documents must be updated to maintain consistency with the file system. It is not just file names and time attributes that should be dynamic. Production file repositories routinely

add, modify and delete documents and organisations can change their foci. Maintaining realism requires other elements of fake file content to also keep pace with the changes occurring in the file system.

Scalable Protective Coverage

Coverage is a term used to quantify the field of view of a sensor or system of sensors (Dhillon and Chakrabarty, 2003). Lack of coverage, or 'scale insensitivity', is an intrusion detection systems design problem, limiting the cyber sensor's effectiveness to envelop a target environment (Mann, 1984; Roesch et al., 1999; Bass, 2000). Honeypots often face scalability and coverage challenges due to their fixed position within the network (Spitzner, 2003). In fake file systems, coverage may vary depending on the algorithm or processes used to generate the content, or the access method employed by the data thief. For instance, a fake file may not cover a user manually reviewing a specific directory, but provide suitable coverage against content searching. This paper defines these coverage constraints as horizontal (the inability to support a broad range of document content) and vertical (the inability to cover the volume of targets). These types are illustrated in figure 5.

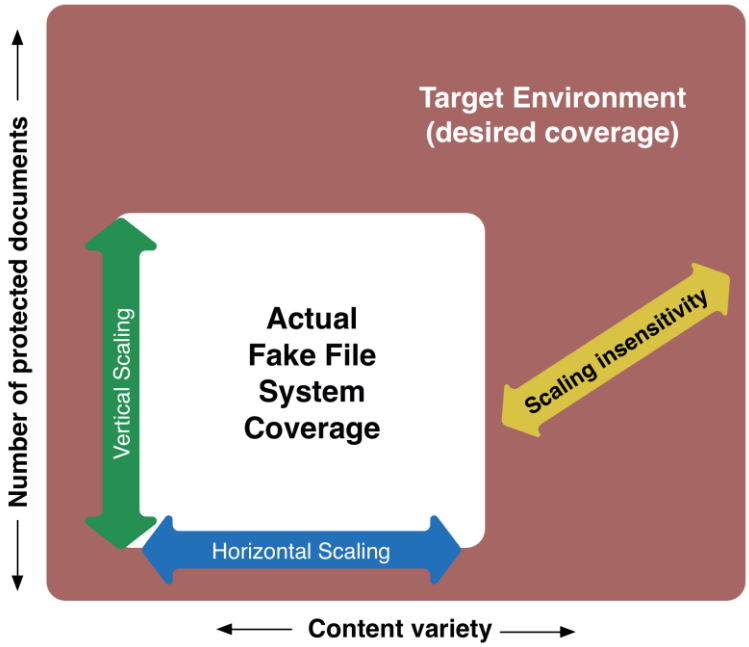

Figure 5: Vertical and horizontal scaling insensitivity

Horizontal scale insensitivity can manifest in a reduced set of support for languages, formatting styles and embedded artefacts, and inability to manage colloquialisms, abbreviations, misspellings, and new lexicography. For instance, the current generation of fake file systems largely built content based on fixed formats or templates (Stribling et al., 2005; White and Thompson, 2006; Bowen et al., 2009; Bojinov et al., 2010), which are likely to require significant investment in development resources to migrate.

Vertical scaling is also a highly desirable attribute. It is important for a fake file to protect the intended targets (Spafford and Zamboni, 2000). Typically, this means generating a fake file content that mimics one or more sensitive documents (Whitham, 2013b). Ideally, a fake file system should be able to produce as small a number of fake files as possible to cover the targets. Coverage of a single document is not ideal, as system owners can employ potentially simpler options, such as file watermarking. Conversely, deploying a large number of fake files across the environment is likely to result in a higher incidence of accidental false alarms and user disruption. The intended coverage area could unintentionally expand beyond the intended targets, covering documents with similar content, terminology, authorship style and audience (Manber et al., 1994; Heintze et al., 1996).

Minimise Sensitive Artefacts and Copyright Infringement

Voris, Boggs and Stolfo (2012) proposed a method of translating existing documents into a language not present on the file system. Their research showed great promise to address one of the weaknesses of fake documents, where false positives are generated due to accidental accesses by benign users, and improve the level of realism and association with the protected material. There are some limitations with the translation method. Firstly, these

fake file constructs are likely to contain sensitive artefacts from the documents that the system is attempting to protect, creating a protection paradox. Secondly, care should also be taken for the module implementation not to contradict national and international laws (Cenys et al., 2005). For instance, the proposal from Kushner (2003) to rebrand previously released films as new release titles in order to trap criminals downloading movies, may violate the copyright provisions associated with the decoy film. Care should be taken when drawing on content, such as names, numbers, discoveries, decisions and images within an organisation's file system to ensure that its whole or partial use does not violate privacy legislation or confidentially agreements.

Several experiments in the field of natural science have found that the relationship between realism and the effectiveness of the mimic is nonlinear, providing diminishing returns for added realism (Dittrich et al., 1993; Goodale and Sneddon, 1977). Moreover, mimics that seem imperfect to humans may actually appear as good mimics to other species (Dittrich et al., 1993; Cuthill and Bennett, 1993). White and Thompson (2006) also observed that one of the properties of good decoys was that they are realistic, at least at a distance. Bowen, et al. (2009) believed that in the case of fake documents, it is imperative that these files have realistic file names and modification dates lest even casual observation reveal their phony nature. In comparison, the believability of document text is of a lower priority. This is because an attacker would have already triggered an alert when opening the document by the time they analyse the content.

Threat actors frequently employ search tools to find content or particular types of files, and often this process is automated (Yuill et al., 2004). Bowen, et al. (2009) considered measuring frequently occurring search terms associated with major categories of interest (e.g., words or terms drawn from finance, medical information, intellectual property) and use these as the constituent words in fake documents. It may be possible to present the fake file as a genuine target to automated content based indexing technologies. Matching search ranking results with target content could deliver a suitable level of content 'enticingness', while avoiding the problems of direct extraction of complete sentences from the target documents. While an illegible document may be recognised by humans, the indexable text may perform adequately against these automated processes or causal human searches.

No Distinguishable Characteristics

Generation of fake files with uniquely identifiable characteristics can allow an attacker to develop detection capabilities to counter the deception. Producing convincing fake files requires careful planning because humans can recognise deceptive patterns (Rowe, 2005). This ability improves through continual exposure to deception (Mitchell and Thompson, 1986). The composition of synthetic text content should contain sufficient variability to avoid simple statistical analysis (Rowe, et al., 2004) and algorithms that can detect generated texts (Lavergne et al., 2008). Most importantly, the task of identifying a fake file should not be reducible to identifying a particular invariant that exists in all generated fake files, which can be used to develop a signature (Bowen et al., 2009). A single search or test function would thus easily distinguish the real from the fake, such as documents written in unusual languages, uncharacteristic formats or text that comprise random words, rather than structured sentences (Voris et al., 2012).

Honeypots face similar challenges. Network intruders can search for parameter values for the hardware devices, kernel or virtual system configuration settings and memory values (Holz and Raynal, 2005; Detristan et al., 2003). Anomaly-based detection can also be employed against deception systems, comparing the metrics of fake devices and data against those of typical computer systems (Rowe, 2006), inspecting randomly-chosen files or directories to see if they look 'normal' (Fu et al., 2006), checking content against other nearby material to determine if it is out of place (Lavoie and Krishnamoorthy, 2010), or employing timing tests (Zou and Cunningham, 2006). Rowe (2005) also highlighted that if an attacker is able to find material in the local drive that is nearly identical to what is written in a document, he or she may be able to conclude that the document in question was generated using content harvesting techniques. As previously discussed, the next generation of fake file content generation systems may need to appreciate factors such as: author styles and linguistic structures, sentence complexity, paragraph lengths, layout, numbers of embedded artefacts found in the texts they are attempting to mimic. These systems may also need to ensure that the synthetic content is no more or less legible than those documents that it is trying to protect, include grammatical peculiarities and spelling mistakes.

CONCLUSIONS AND FURTHER RESEARCH

Successfully detecting data breaches and the loss of electronic records remains a serious challenge. Deception approaches, namely the employment of fake files, has demonstrated value in generating uncertainty in the environment, and when positioned and configured correctly, can generate a very small number of high interest alerts. Even greater value may be obtained when paired with traditional security technologies.

The greatest barrier to producing a commercial fake file solution appears to be the ability to reliably and efficiently scale to mimic a range of data content. Several attempts to automate this process have been tried, largely through the application of fixed templates and/or handcrafted rules, which introduce complexity in management and prevent their portability.

This paper introduces a set of requirements for fake file content generation: (1) enticing, (2) realism, (3) minimise disruption, (4) adaptive, (5) scalable protective coverage, (6) minimise sensitive artefacts and copyright infringement, and (7) no distinguishable characteristics. The requirements have been developed from literature on natural science, magical performances, human deceit, military operations, intrusion detection and previous fake file constructions. These requirements will help aid research into deception effectiveness, and deception counter-measure investigations. In practice, formal requirements allow the development of systems that can be deployed to provide deception-based protection to real systems efficiently and effectively

The next stage of the research is to develop a set of quantifiable metrics, based on these seven requirements, to aid the comparison between different methods. This research will continue through the application of natural language processing to the problem of fake file content automation.

Future research could involve ranking the requirements by importance, mapped against desired outcomes, such as file system watermarking, deterrence or influencing an intruder or insider to perform a specific activity. There is also a clear gap in the automation of deceptive content.

REFERENCES

Adam N R and Worthmann J C (1989). Security-control methods for statistical databases: a comparative study, *ACM Computing Surveys (CSUR)* 21(4), 515–556.

Bass T (2000). Intrusion detection systems and multi-sensor data fusion, *Communications of the ACM* 43(4), 99–105.

Bell T E and Thayer T (1976). Software requirements: Are they really a problem?, *in Proceedings of the 2nd international conference on Software engineering*, IEEE Computer Society Press, pp. 61–68.

Benington H D (1987). Production of large computer programs, *in ICSE*, Vol. 87, pp. 299–310.

Ben-Salem and Stolfo S J (2011). Decoy document deployment for effective masquerade attack detection, *in Detection of Intrusions and Malware, and Vulnerability Assessment*, Springer, pp. 35–54.

Bercovitch M, Renford M, Hasson L, Shabtai A, Rokach L and Elovici Y (2011). HoneyGen: An automated honeytokens generator*, in Intelligence and Security Informatics (ISI), 2011 IEEE International Conference on*, IEEE, pp. 131–136.

Bojinov H, Bursztein E, Boyen X and Boneh D (2010). Kamouflage: Loss-resistant password management, *in Computer Security--ESORICS 2010*, Springer, pp. 286–302.

Bowen B, Hershkop S, Keromytis A, and Stolfo S (2009). Baiting inside attackers using decoy documents, *in Conference on Security and Privacy in Communication Networks*.

Callaway C B and Lester J C (2002). Narrative prose generation, *Artificial Intelligence* 139(2), 213–252.

Carey M (2013), 'Honeydocs', https://www.honeydocs.com/.

Cenys A, Rainys D, Radvilavicius L and Goranin N (2005). Implementation of Honeytoken Module In DBMS Oracle 9ir2 Enterprise Edition for Internal Malicious Activity Detection, *IEEE Computer Society's TC on Security and Privacy* pp. 1–13.

Cohen F (2000). A mathematical structure of simple defensive network deception, *in Computers & Security*, vol. 19, no. 6, pp. 520–528.

Cohen F, and Thomas E (2001). Red teaming experiments with deception technologies, taken from http://all.net/journal/deception/experiments/experiments.html.

Conti, G., Abdullah, K., Grizzard, J., Stasko, J., Copeland, J. A., Ahamad, M., ... & Lee, C. P. (2006). Countering security information overload through alert and packet visualization. *Computer Graphics and Applications, IEEE*, *26*(2), 60-70.

Cuthill I C and Bennett A T (1993). Mimicry and the eye of the beholder, *Proceedings of the Royal Society of London. Series B: Biological Sciences* 253(1337), 203–204.

Daniel D and Herbig K (1982). *Strategic Military Deception*, Pergamon Policy Studies on Security Affairs, Pergamon Press.

Detristan T, Ulenspiegel T, Malcom Y and Underduk M (2003), 'Polymorphic shell-code engine using spectrum analysis'.

Dewar M (1989). *The Art of Deception*, David and Charles Military Books.

Dhillon S S and Chakrabarty K (2003). *Sensor placement for effective coverage and surveillance in distributed sensor networks*, Vol. 3, IEEE.

Dickerson B D (2003). Adaptability-a new principle of war, Technical report, DTIC Document.

Dittrich W, Gilbert F, Green P, Mcgregor P and Grewcock D (1993). *Imperfect mimicry: a pigeon's perspective, Proceedings: Biological Sciences* pp. 195–200.

Elhadad M and Robin J (1992). Controlling content realization with functional unification grammars, *in Aspects of automated natural language generation*, Springer, pp. 89–104.

Feldman S (1999). NLP Meets the Jabberwocky: Natural Language Processing in Information Retrieval, *ONLINE-WESTON THEN WILTON* 23, 62–73.

Fu X, Yu W, Cheng D, Tan X, Streff K and Graham S (2006). On recognizing virtual honeypots and countermeasures, *in Dependable, Autonomic and Secure Computing*, 2nd IEEE International Symposium on, IEEE, pp. 211–218.

Gerwehr S, Weissler R and Rothenberg J (2000). Employing deception in information systems to thwart adversary reconnaissance-phase activities, *in National Defense Research Institute Project Memorandum PM-1124-NSA*, RAND.

Goldstein J, Kantrowitz M, Mittal V and Carbonell J (1999). Summarizing text documents: sentence selection and evaluation metrics, *in Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, ACM, pp. 121–128.

Goodale M and Sneddon I (1977). The effect of distastefulness of the model on the predation of artificial Batesian mimics, *Animal Behaviour* 25, 660–665.

Hanlon R (2007). Cephalopod dynamic camouflage, *Current Biology* 17(11), R400–R404.

Haswell J (1985). *The Tangled Web: The Art of Tactical and Strategic Deception*, John Goodchild Publishers.

Heintze N et al. (1996). Scalable document fingerprinting, *in 1996 USENIX workshop on electronic commerce*, Vol. 3.

Holz T and Raynal F (2005). Detecting honeypots and other suspicious environments, *in Information Assurance Workshop, 2005. IAW'05. Proceedings from the Sixth Annual IEEE SMC, IEEE*, pp. 29–36.

Jackson, M. (1995). *Software requirements and specifications* (p. 1). Reading, MA: Addison-Wesley.

Jackson, T., & Hart, D. (2004, April). Data decoys for confidentiality in a distributed computation: matrix multiplication. In *Proceedings of the 42nd annual Southeast regional conference* (pp. 307-308). ACM.

Joshi R and Sardana A (2011). *Honeypots: A New Paradigm to Information Security*, Science Publishers.

Kontaxis G, Polychronakis M and Keromytis A D (2014). Computational Decoys for Cloud Security, *in Secure Cloud Computing*, Springer, pp. 261–270.

Kushner D (2003). Digital decoys [fake MP3 song files to deter music pirating], *Spectrum*, IEEE 40(5), 27.

Latimer J (2003). *Deception in War: The Art of the Bluff, the Value of Deceit, and the Most Thrilling Episodes of Cunning in Military History, from the Trojan Horse to the Gulf War*, Overlook Press.

Lavergne T, Urvoy T and Yvon F (2008). Detecting Fake Content with Relative Entropy Scoring., *in PAN*.

Lavoie A and Krishnamoorthy M (2010). Algorithmic detection of computer generated text, *arXiv preprint arXiv:1008.0706*.

Liddy E D (1998). Enhanced text retrieval using natural language processing, *Bulletin of the American Society for Information Science and Technology* 24(4), 14–16.

Linden A and Fenn J (2003). Understanding Gartner's hype cycles, *Strategic Analysis Report No R-20-1971*. Gartner, Inc.

Liu B, Liu Z, Zhang J, Wei T and Zou W (2012). How many eyes are spying on your shared folders?, *in Proceedings of the 2012 ACM workshop on Privacy in the electronic society*, ACM, pp. 109–116.

Manber U et al. (1994). Finding Similar Files in a Large File System., *in Usenix Winter*, pp. 1–10.

Mann W C (1984). Discourse structures for text generation, *in Proceedings of the 10th International Conference on Computational Linguistics and 22nd annual meeting on Association for Computational Linguistics*, Association for Computational Linguistics, pp. 367–375.

Mitchell, R. W., & Thompson, N. S. (Eds.). (1986). *Deception: Perspectives on human and nonhuman deceit*. SUNY Press.

Nikiforakis N, Balduzzi M, Van Acker S, Joosen W and Balzarotti D (2011). Expos- ing the lack of privacy in file hosting services, *in Proceedings of the 4th USENIX conference on Large-scale exploits and emergent threats, LEET*, Vol. 11.

Nuseibeh B and Easterbrook S (2000). Requirements engineering: a roadmap, in *Proceedings of the Conference on the Future of Software Engineering*, ACM, pp. 35–46.

Perroni J (1984). *Operational Deception: The Key to Victory*, US Naval War College.

Roesch M et al. (1999). Snort: Lightweight Intrusion Detection for Networks., *in LISA*, Vol. 99, pp. 229–238.

Rowe N C (2004). A model of deception during cyber-attacks on information systems, *in Multi-Agent Security and Survivability, 2004 IEEE First Symposium on*, IEEE, pp. 21–30.

Rowe N C (2005). Automatic detection of fake file systems, *in International Conference on Intelligence Analysis Methods and Tools*.

Rowe N (2006). Measuring the effectiveness of honeypot counter-counterdeception, in *39th Hawaii International Conference on Systems Sciences*, Poipu, HI.

Schwaber, K. (1997). Scrum development process. In *Business Object Design and Implementation* (pp. 117-134). Springer London.

Sommerville I and Kotonya G (1998). *Requirements engineering: processes and techniques*, John Wiley & Sons, Inc.

Spafford E H and Zamboni D (2000). Intrusion detection using autonomous agents, *Computer networks* 34(4), 547–570.

Spitzner, L. (2003). Honeypots: Catching the insider threat. In *Computer Security Applications Conference, 2003. Proceedings. 19th Annual* (pp. 170-179). IEEE.

Stede M (1996). Lexical options in multilingual generation from a knowledge base, *in Trends in Natural Language Generation An Artificial Intelligence Perspective*, Springer, pp. 222–237.

Stoll C (1987). What do you feed a Trojan horse? *in Proceedings of the 10th National Computer Security Conference*, Baltimore, Md., pp. 21–24.

Stoll, C. (2005). *The cuckoo's egg: tracking a spy through the maze of computer espionage*. Simon and Schuster.

Stribling J, Aguayo D and Krohn M (2005). Rooter: A methodology for the typical unification of access points and redundancy, *Journal of Irreproducible Results* 49(3), 5.

Tirenin W and Faatz D (1999). A Concept for Strategic Cyber Defense, *in Military Communications Conference Proceedings*, MILCOM, p. 458-463.

United States Government (2012). Military Deception. Joint Publication 3-13.4.

Voris J, Boggs N and Stolfo S J (2012). Lost in Translation: Improving Decoy Documents via Automated Translation, *in Security and Privacy Workshops (SPW)*, 2012 IEEE Symposium on, IEEE, pp. 129–133.

Wang L, Li C, Tan Q, and Wang, X (2014). Generation and Distribution of Decoy Document System, *in Trustworthy Computing and Services*, Springer, pp 123-129

Watson, B. (2006). *Exit Rommel: The Tunisian Campaign, 1942-43*. Stackpole Books.

White J and Panda B (2009). Automatic Identification of Critical Data Items in a Database to Mitigate the Effects of Malicious Insiders, *in Information Systems Security*. Springer, pp. 208–221.

White J and Thompson D (2006). Using Synthetic Decoys to Digitally Watermark Personally-Identifying Data and to Promote Data Security., *in Security and Management*, pp. 91–99.

Whitham B (2013a). Canary Files: Generating Fake Files to Detect Critical Data Loss From Complex Computer Networks, *in The Second International Conference on Cyber Security, Cyber Warfare and Digital Forensic (CyberSec2013)*, The Society of Digital Information and Wireless Communication, pp. 170–179.

Whitham B (2013b). Automating the Generation of Fake Documents to Detect Network Intruders, *International Journal of Cyber-Security and Digital Forensics (IJCSDF)* 2(1), 103–118.

Young M and Stamp R (1990). *Trojan horses: deception operations in the Second World War*, Bodley Head.

Yuill J, Zappe M, Denning D and Feer F (2004). Honeyfiles: deceptive files for intrusion detection, *in Information Assurance Workshop, 2004. Proceedings from the Fifth Annual IEEE SMC*, IEEE, pp. 116–122.

Zou C C and Cunningham R (2006). Honeypot-aware advanced botnet construction and maintenance, *in Dependable Systems and Networks, 2006. DSN 2006. International Conference on*, IEEE, pp. 199–208.