Australian Government
Department of Defence
Defence Science and
Technology Organisation

Attribution of Spear Phishing Attacks: A Literature Survey

Van Nguyen

Cyber and Electronic Warfare Division

Defence Science and Technology Organisation

DSTO–TR–2865

ABSTRACT

Spear phishing involves the use of social engineering and contextual information to entice a targeted victim into unwitting leakage of sensitive information for purposes of identity crime or espionage. The high success rate together with the potential scale of damage caused by spear phishing attacks has motivated cyber researchers and practitioners to investigate more effective and strategic defensive, deterrent and offensive mechanisms against spear phishers. Obviously, the practicability of any such defence mechanism depends on the extent to which a defender has knowledge of the adversary behind a spear phishing attack. This necessitates the defending party to perform attribution in order to identify the spear phisher and/or his/her accomplice. In this survey, I broadly define attribution of spear phishing as any attempt to infer the identity or characteristics of the source of the attack which may include a machine, a human individual, or an organisation. Though highly desirable, this attribution mission is a very challenging task. This survey represents an initial step in this direction. Ultimately, the survey aims to sketch the landscape of attribution methods pertaining to spear phishing, as well as to provide constructive remarks and relevant recommendations for an organisation wishing to perform this attribution mission.

Published by

DSTO Defence Science and Technology Organisation
PO Box 1500
Edinburgh, South Australia 5111, Australia

Telephone: 1300 DEFENCE
Facsimile: (08) 7389 6567

AR No. 015-662
August, 2013

APPROVED FOR PUBLIC RELEASE

Attribution of Spear Phishing Attacks: A Literature Survey

Executive Summary

Spear phishing is an advanced form of cyber exploitation that targets and exploits the vulnerabilities of human users, often the weakest link in the security chain of a computer system, by means of social engineering. A typical attack of this type would involve an attacker contacting targeted victims via email, using the relevant contextual information and timing to trick them into divulging sensitive information. Spear phishing attacks have been aimed at individuals and companies, but also at government and defence organisations to exfiltrate classified data, as reported (c.f., [116]). The high success rate and the potentially significant damage caused by a spear phishing attack has motivated cyber researchers and practitioners to investigate a more effective but ambitious defence strategy: defending against the attacker, rather than defending against an attack. Having knowledge of the potential offenders allows an organisation to complement existing reactive, passive and tactical detection techniques with proactive and strategic approaches, and to potentially decrease the potency of successful attacks by holding those responsible for an attack legally and financially accountable, thereby deterring other potential offenders.

Solving the problem of spear phishing attribution is very desirable, albeit very challenging. This report represents an initial step in this direction by providing a survey of literature relevant to the attribution of spear phishing attacks. Based on a state-of-the-art offender profiling model, the report formulates the attribution problem of spear phishing where each component of an attack is viewed as a crime scene, and information available in each crime scene is identified and categorised into different types of evidence. Depending on the specific goal of attribution, and the type and amount of evidence available, different attribution methods and techniques can be performed. To this end, the report discusses at length, for each crime scene, the attribution methods potentially applicable to the different types of evidence. Since an extensive search for literature directly addressing attribution of spear phishing revealed very few relevant results, the report analyses spear phishing attribution in a larger context, and review relevant attribution methods in related disciplines. The taxonomy of evidence derived and discussed in the report is depicted in Figure 1.

In addition to reactive attribution methods, potentially much more information about the attack can be obtained via proactive/active attribution approaches. In this regard, the report also provides a discussion of the large scale realisation of proactive/active approaches by means of honeypots, decoy systems and field investigations. In many cases, however, the successful application of the mentioned methods may identify the accomplice (e.g., those responsible for individual components of a spear phishing attack), but not the ultimate spear phisher. To address this problem, the report discusses reconstruction of spear phishing attacks which involves forming logical conclusions based on the direct/indirect analyses of the pieces of evidence across the crime scenes as well as other sources of information, thereby enabling the attribution practitioners to obtain more complete information about the ultimate spear phisher. Toward characterisation of the ultimate adversary (in order to assist investigators to narrow down the list of real-life suspects), the report presents existing conventional offender profiling and cyber adversary characterisation methods that can be leveraged and adapted for use in the spear phishing

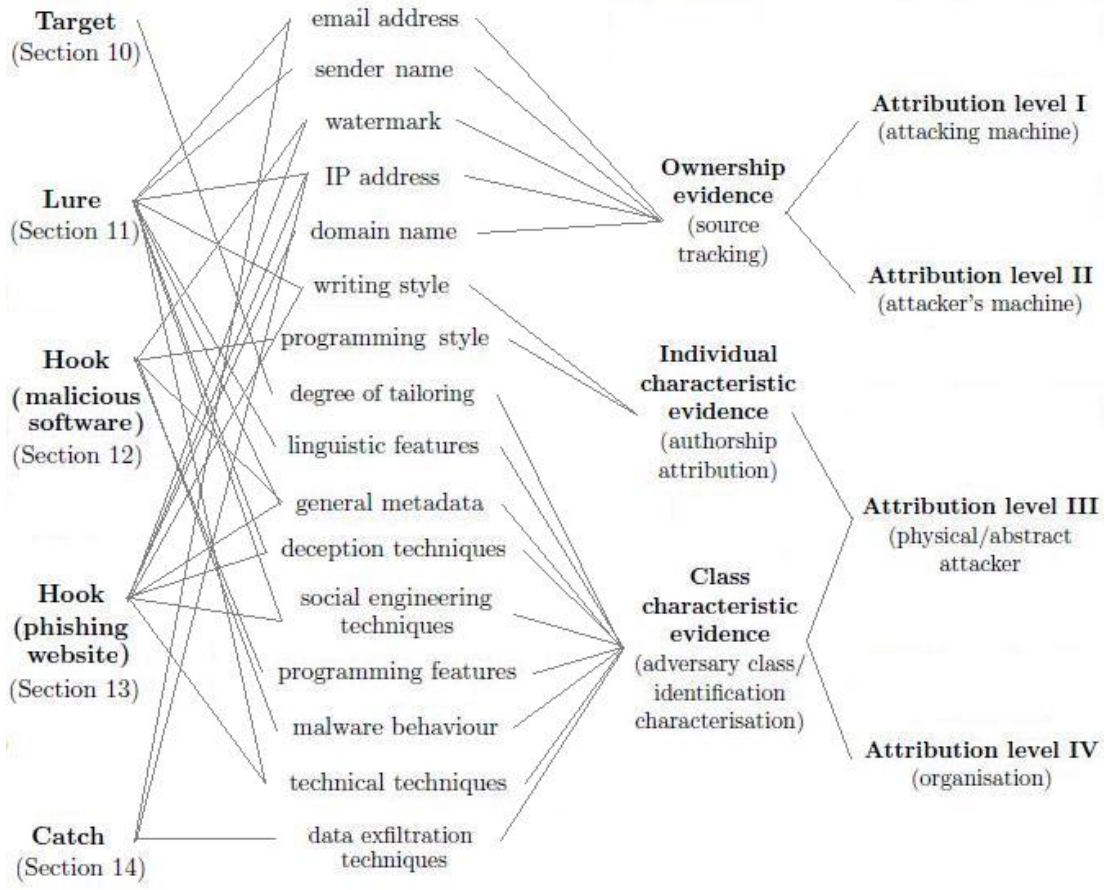


Figure 1: *A possible taxonomy of evidence pertaining to spear phishing attacks.*

context.

As it is indicated throughout the report, attribution results delivered by any of the methods discussed in the survey should be treated as suggestive rather than conclusive. Currently, due to various factors, it is not feasible to reliably pinpoint the exact attacker behind any single spear phishing attack using technical methods alone. As such, extant attribution methods should be applied either to characterise the attacker or to gain information about the attacker in order to narrow down the group of suspects. In general, attribution of spear phishing attacks in practice remains dependent on the goal and resources of the nation, organisation or individual concerned. To this end, the report provides a comprehensive survey of methods and techniques pertaining to varying levels of attribution which can potentially assist the organisation in undertaking the attribution mission.

Author

Van Nguyen
Cyber and Electronic Warfare Division

Van Nguyen joined DSTO in 2010 as a research scientist working in cyber security. Van obtained her PhD in 2010 from the University of South Australia, with a thesis on the implementation of membrane computing models on reconfigurable hardware.

THIS PAGE IS INTENTIONALLY BLANK

Contents

THIS PAGE IS INTENTIONALLY BLANK

1 Introduction

Spear phishing involves the use of social engineering and contextual information to entice a targeted victim into unwitting leakage of sensitive information for purposes of identity crime[1] or espionage. Due to its high success rate[2] and the potentially significant damage it can cause, spear phishing is a threat to the security of a nation and the general well-being of its people. This form of cyber exploitation is becoming a considerable concern for individuals, organisations and governments. The public and private sectors have responded by protecting against spear phishing in various ways. They have, for example, implemented/adopted technical solutions for the detection and prevention of attacks, generated/revised relevant policies, and organised spear phishing-aware user education programs. While these efforts promise to reduce the number of successful attacks, and thus to mitigate the negative impact of spear phishing, they constitute a merely passive and short-term defence strategy which might not be effective once the full potential of spear phishing is exerted. This concern, together with the potential scale of damage caused by spear phishing attacks, has motivated cyber researchers and practitioners to investigate a more effective but ambitious defence strategy: *defending against the attacker*, rather than *defending against an attack*.

But, *who is the attacker?* This question is at the essence of what is known as the *attribution problem*. Having different definitions across disciplines, attribution is in general concerned with inferring the cause or source actor of an action. In the specific context of spear phishing, having knowledge of the potential offenders allows an organisation to complement existing reactive, passive and tactical detection techniques with proactive and strategic approaches, and, if possible, to decrease the potency of successful attacks by holding those responsible for an attack legally and financially accountable, thereby deterring other potential offenders. Solving the problem of spear phishing attribution is very desirable. But at the same time, it presents many challenges, including technical, social and political challenges. Although these challenges render the attribution task remarkably difficult, this has not discouraged efforts toward finding a solution to this problem.

This survey represents an initial step in this direction. Herein, I discuss the attribution problem as it pertains to spear phishing. I also present a list of techniques, methods and approaches relevant to the attribution task which is presumed to be performed after a spear phishing email has been received and detected. This survey is not meant to be exhaustive, but rather is intended to serve as a starting point; it provides information and recommendations to an organisation wishing to undertake this mission.

It is assumed that readers already possess a basic knowledge of computer architecture and network security; consequently the survey does not include explanations of the miscellaneous technical concepts used. It is also acknowledged that, by an abuse of language, the terms *attack* and *attacker* are frequently used in discussions of spear phishing; I follow this usage at times hereafter. As the ultimate goal of spear phishing is the theft of data or information, not the interruption or disruption of the normal operations of a computer

[1] Identity crime refers to identity theft together with its associated crime such as money laundering and fraud against financial individuals and organisations.

[2] Spear phishing schemes have been shown to achieve a significantly higher response rate than blanket phishing attacks (see Section 2.2).

system, it is strictly *not* a kind of cyber attack — it is rather considered a kind of cyber exploitation or cyber espionage[3]. Nonetheless, this survey refers to many attribution methods and techniques discussed in the literature on cyber offences, including distributed denial of service (DDoS) attacks, network intrusion and malware exploitation. For the sake of simplicity, therefore, the term *cyber attack* is deliberately used in the survey to refer to various types of cyber offences, and *attacker* (or more general, *adversary*) is used to refer variously to offenders, intruders, exploiters, perpetrators and so forth.

A full presentation of the many thematic methods and techniques under discussion would be mathematically dense. To avoid confounding the reader with hard-to-follow details and thus deflecting the reader from the key concepts to be conveyed, the survey content is discussed at a conceptual level, in a way that is as intuitive as possible. When an inclusion of mathematical equations cannot be avoided, I will couple it with a conceptual explanation. On a related note, the terminologies used in different research areas are confusing (different terms used to indicate a single concept, or vice versa, a single term adopted for description of concepts with totally different meanings). Therefore, I will attempt to define the terminology used in the survey when the situation demands.

The survey is organised as follows. Section 2 presents the necessary background information relevant to spear phishing and the associated attribution problem together with some of its major issues and obstacles. Section 3 discusses the prospect of attribution of spear phishing attacks, while Sections 4, 5, 6 and 7 reviews foundational methods (i.e., those pertaining to authorship attribution, text categorisation and machine learning) relevant to attribution in the spear phishing context. Section 8 briefly reviews the literature of offender profiling, and presents how attribution of spear phishing as a problem is approached and addressed in this document. Section 9 formulates the attribution problem of spear phishing and broadly defines the scope of the survey. Section 10, 11, 12, 13 and 14 describe a wide range of techniques relevant to the attribution of the various components of a spear phishing scheme. In a different vein, Section 15 raises the importance of, and presents techniques pertaining to, proactive approaches to large-scale attribution of spear phishing attacks. This section is followed by a brief reflection (Section 16) and further discussion (Section 17) of the attribution methods presented in the previous sections. Finally, Section 18 describes offender profiling models in both the conventional and cyber contexts, before a short summary of the survey is given in Section 19[4].

[3]This perspective applies at the time of this writing. Due to the dynamic nature of cyber crime, this view of spear phishing may no longer hold true in the future.

[4]Thanks are due to Michael Docking, Richard Appleby, Olivier de Vel and Poh Lian Choong for their proofreading and feedback on the survey.

2 Background

2.1 Phishing

Named with reference to 'fishing' in the physical world, *phishing*[5] is a form of cyber exploitation that involves tricking a victim into volunteering sensitive credentials, such as credit card details and passwords, and then using the stolen credentials for financial gain. Phishing is distinguished from other types of cyber exploitation in that it *targets* and *exploits* the vulnerabilities of *human users*, often the weakest link in the security chain of a computer system, by means of *social engineering*.

A concept strongly tied to phishing nowadays, social engineering is in fact a branch of study in psychology and sociology that examines human nature and behaviour from the perspective of persuasion and influence. Social engineering has long been used in practice as an effective method to manipulate a human individual into performing an action or disclosing a desired piece of information, either by building up a relationship and false confidence with the person, or by exploiting the person's weaknesses. Social engineering is not a single technique, but rather a collection of techniques (see [267] for such a collection). These techniques are deliberately crafted to exploit certain traits that are inherent in human nature, for instance, *the desire to be helpful*, *a tendency to trust people*, *fear of getting into trouble*, and *willingness to cut corners* ([234] cited in [267]).

With respect to phishing, one has witnessed *technology-based social engineering* [267] — a new form of social engineering in which conventional techniques in social engineering are leveraged against a computer system, and are wholly or partly automated by means of technology. As such, phishing scams in the early days were carried out by only skillful hackers. However, due to specialisation of labour in the underground economy, phishing toolkits are now available at a reasonably low price, enabling almost every computer user, with technical expertise ranging from novice to proficient and with bad intent, to be capable of launching a phishing attack [143]. As a consequence, phishing attacks have proliferated and become a considerable threat to society. Efforts to protect against phishing have been made. However, as demonstrated in practice, innovation in devising new anti-phishing methods leads to the invention of new anti-detection mechanisms, resulting in an 'arms race' between phishers and their opponents. Economic and financial damage incurred by phishing is estimated by Moore [211] to be, at an absolute minimum, $320m per annum.

A typical phishing scheme consists of three components, namely the *Lure*, the *Hook* and the *Catch* [143]:

- **Lure**

  The lure component of a phishing scheme is often an email, which contains either a malicious attachment (*malware-based phishing*) or a link to a phishing website

[5] Hackers often replaces the letter *f* with the letters *ph* in a typed hacker dialect [143].

(*deception-based phishing*)[6]. The email appears to come from a legitimate sender and often contains a convincing story in order to 'lure' the victim to open the attached file (usually a PDF or a Microsoft Office document) or to visit the recommended website. For example, the email may purportedly trigger the user's curiosity by advertising the pornographic or politically controversial content of the attachment; or the email may urge the user to verify his/her account details on a bank website (whose mimicking link is provided in the email for user convenience), to avoid having his/her account being cancelled or compromised. A victim who fails to turn away from the *bait* (the seemingly interesting attachment or seemingly legitimate website) is likely to get hooked.

- **Hook**

  The hook refers to the malicious software embedded in the attachment or phishing website, which is designed to steal the victim's sensitive information. In malware-based phishing, the hook is typically spyware (e.g., a keylogger, screen capture, session hijacker, web trojan) or malware (e.g., hosts file poisoning, system reconfiguration and data theft) which is installed onto the victim's computer via drive-by-download or malicious attachment[7]. The spyware resides in the victim computer, monitors and captures the desired information, and sends the stolen information back to the phisher. In deception-based phishing, the hook rests in the phishing website and seizes sensitive information as it is being entered on the website[8]. To gain the trust of the victim, the phishing website mimics the look-and-feel of the legitimate website[9] using a wide range of web spoofing techniques, including the use of Javascript (to create the deceptive look-and-feel), convincing URLs (to create realistic-looking URLs, e.g., `www.paypal@150.44.134.189`) and homographs (to create deceptively similar URLs, e.g., `www.paypai.com` and `www.paypal.com`). Sensitive information entered on the phishing website is captured in different ways with varying degrees of sophistication, most advanced among which include *man-in-the-middle* and *man-in-the-browser* methods (please see [143] for more information about the mentioned methods). Once the information is obtained, it is ready for collection by a phisher.

- **Catch**

  The catch involves collecting the stolen information and using it for the benefit of the phishers. The stolen information can be collected directly (being sent back to the phisher, usually a web mail) or in batch (being stored at the server and collected by the phishers at some point). To minimise the chance of being caught, a very sophisticated catch is performed via a covert channel, such as relaying the stolen

[6]Other types of phishing that are not covered in this survey include: DNS-based phishing (*pharming*), content-injection phishing, man-in-the-middle phishing and search engine phishing (please refer to [143] for a detailed description of the different types of phishing).

[7]To keep the exposition simple, I refer to different types of malicious code as *malware* hereafter.

[8]In the early days of deception-based phishing, the hook also resided in an HTML form embedded in a phishing email.

[9]Diverse targeted websites of current phishing attacks include online auctions (eBay), payment sites (PayPal), share dealers (E*Trade), gambling websites (PartyPoker), social-networking sites (MySpace) and merchants (Amazon) [211].

information through public repositories (e.g., newsgroups, chat rooms or public forums). For example, the personal credentials can be embedded in an image with a secret code posted on a public forum. In this way, only the phisher can detect and download the information. The credentials are then either used to conduct illegal acts or are sold to other criminals.

Despite the existence of numerous anti-phishing tools, phishing is alive and undergoes continuous development in cyberspace. Phishing is growing in scope: originally an email-borne attack, phishing now utilises other communication channels, such as telephone via VoIP (*Vishing*), and SMS messages (*SMishing*). Phishing is also growing in sophistication, resulting in a more advanced, and more dangerous, form of phishing. This variation of phishing is referred to as *spear phishing*.

2.2 Spear phishing

One weakness of a phishing attack is that the same 'lure' is distributed to a large number of potential victims. As such, phishing emails are not very successful in reaching and convincing the victims — bulk distribution makes phishing emails relatively easy to detect and filter using an automated system, and the irrelevant information possibly contained in a phishing email often leads to its being ignored or deleted by the receiver. For instance, a phishing email that asks a receiver to perform some action to secure his/her ABC bank account would fail to persuade a XYZ account holder. Spear phishing is an attempt to remove this drawback. Unlike 'blanket' phishing (or simply 'phishing'), which takes an *opportunistic* approach — a phisher casts a net wide with a hope to hook some innocent or unlucky 'phish' — spear phishing takes on a *tailored* approach: the targeting and spearing of a specific 'phish'. With spear phishing, a phisher is willing to invest more effort and time into crafting his attacking scheme in order to maximise the likelihood of success.

Spear phishing inherits many features of phishing, but is much more powerful and efficient due to the incorporation of *contextual information* and *timing* into the phishing scheme. Hence spear phishing is also referred to as *context-aware phishing* [143] or *targeted phishing*. As an example for the purpose of illustration, if a person, *soon after* completing his/her registration for an ABC online account, receives a phishing email which seemingly comes from *ABC bank* and requests him/her to activate his *ABC online account* using the provided (malicious) link, this attack has a much higher chance of success than an attack aimed at a person who does not have an ABC account, or who registered for an ABC account some time ago, regardless of how technologically savvy the person is. A recent statistical figure shows that whereas the response rate for blanket phishing schemes is around 3-5%, the response rate for spear phishing schemes is as high as 80% [213]. This suggests that a spear phishing scheme, *if carefully crafted*, has a very high success rate and so, if a big enough 'phish' is selected, it can have a huge potential reward. Indeed, there is a new and most focused variant of spear phishing, namely *whaling*, which exclusively targets groups of high-level executives in an organisation.

Like traditional phishing attacks, spear phishing attacks include the three components Lure, Hook and Catch. However, the lure component in a spear phishing scheme is often

much more tailored to a specific victim; this necessitates the inclusion of an additional component. I refer to this additional component as *Target*.

- **Target**

  The first component to be executed in a spear phishing scheme, Target involves collecting information relevant to a selected individual so as to make a phishing scheme more convincing. Information about the victim can be directly obtained via an insider (e.g., an employee of an organisation associated with, or an acquaintance of, the victim), an untrusted/undercover outsider (e.g., a neighbour or a taxi driver with whom the victim has had a friendly chat), or a direct communication between a phisher and the victim (e.g., a foreign student showing an interest in the victim's work, or seeking a research scholarship in his institution). In the absence of these types of insiders and outsiders, user browsers, social network sites (e.g., Facebook[10], MySpace[11], Twitter[12], Friendster[13], Orkut[14] and LinkedIn[15]), public websites and public data repositories are fertile fields for a phisher to harvest information about the victim. For example, Jakobsson and colleagues [142] illustrated methods to collect user information from the web browser via the *Browser Recon Attack*; Griffith and Jakobsson [115] presented feasible approaches to obtain mother's maiden names (and other personal information stored in public repositories) of individuals; and Jagatic and colleagues [141] analysed possible techniques to gather information about individuals using social networks.

Due to its high efficacy, spear phishing is aimed not only at individuals, but also at organisations of various kinds. In an *individual spear phishing attack*, a phisher selects and tricks a victim by impersonating a person who is associated with the victim in order to commit financial fraud. In a *corporate spear phishing attack*, a phisher targets employees of a selected corporation by posing as a human resources or technical support person in order to gain access to the corporate network. The goal of the phisher in this type of attack is to steal money or to steal intellectual property of the corporation. More alarmingly, spear phishing attacks on government and defence organisations to exfiltrate classified data (c.f., [116]) have been reported. This form of spear phishing is referred to as *spear phishing as espionage*. According to [286], spear phishers and their alikes largely target employees with 'a high or medium ranking seniority' in attacking an organisation, and defence policy experts, diplomatic missions, and human rights activists and researchers in attacks on individuals.

While the economic damage caused by individual and corporate spear phishing can be estimated in terms of monetary cost, the consequence of the leaking of classified data related to national security to a hostile party — as could occur via the last form of spear phishing — could be immeasurable. As a result, instead of merely depending on passive, tactical and defensive techniques, one should complement them with active, strategic and

[10]Facebook. `http://www.facebook.com`.
[11]MySpace. `http://www.myspace.com`.
[12]Twitter. `http://www.twitter.com`.
[13]Friendster. `http://www.friendster.com`.
[14]Orkut. `http://www.orkut.com`.
[15]LinkedIn. `http://www.linkedin.com`.

possibly offensive methods. The efficacy of this vision is, at least in part, determined by knowledge about the potential attackers, for instance, *'Who could the attacker be?'*, *'What are the attacker's motives, goals and intentions?'*, or *'How much funding and other resources does the attacker have access to?'*. Research on *attribution of spear phishing* ultimately aims to help anti-phishing efforts find answers to one or more of these questions.

2.3 The attribution problem

There is no universal definition of the attribution problem in cyberspace. Due to the difficulty of the attribution problem, researchers and practitioners have been focused on addressing individual aspects of attribution, and have defined attribution accordingly. As a consequence, there is a large number of definitions of, as well as analyses relevant to, the attribution problem. At one end of the spectrum, attribution is specifically defined as 'determining the identity or location of an attacker or an attacker's intermediary' [308]. At the other end of the spectrum, attribution encompasses a very broad scope: determining the *modus operandi* of large-scale cyber attacks [288, 289].

Attribution is also studied at different levels. The four levels of attribution adapted from the categories identified by Dobitz and colleagues [77] are as follows.

- *Attribution level I* refers to attribution of an act to the attacking machines (which are often proxies relaying an attack between the attacker and his/her victim). This level serves as a starting point for other levels of attribution, and assists short-term defence against, and mitigation of, a cyber attack.
- *Attribution level II* refers to attribution of an act to the attackers' machines (or controlling machines). This level potentially provides significant information about the human attacker, and assists a longer-term defence against a cyber attack. It also allows for offensive response against the controlling machines.
- *Attribution level III* refers to attribution of an act to human attackers. I divide this level of attribution into two sub-categories as follows:
  - *attribution level IIIa* is concerned with attributing the act to the actual human attacker, which allows for responses through legal and diplomatic channels, and deters potential future attacks. This level of attribution requires a combination of technical and other intelligence approaches.
  - *attribution level IIIb* is concerned with attributing the act to a class of human attackers. No response is possible with this level of attribution, it however provides some insight about potential attackers and assists mid- and long-term defence against a cyber attack. Also, this level of attribution potentially supplies useful information for the next level of attribution that follows.
- *Attribution level IV* refers to attribution of an act to the organisation sponsoring the act. Again, this level of attribution requires a combination of technical and other intelligence approaches.

The source of an attack is also analysed from multiple perspectives. Dobitz and colleagues [77] identified three categories of cyber offenders: *individuals* (which includes recreational hackers, criminals, and political or religious activists), *groups* (which includes adversarial groups, organized crime groups and terrorist groups), and *states* (which includes rogue and developing nations). With respect to the level of state sponsorship, cyber offenders are classified into *no state affiliation*, *state allowed*, *state funded* and *state directed* [77]. According to Dr. Lawrence Gershwin, the U.S. National Intelligence Officer for Science, there are five categories of threat actors that threaten information systems: *hackers* (i.e., those engaged in attacks out of hobby and not having the tradecraft or motivation to pose a significant threat), *hacktivists* (i.e., those engaged in attacks for the purpose of propaganda), *industrial spies and organized crime groups* (i.e., those primarily motivated by money), *terrorists* (i.e., those still largely resorted to conventional attack methods such as bombs), and *national governments or nation states* (i.e., those with the resources and time-horizon to cause significant damage to critical infrastructure) [3].

> In this survey, I broadly define attribution of spear phishing as *any attempt to infer the identity or characteristics of the source of a spear phishing attack* where the source of an attack can be any of the entities defined in the four levels of attribution above[16]. This definition has been chosen deliberately to encompass the diversity of work on attribution that is potentially applicable to different aspects of spear phishing.

Regardless of how the attribution problem is defined, however, there are unavoidable obstacles that any attempt to attribute a spear phishing attack in particular, and a cyber offence in general, must face. The next section gives an overview of such obstacles.

2.4 The challenge of attribution

Attribution of a conventional crime is a very difficult task. Attribution of a cyber crime is often even harder, since many relevant standards in the physical world do not apply in cyberspace[17]. Attribution of spear phishing is not an exception in this regard — it faces many obstacles. These obstacles are roughly grouped into different categories as follows.

2.4.1 Technical obstacle

One way to attribute a spear phishing attack is to determine the source of the phishing email (i.e., the email address of the account from which the phishing email is sent, or the IP address of the machine from which the phishing message originates). Unfortunately, efforts to identify the source of a phishing message are greatly hindered by anonymity — a notorious, but in some respects very desirable, feature of the Internet. The Simple Message

[16] For the sake of simplicity, throughout the survey I address the originator of a given attack in a singular form, i.e., there is *an* attacker, or *an* organisation (group) responsible for a given attack, and other individuals involved in the commission of the attack are considered his/her accomplices.

[17] The challenge of attribution is much discussed in the literature. Please refer to [56, 134, 62] for a deeper discussion of this topic.

Transmission protocol (SMTP) for transferring emails and the destination-oriented routing mechanism of the internet for transporting network packets do not use, and thus do not verify, the source address of messages they receive and transfer. This allows an attacker to spoof a source IP or email address at will. In the case of IP address spoofing, the true IP address can be, in principle, recovered by *backtracking* (or *traceback*): reconstructing the path of a message, starting from the victim and progressing toward the attacker. However, the distributed management of the internet, which allows each network to be run by its owner in accordance with a local policy, and the use of network address translation (NAT) devices which hide the true IP addresses of the machines on the network, together with the stateless message routing protocols, the use of dynamic IP addresses and so forth, all present hurdles for determining the origin of an internet message. Greater hurdles are set up when 'stepping stones' are utilised: for instance, email messages are sent via an open relay, attacks are cascaded through a series of intermediary hosts, using software such as *SSH*, *Telnet* and *rlogin*. These hurdles become even more formidable if an attack is relayed through a zombie-net or an autonomous system. A zombie-net is a (potentially very large) number of compromised machines under the control of a remote attacker who either owns or rents the zombie-net. In contrast, autonomous systems (e.g., Tor) are intentionally designed to provide anonymity and privacy to legal users (e.g., intelligence agencies, activists and dissidents), but have unintentionally served as stepping stone platforms for the hiding of the identity of criminals. There are thousands of compromised hosts in a zombie-net and up to 800 distributed servers in Tor around the world [118]. An internet message relayed through machines in these systems would be very difficult to traceback to its source.

The difficulty of the attribution problem is escalating further when zombies are used as proxies to back-end machines (or mothership) accommodated by rogue networks owned by Internet companies and service providers affiliated with criminal organisations. Such rogue networks are responsible for a range of malicious activities ranging from (i) providing bullet-proof hosting services[18] which are often used to serve exploits and malware for the purpose of phishing/spear phishing [156], to (ii) sending unsolicited emails and hosting phishing websites. Examples of rogue networks include the Russian Business Network (c.f., [26] and [171]) and more recently the US-based company Atrivo (c.f., [10] and [172]). When these rogue networks are used to serve exploits and malware (known as malware networks), the attackers can implement a sophisticated command and control infrastructure between the command and control servers and the zombies, which make mitigation, prevention and attribution of targeted malware-based attacks very challenging. As documented in [33], one such malware network implements multiple layers of control. The first layer of control used blogs, newsgroups, and social networking services (e.g., Twitter, Google Groups[19], Google Blog[20], and Blog.com[21]) as means of direct and persistent control of zombie machines. When zombie machines accessed these services, they were informed of, and then received commands from, servers in the second layer of control which are often located in free web hosting providers. When the command and control servers in the second layer were 'taken-down', zombie machines would receive commands from the social

[18] Bullet-proof hosting services refer to the services that continue to persist even after the hosted resources are found to be malicious or illegal.

[19] Google Group. `http:www.groups.google.com`.

[20] Google Blog. `http://www.googleblog.blogspot.com`.

[21] Blog.com. `http://www.blog.com`.

networking layers in order to establish a connection to the dedicated and very stable servers in the third layer of control located in the People's Republic of China (PRC). Not only does such a rogue network contribute to undermining efforts to defend against advanced malware-based attacks, but they also perplex any attribution attempt.

2.4.2 Social obstacle

Various ideas about how to add attribution capability to the internet infrastructure have been proposed. However, it is commonly believed that implementing such an idea on a global scale would be extraordinarily difficult due to privacy concerns. Furthermore, attribution capabilities, even if implemented in practice, could be misused by non-democratic governments to facilitate human right abuse and to suppress freedom of speech. To obtain user acceptance of attribution, it is important not to totally relinquish anonymity, but rather to achieve an appropriate balance between anonymity and attribution — a great challenge in itself.

2.4.3 Political obstacle

Due to the global nature of cyberspace, attacks are often cross-border and cross-country, so performing attribution of a cyberspace attack is likely to require international cooperation. It could be very hard to convince a foreign state to cooperate, especially if that foreign state is a hostile party of, or in political conflict with, the requesting state.

2.4.4 Legal obstacle

Cyberspace attacks are also cross-jurisdiction. Achieving successful collaboration between jurisdictional systems in order to attribute these attacks is by no means simple — many jurisdictions do not have adequate cyber law; more seriously, some jurisdictions even support cyber gangs for nefarious purposes.

2.4.5 Economical obstacle

Monetary cost associated with implementing attribution technologies causes various parties to hesitate. From the perspective of technology users, many of the users are not willing to bear the cost of attribution investment — a tangible cost for an 'intangible' and distributed benefit. From the perspective of technology manufacturers, many of them are cautious about increasing the cost of a product to add an attribution capability (which is required by only a small portion of the market) and risking the loss of market share to its competitors who offer similar products (without attribution, of course) at a lower price.

2.4.6 Psychological obstacle

Last, but most importantly is *conscious deception*, a phenomenon which, to various extents, is involved in almost every cyberspace offence. If deception techniques are to be

exhaustively exploited, they can potentially offset all attribution efforts. In the presence of deception, it is possible for technologically and scientifically accurate attribution results to be totally incorrect with respect to revealing the *truth* behind an attack. For instance, even if many pieces of evidence indicate that a nation is the source of an attack, one should abstain from making any definitive judgment on the basis of this evidence because the evidence might simply be the product of a deceptive scheme to put blame on that nation (*false flag*).

3 Spear phishing attribution problem

In light of the difficulties presented in the previous section, what is the outlook for attribution of spear phishing? Some might argue that spear phishing attribution is essentially an infeasible task and has no practical usefulness. However, though it is true that the practicality of attribution of spear phishing needs to be studied and verified, this point of view is perhaps over-pessimistic. Indeed, as I will now attempt to show, attribution of spear phishing is an important task and worth pursuing, regardless of the presence of numerous obstacles some of which may never be possible to overcome.

- As an advanced form of cyber attack, spear phishing inevitably presents significant hurdles to any attribution effort. However, by the same token, the complex nature of an attack of this type often leaves ample opportunity for one to recover, at least in part, the digital trail left behind by the attacker. For instance, many of the challenges of attribution presented above are commonly discussed in the context of distributed denial of service attack (DDoS) whose tangible evidence, observable by the victim, is only a large amount of useless traffic. In contrast, a typical spear phishing attack provides far more substantial evidence: it usually consists of multiple components (i.e., Target, Lure, Hook and Catch) each of which potentially reveals a *signature*, *writeprint*, *thumbprint*, characteristics or other information pertaining to the attacker (as the reader will see in later sections of the survey).

- In a focused view, attribution involves identifying the exact (single or collective) individual associated with a spear phishing attack in order to bring a lawsuit against that individual. This requires a piece of evidence that is brought against a suspect to achieve a sufficient degree of certainty, that is, *beyond reasonable doubt*. In other words, the evidence must be of 'forensic quality' [56]. This attribution task can be extremely difficult. In a broader view, national security agencies are concerned with *advanced persistent threats* (APTs)[22], and interested in obtaining intelligence information. In this scenario, attribution information that allows the inference of any knowledge about the criminal (e.g., motive, intent and characteristics) behind a spear phishing attack is valuable. For instance, one may infer from an attribution result stating *the spear phishing scheme is highly-crafted and directed at the defence department with the goal of exfiltrating classified data* that the attacker is more likely to be a hostile party or a nation state who desires political advantage, rather than a recreational hacker or a script kiddie who desires fame or personal enjoyment.

- IP spoofing and anonymity is commonly thought of as being one of the biggest hindrances to the prospect of tracking. However, only cyberspace attacks that are primarily designed for one-way communication (e.g., a DDoS attack whose only goal is to flood the target with useless traffic) may plausibly use only invalid IP addresses. Spear phishing, as well as other types of identity theft and espionage, must support two-way communication. Therefore, even though IP spoofing is a technique that is predominantly utilised in spear phishing, it is usually the case that there is at least one step in the attack involving the use of a valid IP address: the step when the

[22] APTs refers to orchestrated activities to gather intelligence on particular individuals or institutions [33].

stolen information is downloaded and accessed by the phisher. A tracking method that checks this step can potentially reach the phisher or his/her accomplice.

- Due to deception and other factors, it is often the case that a single method will not suffice for the attribution task in question. However, if one is assisted by a variety of attribution techniques, an integral implementation of these techniques can mitigate the weaknesses of each individual technique, strengthen inference, and thus increase the significance level of the result that is obtained.[23]

As discussed above, it seems unlikely that there exists a universal end-to-end attribution method for spear phishing, and likely that a feasible solution will come from efforts to integrate the individual attribution techniques that are available. Interestingly but not surprisingly, however, an extensive search for literature directly addressing attribution of spear phishing reveals very few relevant results — perhaps this is partly due to the fact that spear phishing is a relatively new form of exploitation. The lack of relevant search results does not necessarily reflect a lack of methods and techniques, but it does indicate the necessity of analysing attribution of spear phishing in a larger context and speculating on what related domains may offer to assist in the carrying out of this task. To this end, spear phishing attribution finds itself at the intersection of diverse research disciplines. Directly relevant are attribution techniques for cyber attacks (which includes subfields such as attribution of DDoS, attribution of network intrusion, and attribution of spam/phishing attacks), as well as techniques in email and software forensics. Outside the cyberspace realm are authorship attribution (a classical field in literary studies and linguistics) and criminal profiling. With respect to the computational aspect of attribution, the fields of data mining and machine learning offer a wealth of techniques concerned with collection of data and automation of analysis processes. It is obviously advantageous for an attribution task to capitalise on existing approaches. But the staggeringly large number of available heterogeneous/homogeneous and complementary/alternative attribution-related techniques, from a wide range of research fields, is potentially bewildering, and leaves an attribution practitioner facing difficult decisions regarding which attribution techniques to choose for which parts of a spear phishing attack, as well as how to interrelate/integrate the attribution results from the different parts in order to apprehend the ultimate threat actor. This mandates that practitioners adhere to a *sound* methodology in carrying out the attribution task. Section 8 in the survey discusses such a methodology.

[23]The principle '*garbage in, garbage out*' applies here, of course: a combined attribution result is not always more reliable than any of its constituent individual results. It is therefore critical for the set of techniques to be chosen well, and for the obtained results to be combined and interpreted with care.

4 Review of foundational methods relevant to the attribution of spear phishing

The majority of the attribution methods pertaining to spear phishing are indeed new incarnations of old theories. As the reader will observe below, various pieces of spear phishing evidence are wholly or partly presented in a textual form (e.g., a phishing email, a phishing website, or a malicious piece of source code); methods to determine the author behind such evidence heavily draw on the concepts, theories and empirical analyses accumulated in a well-established research discipline called *authorship attribution*, as well as its related area *text categorisation*. At the same time, computational techniques for attributional analysis, and for constructing an attacker profile, would make use of well-studied classification and clustering methods in *statistics* and *machine learning*.

To avoid perplexing readers with scattered and out-of-context descriptions of the relevant methods offered by the mentioned research disciplines, the next three sections are devoted to a discussion of the methods in a big picture of their respective research field and in a coherent manner. The methods collectively represent a foundation from which many attribution methods applicable to spear phishing are stemmed. The sections are also intended to serve as a reference point to help understand the attribution techniques presented in the subsequent sections of the survey.

5 Authorship attribution

This section presents the fundamentals of authorship attribution in the realm of natural languages. An introduction to authorship attribution will be presented, followed by a brief review of representative techniques in the literature.

5.1 Introduction to authorship attribution

With or without being aware of it, source checking is what one usually does before reading a piece of text. This is due to the fact that knowledge of source influences one's thought, respect and judgement about the text being read. In most circumstances, information about the source is obtained directly (*exogenous* information), e.g., it is displayed on the cover of a book, embedded in the header of an email, verbally informed by another person, or easily ascertained via the handwriting of the text. In a few specific cases, this information is not available, but a reader usually wishes to have knowledge about the text's author. Supposing that the person has an *anonymous* text — and that is all the evidence he or she possesses — can the person attribute the text to its source? This question and its possible answer are fundamentally the motivation and goal of authorship attribution. The assumption behind author attribution is that a piece of text being anonymous does not necessarily mean that it is *untraceable*. In brief, author attribution aims to leverage the characteristics intrinsic to a piece of text (*endogenous* information) in order to derive the identity or characteristics of its author.

Authorship attribution is difficult. The challenge of this topic is reflected in the fact that author attribution has a very long history — it has been extensively investigated throughout the past two centuries — but until now a consensus regarding the best techniques to use has not emerged [147]. Despite the tremendous amount of research and a lack of standards, author attribution continuously develops along with the evolution of language; nowadays it still receives considerable attention. Historically, author attribution was first examined in literary studies and linguistics as a specialised branch of *stylometry*, an area which studies variations in language and measurement of linguistic styles. At present, authorship attribution may be regarded as an overlapping of stylometry and *text categorisation* (see Section 6). While authorship attribution in traditional settings is interested in associating an author's life and mindset with his writing — '[t]he author still reigns, in histories of literature, biographies of writers, interviews, magazines, as in the very consciousness of men of letters anxious to unite their person and their work through diaries and memoirs' [17] and 'the explanation of a text is sought in the person who produced it' [17] — authorship attribution in modern settings mostly focuses on seeking accountability on the part of those who are authors of textual pieces of evidence, e.g., as witnessed in cases involving plagiarism, intelligence, criminal and civil law, and computer forensics.

Irrespective of the differences in goal and motivation, applications of authorship attribution rely on a common set of methods which are described below.

5.2 Techniques for authorship attribution

A traditional (*qualitative*) approach to address the problem of authorship dispute in the old times was based on knowledge and judgement of a human-expert (*human-expert-based methods*). It was not until the late eighteenth century that computational analysis of writing style was first attempted; since then study of authorship in a computational manner has been ceaselessly developing. Attribution methods proposed in such settings collectively constitute the subject matter of *computational/quantitative authorship attribution*, or *authorship attribution*, nowadays.

Human language is highly complex. The complexity, creativity, flexibility and adaptability of human language makes it already a challenge to directly examine text to determine its authors based on the expertise and experience of a human expert. In order to operationalise these analysis procedures, it is necessary to have a non-conventional model of text. To this end, (quantitative) authorship attribution is grounded on a simple model in which text, instead of being perceived as a means to communicate passion, ideas and information, is viewed simply as a sequence of tokens (e.g., words) governed by certain rules. Within this simple model, a sufficient degree of variations of language usage is captured, for instance, a token can have different properties, different lengths, and can be instantiated to different values; or, tokens can be grouped in a variety of ways and their values have irregular distribution in the text. This band of variation allows for human choice in using the language to be measured.

Given the model described, numerous ways to measure human writing styles are, at their most general sense, instantiations of an abstract procedure which consists of the two following phases:

- **Feature selection**: the first step in authorship analysis is the identification of the feature set of interest. This step involves selecting features as textual measurements (e.g., average word-length, sentence-length, word distribution) whose values can distinguish between different authors, and compute values for the measurement for each piece of text, including the anonymous text — this, in effect, transforms every piece of text in consideration to a *vector of features*, and
- **Attributional analysis**: the space of feature vectors that results from the feature selection phase is processed in some way (e.g., by computing similarity or distance among the vectors) to associate an anonymous text to an author.

There exists a number of surveys relevant to authorship attribution in the literature. Joula [147] published an excellent survey on authorship attribution which currently serves as a reference for a large amount of work in the field. Grieve [114] provides a comparative assessment of different features for attribution efficiency for traditional text. Stamatatos [279] conducted a comprehensive survey on authorship attribution methods in modern settings. A review of authorship attribution is also included in a paper by Koppel and colleagues [168]. The discussion that follows is loosely based on the mentioned work.

5.2.1 Feature selection

A main goal of this phase is to select features that have values consistent across a collection of text by an author, but varying across pieces of text by different authors. Proposals for feature selection to date include:

- features based on (i) *lexicon* (word, word properties or sentence), (ii) *characters* (such as graphemes(*A*, *B*, *C*, ...) , digits(*1*, *2*, *3*, ...), whitespace(‘ ’) , symbol (*#*, *&*, ***, ...) and punctuation marks (*.*, *?*, *,*, ...)), (iii) *syntax* (colocation of words, adjective phrases, adverb phrases, ...), and (iv) *semantics* (meaning conveyed via use of words and phrases), and
- application-specific such as those relating to structure and domain-specific vocabulary.

The mentioned textual features are also referred to in the literature as *stylometric/stylistic features* or *stylometric/stylistic markers*. In this survey, I refer to the lexical, character-based, syntactic and semantic features collectively as *linguistic features* and all different types of features in general as *stylistic features*. Selection methods pertaining to linguistic features are presented below; for each of the methods, a brief explanation together with some representative work are included. Application specific features are also briefly introduced and will be discussed in more detail in Section 11.2.

5.2.1.1 Lexical features The early work on authorship attribution in the late 18th and early 19th centuries was focused on attributing the literary work (typically in the form of poetry and drama) of Shakespeare. These efforts are mainly based on *metric* and *rhythm* such as frequencies of end-stopped line, double endings and rhyming lines (c.f., [305]).

The late 19th century witnessed the growth of authorship attribution out of the realm of poetry and drama, and embraced the first appearance of attribution methods based on lexical features. This series of work is initiated by De Morgan [69] who determined authorship via comparison of the average *word-lengths* of an anonymous text and the known texts. Morgan's effort was followed by Mendenhall [205, 206] who improved on the simplistic average word-length feature by computing the statistics for the entire word-length distribution. Though the methods were adopted by other researchers at the time, word-length in general did not gain sufficient popularity because it is sensitive to the differences of the subjects and languages rather than the differences in authorship.

The next lexical features that were studied are based on *sentence-length*: instead of using average word-length and word-length distribution, Eddy [85] and other researchers [309, 314, 315] investigated textual measures based on average sentence-length and sentence-length distribution. Sentence-length also has its pitfalls, the most notable among which is that the variation of sentence-length across texts by a single author is generally large, which in some cases overlapped with the variation of sentence-length across texts by different authors.

Receiving less attention are features based on *contractions* (e.g., *in't* vs *in int*, *o'the* vs. *of the*, and *on's* vs. *on us*) [95] which counts the occurrences of different contraction types

and uses them to distinguish texts written by different authors. Frequencies of *punctuation marks* such as periods, question marks and colons in addition to other attribution features was also studied in [53, 225].

The next lexical features to be presented, which turned out to be more effective than they might seem are those based on 'errors'. For instance, the list of error-types used for quantification of error-based features are taken verbatim from [227] as follows: (1) *Spellings*, (2) *Capitals*, (3) *Punctuation*, (4) *Paragraphing*, (5) *Titles*, (6) *Person*, (7) *Number*, (8) *Case*; (9) *Pronoun and antecedent*; (10) *Verb and subject*; (11) *Modd*, (12) *Tense*; (13) *Voice*; (14) *Possessives*; (15) *Omissions*; (16) *Interlineations*; (17) *Erasures*; (18) *Repetitions*; (19) *Facts or statements*. Though errors are used as a means of identifying authorship in modern forensic document examination, one should keep in mind that in the same way as a person's vocabulary is enriched with time, spelling/grammar errors and mistakes are likely to be corrected over time.

Vocabulary richness [123, 315] is another textual measure used to capture the writing style of an author based on an assumption that each author has a preference for usage of certain words in his/her vocabulary. Here, vocabulary richness is computed as a single measure which is either a ratio of the number of word-tokens to the number of word-types, or a ratio of the length of the text to the size of the text's vocabulary.

Since the univariate analysis carried out in vocabulary richness is deemed not adequate to capture the richness of the vocabulary, Smith [197] proposed a multivariate analysis for vocabulary richness in which *frequencies of individual words* are measured and analysed. For instance, Ellegard [88] compiled a list of words, calculated the ratio for each of the words from the text corpus of an author, and selected those that are, to a certain extent, most representative of that author. The selected values would then be compared with the corresponding values obtained from the anonymous text. *Prima facie*, univariate and multivariate analyses on vocabulary richness seem plausible methods; however (again), the assumption on which they are based has a fundamental flaw: the frequencies of words are more likely to vary according to the subject rather than the author. This recognised drawback reflected the necessity of *content-independent* features. To this end, Mosteller and Wallace [217, 218, 219] introduced the use of *frequencies of function words*[24] as textual measures. Their work is considered one of the most influential work on lexical features, and a seminal work for non-traditional authorship attribution. Not only being content-independent, function words implicitly capture syntactic information, and often occur at higher frequencies in a piece of text. Function words have been used for attribution of the Federalist papers[25] [217, 218, 219], and since then have received significant attention. Nowadays, function words are still among the most popular features used in conjunction with other measures.

5.2.1.2 Character-based features Instead of looking at the whole words, character-based features are focused on the constituents of a word. The first authorship indicators

[24]In contrast to content words, function words are used to express grammatical relationships. Examples of function words include conjunctions (*and*, *thus*, *so*, ...), prepositions (*at*, *by*, *on*, ...), articles (*a*, *an*, *the*, ...) and quantifiers (*all*, *some*, *much*, ...).

[25]The Federalist Papers currently serves as a conventional benchmark for the evaluation of authorship attribution methods.

of this type are those concerned with graphemes. *Grapheme features* based on frequencies of characters of the alphabet were first proposed by Yu [315]. This idea was then further studied in much detail by Merriam [207, 208] who demonstrated that graphemes can be a potentially useful indicator for authorship. For example, Merriam [208] showed that the relative frequency of the letter *O* seemed to distinguish between the two authors Shakespeare and Marlowe: all 36 of Shakespeare's plays has a relative frequency of *O* over the score of 0.78, and 6 out of 7 plays by Marlowe has a relative frequency of *O* under the score of 0.078. Although work on graphemes empirically demonstrated some success, graphemes are not widely accepted as textual measures in the authorship attribution community, partly due to the lack of well-founded and intuitive reasons associated with the technique.

Receiving much more attention are *character-level n-grams* — tokens containing *n* continuous characters. With this definition, *y*, *e*, *s* are 1-gram tokens, and *_y*, *ye*, *es* and *s_* are 2-gram tokens, of *yes*. The most pronounced feature of n-gram analysis is that it can be performed across languages. Keselj and colleagues [154] used this method and demonstrated the success in distinguishing between sets of English, Greek and Chinese authors. An impressive degree of accuracy achieved by n-gram techniques is also presented in [57, 239]. The success of n-grams lies in (a) its provisions of language independence, (b) its implicit capture of the essence of other lexical and character-based methods such as frequencies of graphemes, words and punctuation, (iii) its ability to work with documents of arbitrary length, and (iv) its minimal storage and computational requirements.

N-grams can be used at different levels: word (colocation of words), character, byte (c.f., [103]) and syntactic (c.f., [2]), and have been used in a variety of applications such as text authorship attribution, speech recognition, language modelling, context sensitive spelling correction and optical character recognition [102].

5.2.1.3 Syntactic features It seems intuitive that the writing style of a person is more strongly connected to features at the syntactic level, rather than at the lexical level. For instance, authors are likely to have different preferences regarding the construction of sentences (complex or simple), the voice of verbs (active or passive), the use/construction of phrases (e.g., noun phrases, verb phrases, adjective phrases, and adverb phrases), and the use of parts of speech (e.g., noun, pronoun, verb, adjective, adverb and proposition). The fact that syntactic features are believed to more faithfully represent the writing style of a person, together with the success of function words[26], have motivated efforts to investigate syntactic information as linguistic features for authorship attribution. Stamatatos [279] compiled a list of attributional studies that used syntactic features. For example, Baayen and colleagues [14] as well as Gamon [104] used rewrite rule frequencies as syntactic features; researchers in [279, 277, 122, 191, 294] attempted to extract syntactic information (e.g., noun phrases and verb phrases) and used their frequencies and lengths as syntactic features, researchers in [104, 159, 176, 317] investigated the use of frequencies, and n-gram frequencies, of part-of-speech (POS) tags[27]; Koppel and Schler [159] based

[26]Since function words naturally exist in many syntactic structures, they are also considered syntactic features.

[27]A POS tag is assigned to each word-token in the text and indicates morpho-syntactic information relevant to the word token.

their attribution methods on syntactic errors such as sentence fragments and run-on sentences; and Karlgen and Erilsson [151] made use of adverbial expressions and occurrence of clauses within sentences. The use of syntactic features in conjunction with lexical and character-based features has been demonstrated to enhance the accuracy of attribution results. However, the efficacy of methods using syntactic features as features rely on the availability and accuracy of a natural language processing (NLP) tool. There are in fact some attribution methods using syntactic features that achieved unsatisfactory results due to the low accuracy of the commercial spell checker utilised to extract desired syntactic information (cf., [159]).

5.2.1.4 Semantic features Only a few attempts are directed at using semantic information as textual features. The limited number of efforts devoted to semantic features is due to the difficulty in extracting reliable and accurate semantic information from text. Nevertheless, Gamon [104] developed a tool that produced semantic dependency graphs from which binary semantic features and semantic modification relations are extracted to be used in conjunction with lexical and syntactic information. McCarthy and colleagues [201] presented an idea of using of synonyms, hypernyms, and causal verbs as semantic information for a classification model. Finally, Argamon and colleagues [9] conducted an experiment of authorship attribution on a corpus of English novels based on *functional features*, which associated certain words and phrases with semantic information.

Since there is no single feature selection that is incontrovertibly superior than other features, good results are likely to come from the analysis of a broad set of features, as well as the reported result and recommendations offered by existing work in the literature, to carefully select the features that are most suitable for the task at hand. In this regard, Grieve conducted a comprehensive evaluation of methods based on many of the lexical, character-based and syntactic features [114]. More specifically, Grieve [114] compared the results of thirty-nine methods based on most commonly used linguistic features carried out on the same dataset, and suggested the likely best indicators of authorship. According to this study, attribution based on function words and punctuation marks achieved the best results, followed by methods based on character-level 2-grams and 3-grams. Motivated by the fact that the combination of words and punctuation marks achieved an even better predictive performance than the sole use of n-grams, Grieve devised a weighted combination algorithm that combines sixteen methods based on the linguistic features with the most successful results, where the significance of each individual method is weighted according to the performance it achieved in the experiment. The algorithm was reported to successfully distinguish between twenty possible authors, and distinguish between five possible authors with over 90% accuracy. Grieve [114] concluded that the best approach to quantitative authorship attribution appeared to be one that is based on the results of as many proven attribution algorithms as possible. It should be noted, however, that the feature selection is dependent on author and text genre. The best features for one author may be different for another.

5.2.1.5 Other types of features Content-specific features are heavily used in text categorisation (see Section 6), but are somewhat discouraged from use in research on authorship attribution due to the potential classification bias resulting from the influence

of content-specific features on classifiers to categorise texts according to the topics, rather than to the authors. However, in a controlled situation (e.g., pieces of text belonging to the same domain or genre), content-specific features can be useful in discrimination among authors.

Many kinds of modern text (such as email, webpage, representation and computer code) have customised structure and layout. Features based on the structure and layout (*structural features*) of a piece of text may provide a strong indication of its author (e.g., software programmers tend to have different preferences in the manner they structure their code with respect to indentation, commenting and naming). Also, words in a piece of text can be morphologically related (such as *run*, *ran* and *running*) or unintentionally/intentionally misspelled (such as *phishing* and *fishing*), which necessitates a type of feature that captures this type of information — hence, one has *orthographic features*.

Finally, *metadata* (such as names of the author and the developing tool displayed in the properties of a digital document, or information about the sender and travelling path of an email provided in its header) also serves as an important cue to authorship. Though metadata is often tampered with during commission of cyberspace offences, it still plays a role, and should not be entirely overlooked, in attribution of a digital item.

The types of features presented in this section will be discussed in more detail in Sections 11.2 and 11.3. The most developed automated authorship attribution tool publicly available is JGAAP[28] developed by a research group led by Patrick Juola [148]. Core functionalities provided by JGAAP includes textual analysis, text categorisation and authorship attribution.

5.2.2 Attributional analysis

Attributional analysis involves examining a space of feature vectors, each corresponding to a piece of text, for the purpose of determining authorship for an anonymous text. Taking as input the space of feature vectors, a basic algorithm would first combine multiple vectors belonging to an author into a profile, which can be simply done by averaging the values of each feature item across the vectors. The algorithm then compares the feature vector corresponding to the anonymous text with each of the author profiles, making use of simple statistical methods such as chi-squares (see Section 6.2.1), to determine which pair is the closest match. This type of algorithm and the alike are considered computer-assisted methods. The relatively recent adoption of machine learning algorithms has opened a new horizon for authorship attribution: that of embracing attribution problems of a larger scale and with a higher degree of difficulty; and of addressing the problems in a more efficient computer-based automated manner. Since techniques in machine learning are cornerstones of many authorship attribution methods discussed in this survey — not only those based on textual evidence — they merit a separate discussion (see Section 7).

[28] JGAAP is available for download at `http://evllabs.com/jgaap/w/index.php/Main_Page`.

6 Text categorisation

The task of authorship attribution essentially involves categorising pieces of text according to human writing style (or *style-based text categorisation*). In a very broad sense, text categorisation studies the bidirectional mapping between a domain of documents and a set of predefined categories (e.g., discussion topics, genres or genders). A mapping from a document to categories, i.e., identifying all the categories for a given document, is referred to as *document-pivoted categorisation*. Conversely, a mapping from a category to documents, i.e. finding all the documents belonging to a pre-defined category, is known as *category-pivoted categorisation*. Generally, text categorisation is *content-based* (i.e., categorisation is performed based on the information extracted from the contents of documents), and thus it utilises a wide range of techniques from *information retrieval*[29]. In modern settings, text categorisation is automated. Therefore, like authorship attribution, computational methods for text categorisation are built on techniques from *statistics* and *machine learning*. Text categorisation has been applied in a range of applications including automatic indexing for Boolean information retrieval systems, document organisation, text filtering, word sense disambiguation, and hierarchical categorisation of Web pages [263].

Since text categorisation possesses efficient methods to store, retrieve and handle a large number of documents each of which potentially consists of a large number of features, methods studied in text categorisation may be useful in assisting many subtasks of attribution of spear phishing where documents are replaced by emails, malicious software code, or attacker profiles. The literature of text categorisation is extensive. In this section, I aim to summarise the fundamentals of text categorisation to assist the reader in appreciating various attribution techniques (including authorship attribution) discussed in the survey that explicitly/implicitly make use of methods investigated in this paradigm.

In general, a text categorisation procedure consists of three global steps: *feature extraction*, *dimensionality reduction*, and *text categorisation*, respective descriptions of which are given below.

6.1 Feature extraction

Feature extraction comprises the steps that are needed to transform raw text into a representation suitable for the categorisation task. This phase corresponds to feature selection in authorship attribution. The term *extraction* is used to emphasise the fact that text categorisation is content-based and thus a list of features are dynamically extracted from the text. The steps of feature extraction are presented below.

- **Preprocessing**: preprocessing involves activities to remove 'noise' from a document to be categorised, including (i) removal of HTML (and other) tags, (ii) removal of 'content-free' words, or *stopwords*, (e.g., function words), and (iii) performance of *word stemming*, or restoring the root of a word, (e.g., *went*, *gone* and *going* are

[29] Information retrieval investigates methods to retrieve desired information from a large volume of text documents.

Table 1: *A list of weighting methods utilised in text categorisation.*

Boolean weighting: let the weight be 1 if the word occurs in the document and 0 otherwise.	$a_{ik} = \begin{cases} 1 \text{ if } f_{ik} > 0 \\ 0 \text{ otherwise} \end{cases}$
Word frequency weighting: let the weight be the frequency of the word in the document	$a_{ik} = f_{ik}$
tf×idf-weighting: incorporates into the weight the frequency of the word throughout all documents in the collection, in addition to the frequency of the word in the document.	$a_{ik} = f_{ik} \times log(\frac{N}{n_i})$
tfc-weighting: takes into account the different length of documents, in addition to the frequency of the word throughout all documents in the collection, and the frequency of the word in the document.	$a_{ik} = \frac{f_{ik} \times log\left(\frac{N}{n_i}\right)}{\sqrt{\sum_{j=1}^{M}\left[f_{jk} \times log\left(\frac{N}{n_j}\right)\right]^2}}$
ltc-weighting: modifies tfc-weighting and uses the logarithm of the word frequency.	$a_{ik} = \frac{log(f_{ik}+1.0) \times log\left(\frac{N}{n_i}\right)}{\sqrt{\sum_{j=1}^{M}\left[log(f_{jk}+1.0) \times log\left(\frac{N}{n_j}\right)\right]^2}}$
Entropy weighting is the most sophisticated and effective scheme which measures the uncertainty of a word. Entropy weighting is -1 if the word is equally distributed over all documents, and 0 if the word occurs in only one document.	$a_{ik} = log(f_{ik} + 1.0) \times \left(1 + \frac{1}{log(N)} \sum_{j=1}^{N}\left[\frac{f_{ij}}{n_i} log\left(\frac{f_{ij}}{n_i}\right)\right]\right)$ where $\frac{1}{log(N)} \sum_{j=1}^{N}\left[\frac{f_{ij}}{n_i} log\left(\frac{f_{ij}}{n_i}\right)\right]$

converted to *go*). Each document is then transformed into a vector of frequencies of words extracted from the document.

- **Indexing**: indexing involves determining the weights of a certain word in a certain document (in the form of a feature vector). Various weighting methods have been proposed (see Table 1) which are based on the two following arguments:

  – a high number of occurrences of a word in a document suggested a strong relevance of the word to the topic of the document, and

  – a high number of occurrences of a word in *all documents* discounts the usefulness of the word in discriminating between the different documents.

6.2 Dimensionality reduction

A conventional way to present the transformed document is the vector space model [258]. In this model, all the documents are represented in a word-by-document matrix (informally referred to as a *bag of words*), in which each entry represents the occurrences of the word in a document. Since the vocabulary used in different documents is often diverging, a word-by-document matrix tends to be very large and sparse. This poses a difficulty to classification algorithms. One solution to this problem is to reduce the dimension of the original feature vectors that represent the documents, through either *feature reduction* or *feature transformation*.

6.2.1 Feature reduction

The most straight forward approach to reduce the number of features (i.e., words) representing a document is to discard features that are less informative to the categorisation task. Feature selection can be simply done by selecting features with frequency exceeding a pre-defined threshold (*frequency thresholding*), or measure how relevant the features are using a variety of techniques including *Shannon entropy* and *chi-square statistics*.

- **Shannon entropy** is originally a method in *information theory* that measures the uncertainty associated with the value of a certain variable. Shannon entropy can be applied to measure the informativeness of a word based on the distribution of the word in all the documents using the following equation:

$$H(P_w) = -\sum_{i=1}^{N} p_{i,w} log_2 p_{i,w}$$

  where $H(P_w)$ is the entropy value for the word w, and $p_{i,w}$ is the unnormalised distribution of the word w over the N documents. A word with a small entropy value (i.e., a word that is not evenly distributed in all the documents) tends to be more discriminative while the one having a large entropy value (i.e., a word that is more evenly distributed in all the documents) is less useful in discriminating among documents. Therefore, to reduce the number of features representing a document, only features with low entropy values should be selected.

- **Chi-square statistics** measure the lack of independence between a specific word and a specific category, using the following equation:

$$\chi^2(w,c) = \frac{N \times (D_c^w D_{\neg c}^{\neg w} - D_c^{\neg w} D_{\neg c}^w)^2}{(D_c^w + D_c^{\neg w}) \times (D_{\neg c}^w + D_{\neg c}^{\neg w}) \times (D_c^w + D_{\neg c}^w) \times (D_c^{\neg w} + D_{\neg c}^{\neg w})}$$

  where D_c^w, $D_c^{\neg w}$ denotes the number of documents from class c that contain, or do not contain, the word w, respectively; $D_{\neg c}^w$, $D_{\neg c}^w$ denotes the number of documents containing the word w that belong to, or do not belong to, the class c, respectively; and N is the total number of documents. Words that are independent from the classes are considered less informative, and consequently are discarded.

6.2.2 Feature transformation

Instead of eliminating features from a feature vector that represents a document, the feature vector can be mapped to a new vector with smaller size by merging closely related features. A cornerstone technique underlying various feature transformation approaches (such as *principle component analysis* and *latent semantic analysis*) is *singular value decomposition.*

6.2.2.1 Singular value decomposition (SVD) Given a $M \times N$ word-by-document matrix A, where M is the number of words, and N is the number of documents, the singular value decomposition of A is mathematically given by:

$$A_{MN} = U_{MM}\Sigma_{MN}V_{NN}^{T}$$

where U_{MM} and V_{NN} are orthogonal matrices, Σ_{MN} is a diagonal matrix, and V_{NN}^{T} is the transpose of V_{NN}.

The underlying idea of singular value decomposition is that, by breaking down the word-by-document matrix A into orthogonal and diagonal matrices, it allows for the transformation of correlated variables into a set of uncorrelated ones. The transformation possesses two important properties: (i) it reveals various relationships between the original features, and (ii) it allows for the identification or ordering of the dimensions along which the features exhibit the most variation. It is these properties that make several approaches to feature transformation possible.

6.2.2.2 Principal component analysis (PCA) PCA is a popular feature transformation method that uses SVD to discard features with small variance over the dataset. Features with more variation in the dataset allow better separation of documents into categories, and thus are retained.

6.2.2.3 Latent semantic analysis (LSA) LSA is a feature transformation method that aims to address the two fundamental problems that human languages pose to machine categorisation: synonymy and polysemy. Latent semantic analysis applies SVD to the original word-by-document matrix to produce/obtain a function that maps the original vectors into new vectors of a lower-dimensional space through combination of the original dimensions (e.g., words with closely related meanings, inferred by looking at their patterns of occurrence). In practical applications, the mapping function is derived from the training set and applying it to test documents.

7 Machine learning

Machine learning is not only considered a mainstay of modern authorship attribution, but also plays an important role in any attribution-relevant method based on data analysis such as clustering and classification. Hence a sufficient understanding of the fundamentals of machine learning is a *prerequisite* to comprehend a wide range of (computational) attribution methods. However, while treatment of individual machine learning methods in isolation, as found in published work on attribution and profiling, offers some specific details, it fails to present the methods in a larger picture of the research field, and in relation to other approaches and techniques. Also, existing descriptions of machine learning algorithms from different perspectives using different terms and names, as well as the overlapping of machine learning and data mining, may also increase the degree of confusion. This section aims to provide a succinct, but systematic and coherent, high-level presentation of key machine learning schemes important for the tasks of attribution. It also aims to clarify the mentioned confusions.

7.1 Machine learning and data mining

Machine learning and data mining can be broadly defined as follows:

- *Machine learning* refers to the study and computer modelling of learning processes [209],
- *Data mining* refers to the process of discovering useful patterns, automatically or semi-automatically, in large quantities of data [311].

Though possessing different definitions, the paradigms of machine learning and data mining are strongly related — it is not uncommon for a method described in the literature of machine learning to be discussed at the same time as a data mining approach. A familiar and intuitive example taken from human life may help to enlighten this perplexing relationship. On entering life, a newborn experiences an entirely unfamiliar world filled with unknown things. To assimilate within the environment, the newborn needs to make sense of things around it — in other words, to extract useful information, or patterns, from a vast amount of data available. The child is said to face a *data mining* problem which it addresses via *learning*[30]. More specifically, the child uses its innate ability to learn — to relate things according to their similarities and differences, and to perceive things through repeated interaction with and within the environment — in order to develop understandings (e.g., recognising the patterns of its mother's face) and judgements (e.g., distinguishing its mother from strangers) about the world. Data mining and machine learning are motivated by the study of these phenomena computationally. Just as the processes of extracting useful information from the data available and of learning carried out by human beings are intertwined, data mining and machine learning are much overlapping disciplines. Data mining often makes use of machine learning algorithms in addition to

[30]Definition and analysis of learning is a philosophical topic. In this example, I aim to convey an intuitive perception of data mining and learning without an intent to explain learning from a philosophical point of view.

statistical methods. However while machine learning puts more focus on the theoretical aspect of automated learning, data mining is mostly concerned with the practical aspects of the learning processes, especially in circumstances where (i) the dataset is real, large, and noisy, and (ii) the efficiency and scalability of the learning algorithms is important. Similarly, machine learning also forms a continuum with statistics with respect to data analysis techniques. As such, many of the machine learning methods discussed below can also be referred to as data mining or statistical methods. A large part of the following section is based on a most recent publication on the topic of data mining by Witten and colleagues [311]. For a more detailed description of the methods and further references, please see [311].

7.2 Fundamentals of machine learning

The objective of a machine learning scheme is to learn about a *concept*, if possible, from a set of *instances* (concrete examples of the concept), and produce a *concept description*. The concept description is then used to obtain knowledge about unknown matters with respect to that concept. For example, a machine learning scheme may learn about *classification of Iris flowers* by examining features (petal/sepal lengths and widths) of *individual Iris flowers* and generate a *classification description* which can be used to determine the classes for unseen Iris flowers.

There are four main types of learning. In *classification learning*, a learning scheme learns to classify unseen instances from a set of classified instances, as illustrated in the example above. In *associating learning*, a learning scheme aims to disclose interesting association among features. In *clustering*, instances that have features closely related to each other are combined together. In *numeric prediction*, which is a variant of classification learning, the learning outcome is a numeric quantity, instead of a concrete class. Numeric prediction, classification, and association learning are sometimes referred to as supervised learning since their learning process is trained on a set of given examples, and guided by the actual outcomes of the examples. Clustering is, in contrast, considered unsupervised learning since no training examples are required. Clustering is usually followed by an application of another learning scheme (e.g., classification or numeric prediction) to provide a description of how a new instance should be placed into the classes defined by the clusters.

In operational settings, a machine learning scheme consists of three components: an input, a learning algorithm, and an output. An understanding of the input and output of a machine learning scheme is important since characteristics of the input and output determine the types of learning algorithm operating between them.

7.3 Machine learning input

A machine learning scheme takes as input a set of *independent* instances of the concept to be learnt. Each instance is represented as a fixed, pre-defined set of features, or *attributes* (e.g., an input Iris flower is given as a set of four attributes: sepal width, sepal length, petal width and petal length). Values assigned to an attribute can be of different types, two most popular among them are *nominal* and *numeric*. A nominal attributes, also known

as a *categorical* attribute, has its value from a pre-specified, finite set of possibilities. A numeric attribute, also referred to as a *continuous*[31] attribute, takes on real-valued or integer-valued measures.

7.4 Machine learning output

The learning outcome of a machine learning scheme is a description of the concept whose instances are given as input. These concept descriptions are also referred to in the literature as learning models, and in case of classification learning, a concept description is also referred to as a classifier. Knowledge of the output can be represented in one of the following forms.

7.4.1 Tables

Tables are the most basic way to display an output from a learning process (in a similar way it is used to represent an input dataset). Each row of the output table contains a distinct combination of attribute values (a *condition*) together with a learning outcome — a numerical quantity in the case of numeric prediction, and a concrete class in the case of classification — associated with the condition. To predict a numeric quantity or a concrete class for an unseen instance, one only needs to look up the appropriate condition in the table and obtain the outcome associated with the condition.

7.4.2 Linear models

Linear models are another simple way to represent learning output. Here, the output knowledge takes the form of a linear equation:

$$PRD = w_0 + w_1 a_1 + w_2 a_2 + ... + w_k a_k$$

where $PRD, a_1, a_2, ..., a_k$ are variables representing the prediction and attribute values for a given instance, and the weight factors $w_1, w_2, ..., w_k$ are constant values determined by the learning process. To test a new instance, the instance's attribute values are simply substituted into $a_1, a_2, ..., a_n$, and the evaluated PRD is the prediction value for the new instance. Since research in the statistical community uses the term *regression*[32] for the process of predicting a numeric quantity, *regression model*, or *linear regression*, are also used interchangeably with linear model.

As shown above, linear models are naturally fitted to learning problems whose attributes of both the inputs and output are of numerical values. However, it can also be

[31] Please note that the term *continuous* in this context should not be interpreted as *continuous* in the mathematical sense.

[32] Again, please note that the term *regression* in statistics should not be understood/interpreted in its popular meaning as returning to a former state [311].

used for binary classification. In this case, instances are grouped into two classes separated by the following hyperplane, or *decision boundary*, defined by the following equation:

$$w_0 + w_1 a_1 + w_2 a_2 + ... + w_n a_k \ = 0$$

7.4.3 Trees

A more sophisticated representation of output knowledge is a *tree* in which each branch node represents a choice between a number of alternatives, and each leaf node represents an outcome which can be of nominal or numerical value. In the former case where outcomes contain nominal values, each leaf node is said to represent a category (or class), and the tree is referred to as a *decision tree*. In the latter case in which the outcomes contain numeric values, each leaf node is said to represent a predicted numeric quantity, and the tree is named a *regression tree*. Testing an unseen instance can be done by successively evaluating its attribute values at the branch nodes, starting from the root of the tree, following down the successful branches until a leaf node is reached. The class or numeric value associated with the leaf node is assigned to the unseen instance.

A decision tree is a more powerful representation than a (classification) linear model in that it can classify instances into more than two categories[33]. Similarly, a regression tree usually produces more accurate results than a (linear) regression model. However a regression tree generated for a practical problem can be difficult to interpret due to its cumbersome size. Therefore a more compact and sophisticated representation is achieved by combining a regression tree and a linear regression into what is called a *model tree*. A model tree closely resembles its predecessor regression tree, however single predicted values contained in the leaf nodes of a regression tree are replaced by regression equations in a model tree. Figure 2 illustrates examples of linear regression, regression tree and model tree.

7.4.4 Rules

A common alternative of decision trees for representing output knowledge is rules.

An example of a rule is displayed below:

`if petal_length =` x `and petal_width =` y `then class =` z

where elements in the *if* clause are collectively referred to as *antecedent*, or precondition, and the element in the *then* clause as *consequent*, or conclusion.

The antecedent contains a series of tests similar to those contained in the branch nodes of a decision tree. In the case of classification, the consequents of rules, which correspond to leaf nodes of the tree, specify classes or a probability distribution over the classes. As such, *prima facie*, rules can be extracted from a decision tree (one rule is generated for

[33]Though a linear model is naturally suited to binary classification, it can be extended to perform multiclass prediction.

each leaf node) and vice versa. However, transformation between rules and a decision tree faces two problems. On the one hand, rules extracted from a decision tree may be more complex than necessary. On the other hand, it can be problematic to express disjunction rules (rules with different antecedents and the same consequent) in a decision tree. While rules gain popularity due to simple and compact representation of knowledge, intuitive representation and ease of interpretation are the strengths of decision trees.

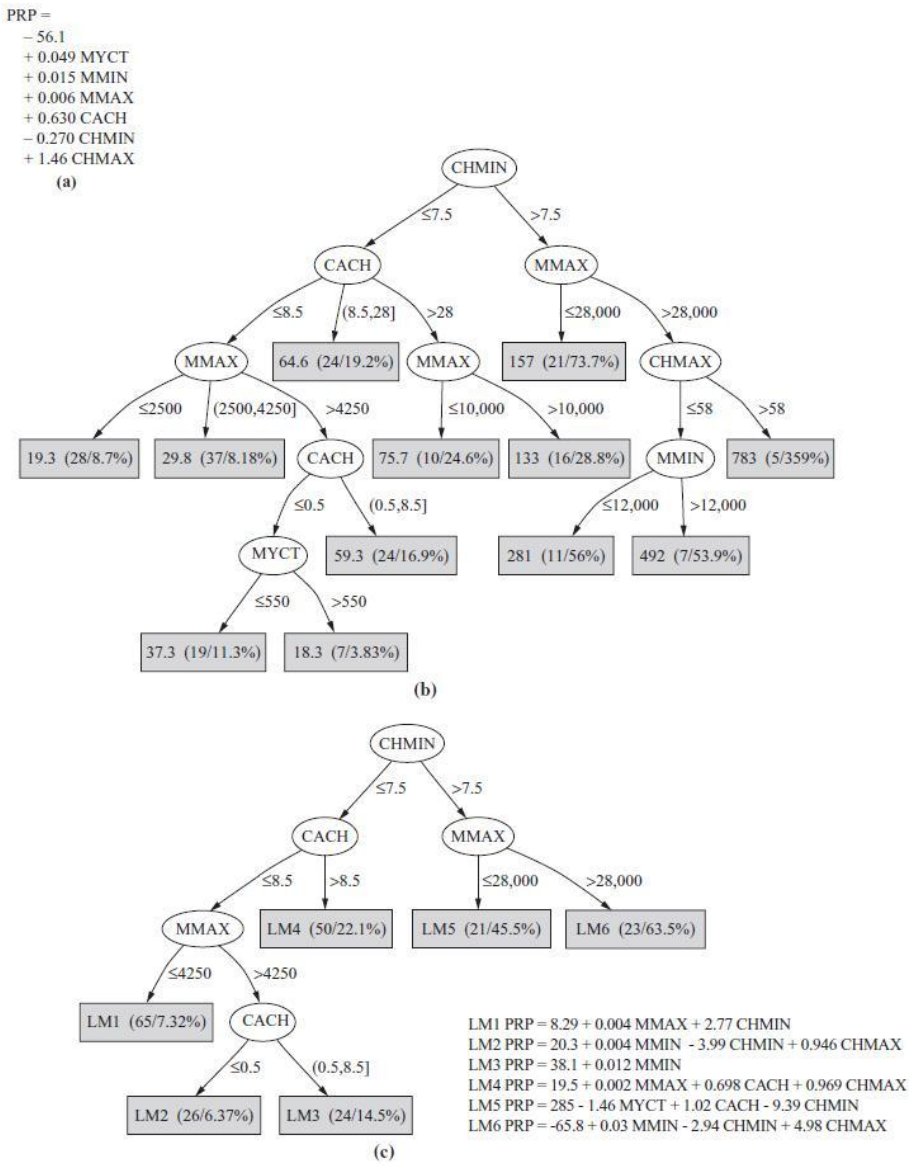


Figure 2: *An illustration of (a) linear regression, (b) regression tree, and (c) model tree [311].*

Association rules are similar to classification rules except for their ability to predict values for not only classes, but any attribute, or potentially any combination of attributes.

Below is an example of a simple association rule:

$$\texttt{if petal\_length =}\ x\ \texttt{then petal\_width =}\ y$$

Since there is an exponential number of combinations of attributes, a major concern in implementation of association learning is to restrict the number of association rules in the learning system. A general guideline is to choose only rules that have a high *coverage*, or *support*, as well as high *accuracy*, or *confidence*. The coverage of an association rule is the number of instances for which it predicts correctly (e.g., the number of Iris flowers that have `petal_length` = x and `petal_width` = y), and the accuracy is the proportion of the number of instances with a matched condition that the rule predicts correctly (e.g., the proportion of Iris flowers with `petal_length` = x that have `petal_width` = y).

7.4.5 Instances

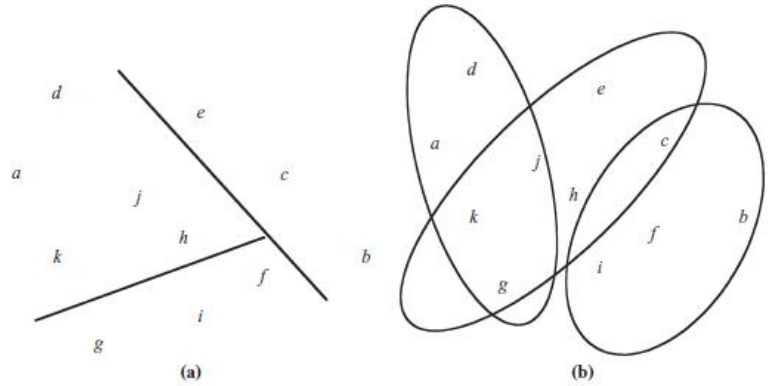
The representations of output knowledge presented above are for *eager learning* (e.g., numeric prediction, classification or association learning). Eager learning tends to produce a generalisation as soon as instances are given as input. In contrast, a *lazy learning* (or *instance-based learning*) method simply memorises a set of training instances and only initiates a learning process when an unseen instance needs to be classified. Hence, the output knowledge is the set of instances itself. To classify a new instance, a learning algorithm compares the new instance to each of the existing instances using a distance metric, and the class of the closest existing instance is assigned to the new instance.

7.4.6 Clusters

In classification learning, a classifier is learnt. Likewise, in clustering, clusters are produced. A clustering output takes the form of a diagram that depicts how instances are grouped together. There are different types of diagrams representing clusters as shown in Figure 3 [311]. Which diagram among these to be used depends on the relationship among the generated clusters, whether they are *exclusive* (i.e., *disjoint*), *overlapping*, *probabilistic*, or *hierarchical*. For example, the diagram in Figure 3(a) is suitable for instances that are grouped into exclusive subsets; the diagram in Figure 3(b) is convenient for representing overlapping clusters; the diagram in Figure 3(c) is dedicated for cases where each instance is assigned a membership probability specifying the extent to which it belongs to each of the clusters; and the diagram in Figure 3(d), also called a *dendogram*, is used for clusters related in a hierarchical manner. A clustering output can be *complete* or *partial*. A complete clustering assigns every instance to a cluster, while a partial clustering on the other hand allows some instances to remain alone.

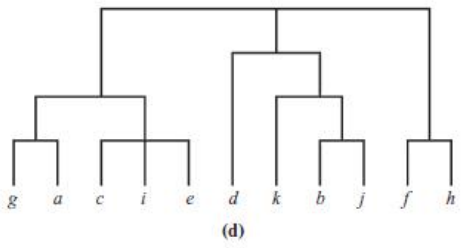
7.5 Machine learning algorithms

Having presented the types of input and output for a machine learning scheme, this section presents major learning algorithms that operate between different pairs of input and output. For each type of machine learning scheme, I first present learning procedures/methods



	1	2	3
a	0.4	0.1	0.5
b	0.1	0.8	0.1
c	0.3	0.3	0.4
d	0.1	0.1	0.8
e	0.4	0.2	0.4
f	0.1	0.4	0.5
g	0.7	0.2	0.1
h	0.5	0.4	0.1
...			

(c)



(d)

Figure 3: *An illustration of the different types of clusters [311].*

in laboratory settings to convey the fundamental principles underlying a range of thematic algorithms. Then I introduce more robust algorithms applicable to practical settings where the algorithms need to deal with missing values and noisy data. For ease of exposition, I denote n the number of input/training instances, $a_1, a_2, ..., a_k$ the k attributes of the instance under discussion, $a_1^{(i)}, a_2^{(i)}, ..., a_k^{(i)}$ the attributes of an instance i, and $w_1, w_2, ..., w_k$ the coefficients (or weights) of a linear expression.

Please note that even though many algorithms in this section are presented for binary classification, they can be applied to more than two classes.

7.5.1 Statistical machine learning

A simple but efficient and well-known approach in this group is *Naïve Bayes*. The name of this method is so coined because it is based on Bayes' theorem and assumes that all attributes of the training instances make an independent and equally important contribution to the decision — an assumption which seldom holds in practice (hence, *naive*).

Despite its unrealistic assumption, Naïve Bayes has been shown to be an efficient learning method, which achieves an impressive result in many real-word complex applications. As a method for classification learning, Naïve Bayes particularly gains its popularity in document classification due to its speed and high accuracy. Naïve Bayes computes the probability that a new instance falls into each of the classes, based on the following Bayes' rule:

$$Pr[C|E] = \frac{Pr[E|C]Pr[C]}{Pr[E]}$$

where C is the class under consideration, E is the particular combination of attribute values for the new instance, Pr[A] denotes the probability of an event A, and Pr[A|B] denotes the probability of A conditional on another event B.

7.5.2 Learning with linear models

Recall that when attribute and output values of a learning problem are all numeric, a natural way to represent the learning outcome is using a linear model which is defined as

$$c = w_0 + w_1a_1 + w_2a_2 + ... + w_na_k$$

A variety of algorithms in this category are presented below.

7.5.2.1 Linear regression for numeric prediction Linear regression computes the weights $w_1, w_2, ..., w_n$ using the *least mean-squared difference* method. Here, for a training instance i of the output value c^i, the predicted output value c^i_p for i is computed as follows:

$$c^i_p = w_0 + w_1a^{(i)}_1 + w_2a^{(i)}_2 + ... + w_ka^{(i)}_k = \sum_{j=0}^{k} w_ja^{(i)}_j$$

Since c^i_p is a predicted value for the training instance i, there is a difference between c^i_p and the actual output value c^i. Linear regression determines the appropriate values for $w_1, w_2, ..., w_k$ by attempting to minimise the sum of squares of the differences over all the training instance. Let d denote the sum of squares defined as follows:

$$d = \sum_{i=1}^{n}(c^i - c^i_p)^2$$

where n denotes the number of training instances.

Linear regression is a simple, efficient, and widely-used method in statistical applications. It also serves as a building block for more complex learning methods.

For a classification problem, basic methods for classification in domains with numeric attributes include: *logistic regression* which produces a function to compute a membership value of a new instance for each class, and *perceptron* and *Winnow* which find a separating hyperplane for a linearly separable problem.

7.5.2.2 Logistic regression for linear classification Logistic regression produces a model that measures the prediction probability that an instance belongs to a specified class. With logistic regression, the prediction probability Pr is given by:

$$Pr = \frac{1}{1 + exp(-w_0 - w_1 a_1 - ... - w_k a_k)},$$

and the decision boundary for a binary classification problem using logistic regression is defined as:

$$Pr = \frac{1}{1 + exp(-w_0 - w_1 a_1 - ... - w_k a_k)} = 0.5$$

Logistic regression is similar to linear regression except that (i) it computes a prediction probability Pr of an instance being a member of a class, rather than the prediction numeric value c_p, and (ii) it determines the weights $w_0, w_1, ..., w_k$ by maximising the membership probability of training instances to their known classes (using the maximum likelihood method), rather than minimising the sum of squares of the differences to the training data.

Discriminant analysis Closely related to logistic regression is discriminant analysis. Discriminant analysis has the same goal as logistic regression, but is based on an assumption that requires the attributes to have normal distributions. One advantage of discriminant analysis is that, in addition to finding a discriminant function that predicts to which class a test instance belongs, it also allows for the identification of features (i.e., attributes) that best reveal the differences among the classes. Discriminant analysis is a good choice when the features are known to be normally distributed. Otherwise, logistic regression in general performs well and requires less assumptions than discriminant analysis, and thus is often preferred to discriminant analysis.

7.5.2.3 Linear classification by finding a separating hyperplane When instances are linearly separable (they can be perfectly separated into two classes by a hyperplane), a learning algorithm can produce a model that directly separates instances of different classes using a hyperplane without a need to compute probability estimation as in logistic regression. A hyperplane in a multi-dimensional space has the following form:

$$w_0 a_0 + w_1 a_1 + w_2 a_2 + ... + w_k a_k = 0$$

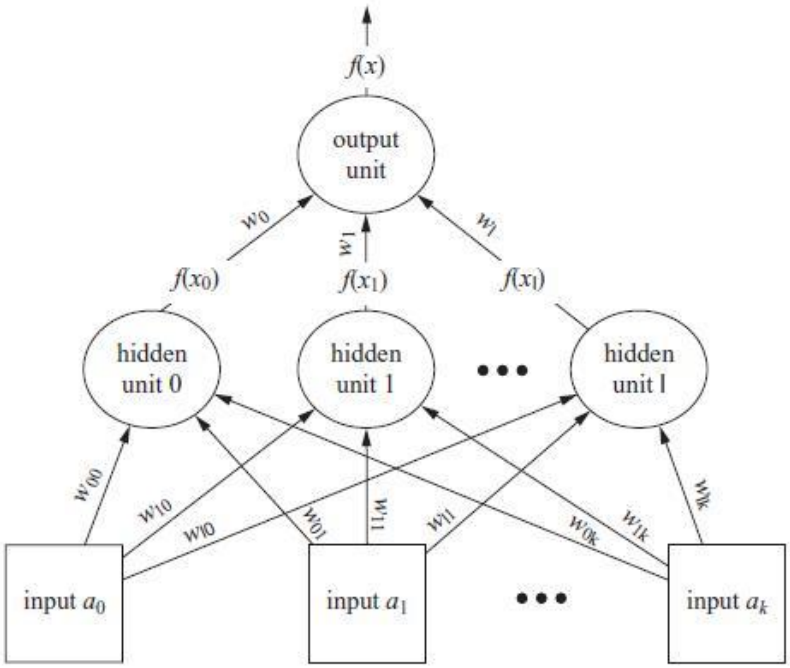


Figure 4: *A graphical illustration of a perceptron [311].*

where a_0 is a *bias* value which always has a value of 1. To classify a new instance, its attribute values are substituted into the LHS of the above equation. If the sum is greater than 0, the new instance is predicted to belong to the first class; otherwise it belongs to the second class. Learning algorithms that determine the weights of the hyperplane are presented below.

- **Perceptron learning rule** Graphically presented in Figure 4, the perceptron learning rule, or *perceptron*, first sets all the weights to zero (or a random value) and repeatedly updates the values in the weight vector if a misclassification is encountered until all the classifications on training data are correct. Weight update can be performed as follows: if the misclassified instance actually belongs to the first class, add its attribute values to the weight vector as following:

$$(w_0 + a_0) + (w_1 + a_1)a_1 + (w_2 + a_2)a_2 + ... + (w_k + a_k)a_k;$$

  otherwise, they are subtracted from it. Weight update essentially moves the hyperplane in the right direction so that it correctly separates the groups of instances. If the data is linearly separable, the algorithm is guaranteed to converge after a finite number of iterations. If the data is not linearly separable, the perception learning algorithm can still be applied in practice, however a upper bound needs to be imposed on the number of iterations in order to guarantee termination.

- **Winnow** An algorithm closely related to the perceptron learning rule is known as *Winnow* which deals with learning problems with binary attributes (an attribute

has 0 or 1 as its value). Having a classification procedure similar to that of the perceptron, Winnow is however distinguished from the perceptron in the two following features:

- Unlike the perceptron which adopts an additive mechanism to adjust values in the weight vector when a misclassification occurs, Winnow employs a multiplicative mechanism: if the misclassified instance actually belongs to the first class, for each of its attributes whose value is 1, multiply the weight associated with the attribute by a user-specified parameter, otherwise divide it by the parameter.
- A threshold θ (a user-specified parameter) is used in prediction making: an instance belongs to class 1 if and only if:

$$w_0a_0 + w_1a_1 + w_2a_2 + ... + w_ka_k > \theta$$

The algorithms presented above naturally produce linear decision boundaries, making themselves inadequate for many practical applications where the data is non-linear. Extensions to the basic algorithms for linear models are needed if one wishes to generate non-linear decision boundaries. One approach in this direction is to transform input data to a higher dimensional space using non-linear mapping. The idea behind non-linear transformation is that a linearly non-separable data set in the original space is likely to become separable in a much higher dimensional space. Robust algorithms popularly used in practical applications employ this technique, including *support vector machines/regression*, *kernel/multilayer perceptron*, *radial basis function networks* and *stochastic gradient descent*. For example, multilayer perceptron, which is the most prominent type of neural network, extends the perceptron described above by introducing nodes in the hidden layers between the input and output layers to address non-linearity. However, interest in multilayer perceptron as well as other types of neural network declined with the arrival of support vector machines (see below). This is due to the fact that support vector machines require fewer parameters to be tuned, and thus are more efficient algorithms in many applications.

7.5.2.4 Support vector machines Support vector machines (SVMs) use a linear model to implement nonlinear class boundaries. SVM achieves this by performing a nonlinear mapping of the feature space containing the input vectors to a high-dimensional feature space where the data is linearly separable. In this way, a linear model constructed in the new space can represent a nonlinear boundary in the original space. However while making use of the nonlinear mapping technique technically resolves the non-linear class boundaries problem, this approach faces two problems: (i) the linear model constructed in the high-dimensional space tends to overfit the training data due to a large number of attributes generated in the high dimensional space and the concomitant high degree of flexibility associated with the decision boundary (performance problem), and (ii) the number of coefficients associated with the linear model constructed in the high-dimensional space is potentially very large, which renders the training process to determine the values for the coefficients infeasible (computational complexity problem).

SVM successfully addresses both of the mentioned problems. SVM addresses the overfitting problem by producing a special type of linear model, known as *maximum-margin hyperplane*, which gives the greatest separation between the two classes. The instances that are closest to the maximum-margin hyperplane are called *support vectors*, a collection of which in turn uniquely define the maximum-margin hyperplane. Since the maximal-margin hyperplane of a SVM is defined by a limited number of support vectors, it has little flexibility and is relatively stable; consequently significantly decreasing the chance of overfitting. With respect to the problem of computational complexity, even though the equation defining the maximum-margin hyperplane (which is frequently evaluated throughout training and classifying new instances) contains a dot product whose computation is very expensive in a high-dimensional space, it is fortunately possible for the dot product to be pre-computed in the original feature space before the nonlinear mapping is performed using a so-called *kernel function*. Commonly used kernel functions are *polynomial kernel*, *radial basis function (RBF) kernel* and *sigmoid kernel*. Which kernel function to use depends on the specific application at hand. In particular, the combination of SVM and the RBF kernel implements a type of neural network known as an *RBF network*, and the combination of SVM and the sigmoid kernel implements another type of neural network which is a *multilayer perceptron*.

7.5.3 Decision tree construction

Construction of a decision tree, by the *divide-and-conquer algorithm*, can be carried out recursively in an intuitive manner: select an attribute to be the root node and create one branch for each value of the attribute. If all instances at a branch have the same class, a leaf node associated with the class is assigned to the branch. These steps are recursively performed for each branch until all branches end with leaf nodes. The problem is to determine which attribute to split on at each step so that the processing on the branches of the tree terminates as soon as possible, resulting in the smallest possible tree. One method that assists in making this decision is based on the notion of *information value*, or *entropy*, which is defined as:

$$\text{entropy}(p_1, p_2, ..., p_k) = -p_1\log p_1 - p_2\log p_2 \ldots - p_k\log p_k$$

where the logarithms are expressed in base 2 and p_i denotes the ratio at the node of the number of instances of class i to the total number of instances.

The entropy at a node is measured in *bits*, and represents the amount of information the node would need to obtain in order to specify the class for an instance, given that the instance reaches that node. When all the training instances at a node belong to a class (i.e., no more information is needed to determine the class for an instance reaching the node), the entropy of the node has its minimum value (0 bits). Conversely, the entropy of a node reaches its maximum value (1 bits) when all the training instances reaching the node are equally divided into different classes (e.g., a new instance reaching this node has an equal chance to belong to any of the classes). Since different attributes selected to split on at a node will result in different entropy values for that node, the attribute to be chosen should be the one that offers a maximum *information gain* (which is the maximum difference between the entropy of the parent node and that of the current node). The

construction of the tree stops when at each branch the data cannot be split any further, or the information gain is zero.

The divide-and-conquer algorithm is efficient, but tends to overfit to the training data[34]. The algorithm also does not deal with numeric attributes and missing values. Various algorithms have been proposed to extend and improve on the divide-and-conquer algorithm, among which are the current state-of-the-art C4.5 and C5.0 algorithms. C4.5 and C5.0 algorithms are able to handle numeric attributes and missing values, as well as implement a post-pruning method to produce a more compact tree that generalise better to sets of unseen instances. They also facilitate conversion between a decision tree and rules. C4.5 and C5.0 algorithms are implemented in the Classification and Regression Tree (CART) system[35], a well-known system for learning classification and regression tree.

7.5.4 Rule construction

7.5.4.1 Classification rule algorithms A basic algorithm of this type is known as a *covering algorithm* which generates classification rules by generating one or more rules for each class that collectively cover all the training instances belonging to that class. More specifically, for each known class, the covering algorithm first builds a rule and removes all the instances the rule covers; then it recursively builds rules (using the previous step) for the remaining instances until none are left. If more than one rule is generated for a class, it is possible for the instances covered by the rules to overlap.

Like the divide-and-conquer algorithm which selects at each stage an attribute to add (as a node) to the tree, the covering algorithm at each stage selects an attribute to add to the antecedent of the rule under construction. However the covering and divide-and-conquer algorithms are different in two major ways: (i) while rules are constructed for a specific class, a decision made on a node of the tree is concerned with multiple classes, and (ii) the divide-and-conquer algorithm selects an attribute that best separates the classes, whereas the covering algorithm selects an attribute that has the highest accuracy (in associating the remaining instances with the specified class).

The procedure to construct a rule R for a class C, as implemented in the PRISM algorithm[36], is as follows [311]:

```
For each class C
   Initialise E to the instance set
   While E contains instances in class C
      Create a rule R with an empty left-hand side that predicts class C
      Until R is perfect (or there are no more attributes to use) do
         For each attribute A, not mentioned in R, and each value v
            Consider adding the condition A = v to the LHS of R
            Select A and v to maximise the accuracy p/t (where t is the
            total instances covered by the rule, and p are positive
```

[34]Overfitting is the phenomenon in which a classifier is tuned to the characteristics of the training instances rather than just the characteristics of the classes.

[35]CART. http://www.salford-systems.com/cart.php.

[36]PRISM. http://weka.sourceforge.net/doc/weka/classifiers/rules/Prism.html.

```
        examples among them)
    Add A = v to R
Remove the instances covered by R from E
```

The covering algorithm is an effective approach for simple and noise-free cases; however the algorithm also tends to overfit the training data and does not address the problem of missing values and numeric attributes. Practical rule learners have implemented advanced algorithms to generate good rule sets for noisy data. For example, the RIPPER algorithm implemented a global optimisation step on the set of rules generated that revises or replaces individual rules [311]. The global optimisation step was shown to improve both on the size and the performance of the rule sets. Frank and Witten [94] also proposed a method to combine the divide-and-conquer strategy and the covering rule. In this method, a single rule is generated by constructing a pruned decision tree for the current set of instances, then the leaf with the largest coverage is converted into a rule, and the tree is discarded. To make the process of generating a rule more efficient in this fashion, Frank and Witten built a partial decision tree, instead of a full tree, which can be done by integration of the construction and pruning operations in order to find a 'stable' subtree that can be simplified no further. Frank and Witter's algorithm has been shown to generate rules that are as accurate as rules generated by C4.5, and more accurate than those generated by RIPPER, yet with larger rule sets.

7.5.4.2 Constructing mining association rules At first glance, constructing association rules can be carried out in the same way as classification rules except that the consequent of a rule is an arbitrary combination of attributes, rather than a specific class. Though this approach is entirely valid at the theoretical level, it is practically infeasible due to the very large number of rules needed to express every possible combination of attributes with every possible combination of values as their consequents. As mentioned above, one approach to limit the number of association rules is to choose only rules that have high coverage and accuracy. To avoid wasting computing time and resources to generate all the rules, of which some are then removed, the *Apriori* algorithm [311] performs the necessary filtering before association rules are generated. The algorithm first requires that the input dataset is expressed in terms of *item sets* where each *item* is an attribute-value pair; for example, `petal_length=x` is an example of 1-item set, and `petal_length= x, petal_length=y` is an example of a 2-item set. The algorithm then proceeds through the following phases: (i) n-item sets ($n >= 1$) with coverage greater than the pre-specified threshold are generated (this in effect filters out all item sets with their coverage less than the threshold), and (ii) a rule, or set of rules, is generated for each item set. Since an item-set can be satisfied by more than one rule, the rules with the highest coverage are selected.

The Apriori algorithm described above, however, still requires many scans through the dataset to generate candidate item sets and to test whether the coverage exceeds the threshold. This can be inefficient in practice, especially when the dataset is large. The use of a *frequent-pattern tree*, or *FP-tree* can reduce the number of scans needed. The conceptual principle underlying an FP-tree is that the dataset is mapped to the FP-tree in a compact form: each node of a tree is a 1-item set whose frequency is greater than

the threshold. The FP-tree is then recursively processed to grow larger item sets directly, without having to test them against the entire database.

7.5.5 Instance-based learning

In instance-based learning, a set of input instances are stored as is. Therefore an instance-based learning algorithm does not need to build a classifier. Instead, its only responsibility is to devise a metric to determine the class for a new instance in the space of existing instances. The most popular metric for instance-based learning is Euclidean distance which measures the distance between instances i and j as:

$$\sqrt{(a_1^{(i)} - a_1^{(j)})^2 + (a_2^{(i)} - a_2^{(j)})^2 + \ldots (a_k^{(i)} - a_k^{(j)})^2}$$

To determine the class for a new instance, the distance between the new instance and every member of the training set is calculated, and the class of the instance with the smallest distance (*nearest-neighbour*) is predicted for the new instance.

Nearest-neighbour algorithms are simple and effective. However, the procedure is linear in the number of training instances, thus instance-based learning is often slow. Efforts have been made to devise more efficient methods to find nearest neighbours, including the following:

- *Reducing the number of exemplars*: the number of exemplars (the existing instances) can be significantly reduced if correctly classified instances are discarded and only those misclassified are stored. However, this method is ineffective in the presence of noise because noisy instances will be accumulated as exemplars.

- *KD-trees*: it has been demonstrated that nearest neighbours can be more efficiently found when the training set is represented as a *kD-tree*. A kD-tree is a binary tree in which every node is a k-dimensional point, with k being the number of attributes. Graphically interpreted, a kD-tree allows a fast (approximate) identification of the region of points 'containing' the point representing the new instance, and thus is more efficient than methods that examine all points to find the nearest neighbours.

- *Metric/ball trees*: though kD-trees significantly speed up a learning process, their performance diminish once the number of attributes is greater than 10. Other trees, such as *metric trees* and its instance *ball trees* have been recently proposed that can deal with a much larger number of instances. In particular, some metric trees can handle thousands of features successfully.

Like the Naïve Bayes method, each attribute in instance-based learning has the same influence on the decision. As such, instance-based learning is sensitive to data with noise. One simple solution to overcome this is to use *k-nearest-neighbour* strategy, which selects a group of k instances that are closest to the new instance and determine the class for the new instance via a majority vote. Other approaches dealing with noise include:

- *Pruning noisy exemplars*: the performance of an exemplar is measured based on the number of correct classifications and misclassifications carried out by the exemplar. Then based on two pre-specified thresholds, if an exemplar's performance is less than the lower threshold, it is deleted from the set of exemplars; otherwise if the exemplar's performance is greater than the higher threshold, it is used to classify new instances.

- *Weighting attributes*: to take into account the relevance and importance of individual attributes in performing classification, weights w_1, w_2, ..., w_m are incorporated into the distance metric as follows:

$$\sqrt{w_1^2 (x_1 - y_1)^2 + w_2^2 (x_2 - y_2)^2 + \ldots + w_m^2 (x_m - y_m)^2}$$

  The weights are determined based on measures of the contribution of each attribute (of each training instance) to the decision.

- *Generalising exemplars*: generalised exemplars are represented as hyperrectangles, a rectangular region of the high-dimensional instance space. When a new instance is classified correctly, it is merged with the nearest exemplar: a hyperrectangle is created if the nearest exemplar is a single instance, otherwise the hyperrectangle is enlarged to accommodate the new instance. Conversely, when the new instance is misclassified, the hyperectangle shrinks away from the new instance.

7.5.6 Clustering

The classic algorithm to generate clusters is known as *k-means* where k is a user-specified parameter which indicates the number of clusters being grouped. K-means works in numeric domains, and partition instances into groups of disjoint clusters as shown in Figure 3. The k-mean algorithm proceeds as follows. First, a user specifies a value for k. The algorithm then (i) selects k random instances, or *seeds*, as cluster centres, (ii) assigns every instance to one of the k clusters whose centre is closest to it based on Euclidian distance, (iii) calculates the *centroid*, or mean, of all the instances in a cluster, which becomes the new centre of the cluster, and (iv) repeats the previous three steps until the cluster centres are stabilised. A bad choice of seeds, which may occur in random seed selection, can impede both the speed and accuracy of the k-means algorithm. Therefore, instead of choosing seeds randomly, a seemingly better seed selection strategy is to spread the k initial cluster centers away from each other. This strategy is indeed implemented in *k-means++*: the first seed is chosen at random, then each subsequent cluster center is chosen from the remaining data points with a probability proportional to its distance squared to the point's closest cluster centre. Like instance-based learning, clustering algorithms can take advantage of kD-trees, or ball-trees, to find the nearest neighbour — the nearest cluster centre in this case — to speed up the clustering process.

7.5.6.1 Hierarchical clustering Another popular type of clustering is hierarchical clustering in which a hierarchical structure of clusters are formed either in a top-down or bottom-up fashion. The top-down approach involves building an initial cluster, then

recursively splitting the clusters until no more splits that can be done. Bottom-up, or *agglomerative*, clustering treats each instance as an independent cluster. Two clusters that are closest to each other are then recursively merged until the top-level cluster is formed. The distance between a pair of clusters can be computed in various ways including *single-linkage* (the distance between two closest members of the clusters), *complete-linkage* (the clusters that are considered closely related if instances in their union are relatively similar), *centroid linkage* (the distance between the centroids of the clusters), *average linkage* (average distance of each pair of members of the two clusters), and *group average* (average distance between all members of the merged cluster).

7.5.6.2 Subspace clustering Subspace clustering is a relatively new concept emerging from the publication of the first subspace clustering algorithm, *CLIQUE*, by IBM researchers. Since then many other subspace clustering algorithms have been developed and studied (see Figure 5).

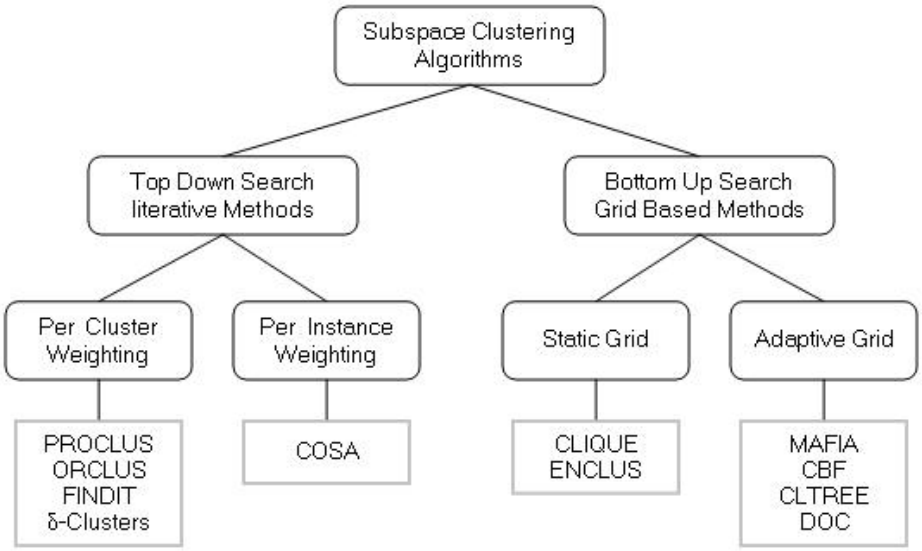


Figure 5: *Subspace clustering algorithms [209].*

Traditional clustering techniques discover clusters of instances by examining all the features of the instances. In a high-dimensional space (i.e., each instance contains a large number of features), distance measures become less effective — given a large enough number of dimensions, points in a multidimensional space (or instances) have almost equal distances from each other — resulting in a significant decrease in the clustering algorithms' performance. Common ways to address this problem apply methods from feature selection (which select the most relevant subset of features) or feature transformation (which transform the original feature vector that representing an instance to a new vector with smaller size)(see Section 6.2.2). Though these methods are very effective in reducing the number of dimensions, they face difficulty when dealing with the following scenarios:

(i) there are multiple equivalently relevant subsets of features, and choosing one of them (as in feature selection) will not be an optimal solution, and (ii) there are many redundant and irrelevant features, and thus blindly merging them into a new feature vector (as in feature transformation) will also be suboptimal. One solution to address this problem is to perform subspace clustering: clustering is formed based on different subsets of features (or different subspaces of the high-dimensional data). Another advantage of subspace clustering is that it is most suitable in scenarios where feature vectors are related in different ways in different subsets of features, and one is interested in explicitly disclosing this type of relationship.

7.5.6.3 DBSCAN While the above clustering approaches are based on some kind of similarity or distance measure, the DBSCAN algorithm [91] is based on the notion of *density*. There is more than one way to compute density. The traditional centre-based approach estimates density by counting the number of instances within a specified radius of a particular instance. Based on this definition, an instance can be classified as a *core instance* (the instance that is inside the dense region), a *border instance* (the instance that falls on the edge of the dense region) and a *noise instance* (the instance that is neither inside or on the edge of the dense region). A type of partial clustering, DBSCAN locates regions of high density that are separated from one another by regions of low density, by performing the three following steps: (i) assigning two core instances that are close enough (i.e., within a specified radius of one another) to the same cluster, (ii) assigning any border instance that is close enough to a core instance to the same cluster of the core instance, and (iii) discards noise instances. Though DBSCAN is simple, it is an effective density-based clustering algorithm.

Case-based reasoning Closely related to clustering with respect to learning algorithms is case-based reasoning. Similar to clustering, case-based reasoning performs reasoning by remembering past experiences: it saves and indexes encountered cases (descriptions of problem and their respective solutions) in a library. When solving a new problem, it retrieves the case with a problem most similar to the new problem using some kind of similarity metric such as nearest neighbour, and the solution in the retrieved case is adapted to generate a solution to the new problem. This adaption phase is the most difficult part of case-based reasoning. Currently, there are no general rules to perform case adaption, and this task is generally left to human expert or automated using domain-specific rules.

7.6 Evaluation of machine learning schemes

A wide variety of machine learning algorithms have been proposed with different strengths and drawbacks. Empirical testing and evaluation of the prediction results produced by a range of learning algorithms is in many cases the most efficient way to select the algorithm most suitable for the problem at hand. Evaluation of a machine learning algorithm is often carried out based on the measures of *accuracy*, *precision* and *recall* defined as below (where *TP*, *TN*, *FP* and *FN* denote *true positive*, *true negative*, *false positive* and *false negative*).

- Accuracy is measured as the ratio of the number of instances whose class was correctly identified to the total number of instances:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}}$$

- Precision is measured as the ratio of the number of instances correctly assigned to the class to the total number of instances assigned to the class:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

- Recall is measured as the ratio of the number of instances correctly assigned to the class to the total number of instances written by the class:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

WEKA: a machine learning workbench for data mining The most popular open-source data mining tool, WEKA facilitates experimentation and evaluation activities for a large collection of state-of-the-art learning algorithms[37]. Written in Java, WEKA was developed by a research group at the University of Waikato, New Zealand, and is publicly available for use under the terms of the GNU General Public License. WEKA offers an efficient and convenient way for users to quickly try out existing learning methods on new datasets, and provides extensive support for an end-to-end experimental process, including preparation and preprocessing of input data, evaluation of learning output, and visualisation of data.

[37] WEKA. `http://www.cs.waikato.ac.nz/ml/weka`.

8 Attribution methodology

As discussed in Section 2.4, cyber crime poses many challenges for attribution. Spear phishing, due to its advanced nature, often magnifies these obstacles: an attack of this type is carried out through multiple steps (e.g., Target, Lure, Hook and Catch) and possibly many stages (e.g., via stepping stones and zombies), by means of different social engineering and technical techniques (e.g., IP, email and web spoofing), and possibly with the assistance of multiple actors (e.g., malware developers, web designers, rogue network owners and zombie-net controllers). Given that a spear phishing scheme could be crafted in such an advanced form, attribution of a spear phishing incident is by no means trivial. Not only is achieving the ultimate goal of attribution tremendously hard, but resolving how to practically approach this attribution problem presents a significant challenge in itself — which pieces of evidence to process? How to process them? How to relate different pieces of evidence? How to integrate the attribution outcomes resulting from the analysis of different pieces of evidence? How can one tell if the method chosen to address any of these problems works? These questions suggest the diversity and complexity of the various threads that must be disentangled, requiring a sound and well-defined methodology to guide attribution practitioners towards their goal. Without a well-defined methodology, the whole attribution process is likely to be obscured; without a sound methodology, the validity of the entire attribution results would simply be questioned.

As crime in cyberspace much resembles crime in the physical world, it may be beneficial to speculate the literature of *offender (or criminal) profiling* — a research area that specialises in investigating techniques relating to identification and apprehension of criminals — in a quest for a reliable methodology (if any) toward solving this jigsaw puzzle.

8.1 Review of offender profiling

Profiling is used by forensic examiners as a main tool to apprehend the unknown perpetrators. In this sense, an offender profile is a collection of inferences about the characteristics of the person who commits a crime (or a series of crime) based on the evidence left on the crime scenes. Offender profiling generally does not aim to pinpoint the exact perpetrator — even though it is ideal if such target can be achieved. Rather, it describes the characteristics of a class of persons who are likely to commit such a crime in order to assist law enforcement in identifying the perpetrators. The role of offender profiling is prevailing when traditional investigative methods fail to identify an offender, in which case profiling is a valuable tool to be leveraged by law enforcement investigators.

From the knowledge point of view, a offender profile can be *nomothetic* (which contains universal characteristics) or *idiographic* (which contains the characteristics of the unique individuals) [291]. A nomothetic profile is typically generated by an inductive method (see below) on groups to infer typical, common, and averaged characteristics of a group of offenders [291], while an idiographic profile is generally constructed by a deductive method (see below) on individuals in order to determine the unique characteristics of the particular offender(s) committing a particular crime.

From a logical point of view, profiling is essentially a reasoning process, and an offender profile is a product of evidence-based reasoning. As such, any extant offender profiling

method would belong to either of the two categories: *inductive profiling* and *deductive profiling*. Both of these types of reasoning involve inferring a *conclusion* from one or more *premises*, *albeit* in different manners, and each has its own merits and drawbacks. In the former case, offender profiling is concerned with a type of inference that proceeds from a set of particular premises to a general conclusion. For example, if most of the people who committed a violent act are male (premise 1), and the majority of them are teenagers (premise 2), it is likely that, using an inductive method, the offender is a male teenager (conclusion). The strength of inductive profiling lies in its capability to generate new knowledge (in the form of hypotheses), but inductive inference is itself not infallible — the derived conclusion is not always valid irrespective of the validity of the premises. In the latter case, offender profiling involves inferring a particular conclusion from general premises. For example, based on a witness's statement the offender must have a facial scar (premise 1), but the suspect has no scar (premise 2), it is therefore the case that the suspect is not the offender (conclusion). Different from the inductive method, deductive profiling offers infallible inference at the cost of being unable to generate any new knowledge (since the derived conclusion is in fact implicitly contained in the premises). Furthermore, deductive reasoning is *monotonic*, i.e., an addition of premises does not revive or withdraw the previously derived conclusions, thus an offender profile constructed in this manner may require regular updates in light of new evidence.

Profiling has been long practiced in crime investigation, and a range of methods have been investigated and published. More specifically, offender profiling methods have been studied in the context of serial homicide, rape, arson, homicide, burglary, child crimes and others [81]. At the current time, the most prevalent profiling method is *Criminal Investigative Analysis*(CIA) developed by the Federal Bureau of Investigation (FBI)[38]. CIA is a nomothetic and inductive profiling method which was originally formulated from interviews of a group of offenders in the 1970s. One of the outcomes of this study is the so-called organised/disorganised dichotomy which has been widely applied in practice. The dichotomy assists profilers in classifying a crime scene as disorganised or organised; and subsequently allows them, based on the evidence on the crime scene, to classify offenders in terms of their level of sophistication, planning and competence.

Despite its relatively long history, the majority of work in the literature of offender profiling remained more an art than a science and tended to target specific crimes, and/or contain various flaws [121, 291]. In traditional methods, a forensic examiner (or a *reconstructionist*) typically practises profiling as an art. Based on his/her experience which is believed to reflect reality, the reconstructionist forms a theory of what happened from the overall crime scene, and subsequently searches for evidence to support his/her theory. No matter how the reconstructionist claims to be objective, conducting profiling in this manner easily falls prey to subjectivity, or *bias*, a phenomenon that is formally termed as the *observer effects* (a.k.a., *context effects*, or *expectancy effects*) [291]. Observer effects refer to the influence of one's state of mind, either consciously or subconsciously, in interpreting a piece of evidence. While the conscious (*overt*) form of bias bears a negative connotation, it is actually less threatening than the subconscious (*covert*) form of bias (which is much harder to be detected and eliminated due to its salient existence). It is this covert bias that potentially jeopardises attribution results from traditional methods. In addition, a large amount of profiling work in the literature concentrates on particular

[38]CIA is also informally referred to as the *FBI profiling* method.

types of crime, which makes it either difficult or inapplicable to adapt the methods to other types of crimes. Even more problematic, some profiling methods may contain flaws in themselves, and thus they potentially yield unreliable results. For example, the CIA method, especially the organised/disorganised dichotomy, has been severely criticised by a number of researchers in the field (e.g., Canter [42], Depue [74] and Turvey[291]):

- the dichotomy suffers from 'the 'Hollywood effect', whereby loosely formulated and often unsubstantiated theories and models are featured in widely disseminated movies and given extra creditability by such broadcast [42],
- 'FBI profiling has failed detailed peer reviews of both casework and publications' [291],
- replacing the proven investigative techniques with CIA 'would be counterproductive to the goal of identifying the unknown offender' [74], and
- 'the dichotomy has been debunked as false and lacking real-world applications' [291].

Among the flaws identified for the CIA method are: the extreme binary classifications of organised and disorganised, the over-simplicity in interpreting crime scene evidence, the general infeasibility of discriminating the origins of behaviours that can result in 'disorganisation' at a crime scene, the failure to account for an offender's development over time, and the failure to facilitate offender classification on *modus operandi* considerations [291].

The literature of offender profiling (as presented above) strongly indicates the necessity of a profiling model that mitigates, and possibly eliminates, extant problems in the current profiling methods. Such a model is expected to comprise the following components: development of a theory about offender profiling, hypotheses generation, operationalisation of methods used in profiling, and empirical validation [121]. In other words, what one desires is *a scientific model of profiling*. From this review, it appears that a complete scientific model for offender profiling does not exist. Yet, research efforts toward this goal have been made in order to rectify limitations of current profiling practice. Most representative among the mentioned work are *Investigative Psychology* (IP) by Canter [42], *Behavioural Evidence Analysis* (BEA) by Turvey [291], and the proposal of a scientific model of profiling by Hicks and Sales [121].

Investigative psychology [42] was studied by Canter as a nomothetic, inductive model for offender profiling. Constructing offender profiles based on a scientific foundation, investigative psychology aims to advance the state of the art of offender profiling. However, Canter's model has a number of limitations. According to Turvey [291], the major weakness of investigative psychology lies in its lack of practicability — the model is 'purely academic' and the produced results are entirely abstract and theoretical. According to Hicks and Sales [121], Canter's model does not address the criteria for a scientific model in a satisfactory manner: (i) the lack of comprehensiveness and conceptual clarity, (ii) the generation of hypotheses is in turn constrained by the lack of clarity in the theory, (iii) the various problems relating to operationalisation of profiling methods such as the lack of a clear explanation of terms, the inadequate research method in establishing an explanatory relationship between offender actions and offender characteristics, (iv) the neglect of addressing profiling in practice, and (v) the failure to address whether any of Canter and colleagues' findings can successfully be applied to real, unsolved cases.

In contrast to Canter's work, Behavioural Evidence Analysis (BEA) proposed by Turvey [291] is idiographic (the model studies the aspects of individual cases and offenders, not groups of similar cases and offenders) and deductive (the model is based on deductive logic, critical thinking, and scientific methods). BEA involves examination and interpretation of physical evidence, victimology and crime scene characteristics. Although the use of scientific methods in profiling was strongly emphasised by Turvey [291], BEA, at its core, contains an ambivalence between science and art: the analysis of physical and behavioural evidence is based on scientific methods, but the translation of evidence into the offender's characteristics is essentially an artful process [121]. Therefore BEA incorporated science into profiling, but the model is in itself not scientific [121].

Motivated by an observation that there is no single model that is sufficiently comprehensive or accurate, as well as no single model that is adequate for use as a template, Hicks and Sales [121] worked toward the development of a science of profiling and proposed guidelines that facilitate the development of such a model. A gross summary of the model is given below[39].

- Regarding the goal of offender profiling, Hicks and Sales [121] focused on developing a model aimed at gathering and analysing crime scene information to assist law enforcement in identifying unknown perpetrators.
- With respect to the specificity of the identification, Hicks and Sales attempt to build a science toward the ideal goal of profiling — that of identification of a single individual (rather than a class of individuals) — which is currently beyond the capability of the existing profiling methods.
- Regarding operationalisation of the profiling methods, Hicks and Sales [121] do not rely on experts' experience and skill set, instead the researchers proposed a scientific approach which contains the three major steps as presented below.
  - *Step 1: evaluating crime scene evidence*: in this step, pieces of physical evidence at the scene are collected and examined by forensics investigators. For example, strands of hair would be sent to a DNA forensics laboratory to determine their owners; a written letter left on the scene would be scrutinised, based on both the handwriting and the content of the letter, to identify a suspect. The goal of this phase is to establish any possible linkage between the evidence and the offender. In a few cases, the results of crime scene evidence analysis can be directly linked to the true offender (e.g., a fingerprint or DNA sample of the culprit is obtained). In many other cases, this direct link is not available due to a lack of valid identifying evidence (e.g., the obtained driver license is forged and the car used by the offender is stolen from an innocent person). If an offender is not able to be determined via crime scene evidence analysis, then *crime scene reconstruction* is needed. In this phase, each piece of evidence is studied and interpreted in the context of the crime and in relation to the other pieces of evidence. In this way, not only is a piece of evidence more informative, but also a more complete picture of the crime can be obtained, assisting an investigator in construction of hypotheses about the offender.

[39] More of the model developed by Hicks and Sales will be discussed in Section 18.1.

- *Step 2: relating information from crime reconstruction to the motives, personality characteristics, and behaviors of known offenders*: unless direct links between pieces of crime scene evidence and offenders can de found, attribution practitioners must examine the potential relationships between the types of crime scene evidence and offender actions derived from crime reconstruction and other offender characteristics that may assist in apprehending an unknown perpetrator. According to Hicks and Sales, the three groups of offender characteristics that can possibly be predicted are motives, personality, and behavior.
- *Step 3: testing profiling predictions*: this step should contain activities to test the predictive power of relationships among crime scene evidence, motive, personality, and offender behaviors.

While profiling of conventional offenders is sufficiently mature to pave the way toward a complete scientific model, (behavioural) profiling of cyber offenders is somewhat at its beginning. Besides the comprehensive efforts to characterise cyber adversaries by Parker and colleagues[40] [231], it appears that the literature of cyberspace offers only a few miscellaneous research efforts that have simply scratched the surface of offender profiling (c.f., [140, 177, 224]). Indeed, the importance of adapting conventional profiling methods to cyber offenders was only recently recognised (c.f., [29]). More specifically, Bongardt [29] explicitly called for cooperation between researchers and practitioners, as well as an integration of techniques, in the fields of Information Security (IS) and conventional criminal profiling (in particular, CIA), toward the development of behavioural profiles of computer network intruders — a first step for the new paradigm of cyber offender profiling.

8.2 Spear phishing attribution methodology

This survey presents attribution methods pertaining to spear phishing according to the scientific profiling approach proposed by Hicks and Sales [121] (see above). More specifically, Sections 9, 10, 11, 12, 13, 14, and 17 are devoted to discussions of methods and techniques relevant to step 1 of the approach. However the evaluation of crime scene evidence will be carried out in the context of cyber crime, and the collection of specific attribution/forensic methods carried out in this step are those specifically applicable to spear phishing. Section 18 discusses how to relate information from crime scene evaluation/reconstruction to the motives, personality characteristics, and behaviors (step 2 of the approach) of conventional offenders, as well as how to characterise offenders specifically in the cyber crime context. Step 3 of the approach by Hicks and Sales [121] involves methods for testing a profiling model. The methods require the existence of the appropriate samples of convicted offenders which are unlikely to be available in the case of spear phishing. For this reason, this step is excluded from discussion in the survey.

[40] Characterisation of cyber adversaries will be discussed in Section 18.2.

9 Attribution of spear phishing evidence

Physical evidence cannot be wrong... Only in its interpretation can there be error.

(ellipsis added)

Paul L. Kirk (1974)

As previously mentioned, the profiling approach studied by Hicks and Sales [121] is used in this survey to systematically investigate the spear phishing attribution problem. This section together with Sections 10, 11, 12, 13, 14 and 17 are devoted to addressing the first, also the most critical, phase of the investigation: analysis and attribution of spear phishing evidence[41]. In this phase, pieces of digital evidence — a special type of physical evidence — left at the scenes are scrutinised in order to identify, and/or to infer as much information as possible about, the source of a spear phishing attack. In this section, I attempt to categorise and define such evidence and crime scenes in the context of spear phishing.

A crime scene is traditionally defined as 'a location where a criminal act has taken place' [291]. In this survey, I adopt a more generic view of crime scenes — crime scenes as *comprehensive sources of investigative information available to an investigator* [121] — with the purpose to accommodate crimes in the digital world. Recall that a typical spear phishing scheme consists of the four components: Target, Lure, Hook and Catch, I identify the crime scenes in accordance with the adopted definition and in correspondence to the components of a spear phishing attack as following:

- **(retrospective) crime scene 1**: source of information regarding gathering information about a victim (Target),
- **crime scene 2**: source of information regarding contacting and tricking the victim (Lure),
- **crime scene 3**: source of information regarding illegitimately capturing information (Hook), and
- **(prospective) crime scene 4**: source of information regarding exfiltration of stolen information (Catch).

Herein, I refer to Target and Catch as 'retrospective' and 'prospective' crime scenes, respectively, due to their non-conventional nature: both the crime scenes are not, at the time, presently observable to the investigators (see Sections 10 and 14 for further discussions)[42].

[41] Analysis and attribution of spear phishing evidence corresponds to evaluation of crime scene evidence of the approach studied by Hicks and Sales [121].

[42] For the sake of simplicity and clarity, it is assumed that attribution will be performed after a spear phishing email has been received and detected. Of course, in practice, attribution can occur at any point after the target component commences, for example, after a spear phishing attack has successfully compromised a computing system. In such a scenario, investigators might need to leverage additional techniques in network and computer forensics in combination with methods presented in the survey to perform attribution (see Section 15.2 for a real-life example of a relevant situation).

Table 2: *An illustration of pieces of spear phishing evidence together with their type and the attribution level(s) they are most likely applicable to.*

Crime scene	**Ownership evidence** (attribution levels I, II)	**Individual characteristic evidence** (attribution level III)	**Class characteristic evidence** (attribution levels IIIb, IV)
Target (Section 10)			the extent to which the spear phishing is tailored to the victim
Lure (Section 11)	IP address, email address, sender's name, watermark (if any)	writing style	linguistic features, general metadata, social engineering techniques, deception techniques, technical techniques
Hook — malicious software (Section 12)	watermark (if any)	programming style	programming features, general metadata, malware behaviour
Hook — (spear) phishing website (Section 13)	IP address, domain name, watermark (if any)	writing style	general metadata, social engineering techniques, deception techniques, technical techniques
Catch (Section 14)	IP address, email address		methods to exfiltrate data from Hook

Pieces of evidence contained in the crime scenes are categorised into the following types:

- *ownership evidence* refers to the type of evidence that identifies an entity or an individual. Examples of ownership evidence include fingerprints and driver's licences (in the physical world), and IP addresses and email addresses (in cyberspace).
- *characteristic evidence*[43] refers to the type of evidence that allows one to infer information about the characteristics of those responsible for a spear phishing attack. A piece of characteristic evidence can be either:
  - *individual characteristic evidence* which potentially reveals characteristics of a single individual (such as writing style and programming style), or
  - *class characteristic evidence* which captures information associated with a class of individuals (such as preferences for development tools and programming languages).

In terms of attribution, ownership evidence in cyberspace is critical to attribution level I and II, individual characteristic evidence potentially provides important clues for attribution level III, and class characteristic evidence is useful for both levels III and IV of

[43]Please note that the definition of characteristic evidence used in this survey is not necessarily the same as those found in the literature.

attribution (please refer to Section 2.3 for a description of different levels of attribution in cyberspace). It is important to note that the different types of evidence are not necessarily disjoint. For example, characteristic evidence subsumes ownership evidence: if a piece of ownership evidence is sufficiently strong to establish a link between the evidence and an individual (e.g., the spear phisher), that will be ideal; otherwise, ownership evidence can serve as individual characteristic evidence as part of the profile for the attacker. In a similar manner, individual characteristic evidence can also be overlapped with class characteristic evidence (e.g., the structure of a program can be considered as both individual and class characteristic evidence). Table 2 displays pieces of spear phishing evidence discussed in this survey together with their respective type and their associated level(s) of attribution.

In order to achieve a consistent representation, attribution methods pertaining to each of the components of a spear phishing attack will now be discussed in Sections 10, 11, 12, 13 and 14 along one or more of the following aspects of attribution:

- *source tracking* based on ownership evidence,
- *authorship attribution* based on individual characteristic evidence, and
- *adversary class identification and characterisation*[44] based on class characteristic evidence.

[44] A class (or group) of adversaries should be distinguished from an adversarial group in the level IV of attribution. Members of an adversarial group are strongly cooperated in an attack, and collectively represent an adversarial entity. In contrast, adversaries belonging to the same class are those that may not be aware of each other, but have common characteristics or modus operandi.

10 Attribution of evidence in the target component

The target component occurs some time prior to any detection of a spear phishing attack, hence evidence from the related crime scene may no longer be available or accessible. However, the existence and nature of the target component may be revealed in part through the extent to which the email is tailored to the victim. Analysis of this remnant evidence in the lure component potentially offers the following sources of valuable information:

- the class of adversaries an adversary belongs to, and
- the types of victim as well as types of information the adversary is after.

Table 3 displays an illustration of such predictions.

Discussion Due to the retrospective nature of Target, there is a limited scope to infer information regarding the adversary from this component. Perhaps, the most likely approach to carry out this task is to devise and execute a set of inductive/deductive rules (such as those illustrated in Table 3). Nevertheless, it is possible to gain a better insight into the adversary through the target component if an organisation is willing to take a proactive approach. This approach involves the organisation planting ahead of any targeting activity a wide range of decoy elements (possibly with varying access restrictions) containing information pertaining to the organisation and its employees. Then the use of one or more of the decoy elements in a spear phishing email is likely to reveal, at least in part, the mechanism utilised to harvest information about the victims, as well as the effort, motivation and capability of the adversary behind a spear phishing attack.

Table 3: *An illustration of predicting information about an attacker based on evidence from the target component. It should be noted that the above examples are presented for the purpose of illustration, and thus their validity needs to be verified by domain experts.*

Evidence relating to the target component	**Predicted class of adversaries**	**Predicted targeted victim**	**Predicted targeted assets**
Victim-tailored information consisting of plain facts that are explicitly and widely available on public websites/repositories	Opportunistic attackers (using web crawlers and alike to learn about the victims)	Random victims (whose details are randomly collected by the tools an attacker has at his/her disposal)	Personal credentials for the purpose of financial gain
Victim-tailored information being very convincing (that is likely resulted from a careful processing and integration of different sources of information available on public websites/repositories)	Targeted/ sophisticated attackers	Specific victims	Personal credentials /sensitive information for the purpose of financial gain or espionage
Specific information about an organisation included, and more than one person in an organisation receiving similar emails	Targeted attackers/ adversaries	Specific organisations	Personal credentials/ sensitive information for the purpose of financial gain or espionage (the attacker may aim to exfiltrate the customer payroll information or intellectual properties of a specific company)
Specific information about an organisation included, and the victim having an interesting role/rank in the organisation	Targeted attackers/ adversaries	Specific personnel in a specific organisation	Specific assets for the purpose of financial gain or espionage
The included information about a specific victim (or an organisation) is not available on the public websites/repositories, and *timing* possibly included	Insiders (which can be human or malware)	Specific victims/ specific organisations	Personal credentials/ sensitive information for the purpose of financial gain or espionage

11 Attribution of evidence in the lure component

The lure component of a spear phishing attack is typically in the form of an email message that appears to come from a legitimate or familiar sender, and convinces a user to perform an unwitting act. There are usually two sources of information one can obtain from a standard email message: information about the email message (e.g., email metadata provided by the email header), and information communicated between the sender and the receiver (contained in the email textual content). The email header provides information about the email such as the containment of an attachment as well as the source and the travelling path of the email message, and the email textual content can indirectly provide various types of information about the sender or intent. It is therefore natural and important to examine both these sources of information in an attempt to attribute a spear phishing email. More specifically, attribution methods discussed in this section include: email source attribution (based on ownership evidence such as IP addresses and email addresses), identification/characterisation of a single adversary (based on the writing style reflected in the email textual content), and identification/characterisation of an adversary class (mainly based on the email metadata).

11.1 Email source attribution

In the trustworthy world, the source of an email can be easily identified based on the ownership evidence (i.e., email/IP addresses and sender's name) provided in the email header. At the application layer, the source of an email would be the user account associated with the sender's email address or name. At the network layer, the source of an email would be the machine associated with the sender's IP address. However in the context of spear phishing, information relating to the email source at the application layer (i.e., email addresses and senders' name) is almost certainly falsified via *email spoofing*: the email is often claimed to have come from human resources personnel in the organisation, a relative/friend, or a social network/corporation with which a victim has an association. In the same vein, information regarding the email source at the network layer (i.e., IP addresses) is often fabricated via *IP spoofing*. In a worse-case scenario, if any mail server on the travelling path of an email message is compromised, or the email is sent from a phishing mail server, all information provided in the email header is unreliable because entries in the header can be modified, deleted or entered with little effort.

It is obvious that identifying the sender of a spear phishing email based on the header information is not a viable approach. This calls for a survey of methods that track the source of an email by tracing backwards the physical trail left by the email on the data communication network. Since email is an application that uses Mail Transfer Agents (MTAs) and Simple Mail Transfer Protocol (SMTP) to transfer electronic messages and physically operates over the Internet, email source tracking, in principle, can be performed either at the application layer (*email tracking*), or at the network layer (*IP traceback/tracking*).

Email tracking Minimal efforts have been made on tracking email messages. The few examples of methods to track email include tracking legitimate email communication

between a company and its customers [84, 108], and tracking general electronic messages via the use of a registered appliance (specific to an Internet Service Provider, or ISP) to mark, record and track the messages [251]. However, most of the work consists of research ideas and proposals.

IP traceback IP traceback – henceforth traceback — methods are aimed at identifying the source of an attack in the face of IP spoofing. It is based on the rationale that irrespective of where a spoofed network packet claims to be sent from, there *exists* in reality a *physical point* in the data communication network where the packet enters the network, and a *physical path* over which the packet traverses to reach the victim. If this path is to be followed, starting from the destination point and tracing backward as far as possible, it is possible to uncover the true identity of the source. This idea has attracted research attention for more than a last decade and resulted in a large amount of published work. Since traceback methods could be generalised or adapted for use in tracking a spear phishing email message, a survey of relevant traceback methods is presented in the next section. To keep the following discussion generic, *messages* are used to refer to both network packets and email messages, *protocols* to refer to those at both the network/transport layer (e.g., IP/TCP and ICMP) and application layer (e.g., SMTP, FTP and HTTP), and *routers* to interchangeably refer to network routers and mail servers.

11.1.1 Traceback methods

The impetus for much research work in the literature of traceback can be attributed to battles against distributed denial of services (DDoS) attacks. As such, many source tracking methods take advantage of the unique characteristics of DDoS and thus are not applicable to the attribution of spear phishing emails. However there are a number of methods that could be generalised (applied at the level of MTAs and SMTP), or adapted (used at the level of network packets but adapted to track email messages), for use in the spear phishing context. The methods to be presented are grouped into four categories: *logging and querying*, *marking*, *filtering*, and *stream matching*[45].

11.1.1.1 Logging and querying In a logging scheme, routers (or MTAs) record information about the network traffic passing through it. Ideally, the routers would keep a log of every traversing packet. In this case, the task of reconstructing the path of a message only involves the end host querying the relevant routers to find out the necessary information. However this brute-force approach is not viable in practice due to the undesirable implications for privacy, and significant processing and storage overhead imposed on every router. To mitigate this problem, a router can adopt either of the following strategies: (i) limit the number of messages to be logged, and/or (ii) limit the amount of information from each message to be logged [308]. In the former approach, a router can be instructed to log only network traffic destined for pre-determined *sensitive* destinations (e.g., a critical department), and/or dynamically log network traffic that seems *suspicious*. Provided that the number of interested destinations is small, this would reduce the amount of data stored in the routers to a significant degree. In the latter approach, for the purpose of

[45]The structure of this section is loosely based on the literature survey by Wheeler [308].

attribution, it is sufficient to store only a fragment of a message, a hash of the message, or a hash of a fragment of the message. Recording a hash of a traversing message not only reduces the amount of data that needs to be stored on the routers, but also avoids the related privacy concern. For instance, (i) Cisco routers provide a capability to record information about data passing through the router, including source/destination IP addresses and ports, number of packets/bytes, IP protocol, and TCP flags [257]; (ii) Source Path Isolation Engine (SPIE) is an efficient way to store IP packet hashes using a Bloom filter [270], together with its high-performance hardware implementation [259]. Proposals for traceback protocols are also available at [233]. Once data from network packets is logged on routers/mail servers, querying can be performed to obtain the necessary information. Querying to date is mostly done manually, which is inefficient in terms of response time and human resource. To this end, a number of protocols have been proposed for automated querying of logging systems (c.f., [283, 11, 12, 13]) and of hosts (c.f., [50]).

11.1.1.2 Marking Marking can be considered a message-centric logging scheme. Instead of a router, or an MTA, keeping a log of messages that it forwards, each message in this scheme carries a log of routers it has visited (c.f., deterministic router stamping at [78]). Indeed, existing MTAs already implement this marking capability as evident in an email header. This type of information in the email header is generally not authenticated, and thus is unreliable when deception is involved. This problem was rectified in the Deciduous approach [52, 51] in which IPSEC authentication headers were used to identify at least some of the routers traversed. To avoid rapidly increasing the packet size by appending logging information to the message header, Dean [72] proposed *algebraic encoding*: a method that uses algebraic techniques to efficiently encode (in terms of space) routing information so that it can be stored in the IPv4 'fragment id' field. Other marking techniques such as probabilistic packet marking [261, 262, 271, 184], (probabilistic) router stamping [78], and separate message transmission [23, 196] were also investigated. However they require a large amount of traffic generated from the offender to reach the defender and thus are not relevant in the context of spear phishing.

11.1.1.3 Filtering Different from logging and marking techniques, filtering methods allow for an identification of the attacker network (or a range of networks) rather than an individual host. One such filtering method is *network ingress filtering*[46] (c.f., [96]). Network ingress filtering requires that all messages entering a network have a valid 'source' value for that network entry. Provided that ingress filtering is properly implemented at every entry point of a network, messages entering the network with spoofed IP addresses outside the valid range will be filtered. In the case that a malicious message successfully evades filtering by having its spoofed IP within the valid range, network ingress filtering significantly narrows down the possible location of the origin of the message. It is claimed that the attribution problem might be solved if ingress filtering is completely implemented on the Internet [143]. Likewise, if an MTA only transfers emails from a list of given trusted senders, identifying the source of email messages would be a much easier task. Major advantages of ingress filtering over other traceback approaches include: (i) no performance and storage overhead imposed on routers, (ii) no undesirable impact on privacy, (iii) ease of

[46]Network ingress filtering is also known as *egress filtering* (where filtering devices are positioned at exit points of the network) or *firewall filtering* (where filtering mechanisms are deployed within firewalls).

implementation due to the existence of filtering capability within routers, and (iv) ingress filtering can be deployed in an incremental manner. However, ingress filtering needs to be deployed at every entry point of the network (or every point at the outer edge of the Internet) and incorporate mobile IPs for it to be fully effective, otherwise the efficacy of the method would be much reduced.

11.1.1.4 Stepping stone attack attribution It would be unlikely for a sophisticated attacker to initiate an attack from his/her own machine. To obscure his/her identity, the attacker often steps through a chain of machines (stepping stones) before launching an attack. In such a scenario, even if all the traceback methods discussed above are successfully carried out, the result is usually a stepping stone machine owned by an innocent user. To address this problem, a number of research work in the literature studied methods to detect such stepping stones based on *stream matching* — a technique that involves examining and matching streams of data coming in and out of a host, or a network, in order to reconstruct a chain of connections starting from the stepping stone closest to the victim back to the attacker's machine. Historically, early stream matching methods targeted *telnet* and *rlogin* due to the fact that these services do not change the content at each hop (terminal to terminal connection) as some encrypted services do. Recent stream matching methods expand their practical applicability by dealing with both unencrypted and encrypted services such as *SSH*. In a closed and tightly controlled network, stream matching can be host-based where stream monitoring and matching capability is implemented on each host. However, in open networks such as the Internet, network-based stream matching where the capability is implemented on a number of monitoring machines around the network is more suitable and widely-accepted.

Stream matching can be done via content (*content-based stream matching*), header (*metadata-based stream matching*) or timing (*timing-based stream matching*). Stanoford-Chen [280, 281], in his ground-breaking paper for content-based stream matching, used *thumbprint* — a short piece of information that summarises the content of the connection — to capture the characteristics that are unique and invariant to streams belonging to a chain of connections. In this way, the connection path from the attacker through a series of stepping stones can be determined by matching the thumbprints from a pool of streams collected by monitoring machines throughout the network. In his/her metadata-based stream matching approach, Yoda [313] first required the time stamps and headers of the packets to be recorded at many places on the Internet. Yoda then matched streams by computing a deviation (i.e., the difference between the average propagation delay and the minimum propagation delay of two connections) for each of the packet streams at various Internet sites from the given stream. This approach is based on the supposition that the deviation for two unrelated connections is large enough to be distinguished from the deviation of connections in the same connection chain. Both the content-based and metadata-based stream matching approaches share a major shortcoming: they do not work in the presence of encryption. This short-coming motivated research on timing-based stream matching. The first timing-based stream matching method, proposed by Zhang and Paxson [320], is based on the correlation of the ON and OFF periods of different connections. A stream is considered to be in an OFF period when there is no data traffic on a stream for more than a time period threshold, otherwise it is in its ON period. Their approach is based on an observation that two streams are likely in the same connection

chain if their OFF periods end, or their ON periods start at similar times. Since the ON and OFF periods can be measured externally, the method holds even when encryption is used. This work triggered a series of stream matching methods based on timing, including the approach based on inter-packet delays (IPDs) of both encrypted and unencrypted data streams [300], and approaches dealing with techniques to disguise traffic such as insertion of timing perturbance and/or superfluous packets (chaff) [28, 80, 321]. It is important to note that stream matching methods are not sufficient to detect zombies due to the fact that a zombie is often instructed to transform and/or delay the attacker's communication, and thus it is generally much harder to match the output stream from a zombie with the corresponding input stream. There does not appear to be a stream matching method that is specifically devoted to reconstructing a chain of connections via zombies.

There are other types of traceback methods compiled by Wheeler [308] such as *input debugging performance* which involves providing upstream routers with an attack traffic signature or a destination address and asking them to report when they see that pattern next time, and *reconfigure and observe network* which involves reconfiguring the network and observing any changes to identify the source of the attack. However, these traceback methods are only applicable for detecting the source of an ongoing attack involving large amounts of traffic and thus not relevant to the attribution task under discussion.

11.1.1.5 Discussion From a research perspective, the literature of traceback is sufficiently mature, providing a wide range of methods to traceback to the source of an attack, possibly traversing stepping stones. As presented above, a number of the existing methods can be adapted for the identification of the source of a spear phishing email. From a practical perspective, the greatest hindrance to the efficacy of traceback methods in reality is the difficulty in achieving global cooperation toward a common attribution goal. In addition, privacy concerns, security of the data used for attribution, the presence of intentional deception, as well the employment of zombies in a sophisticated attack are major practical problems that automated traceback needs to deal with. Automated source traceback in open networks such as the Internet is still a future prospect. Having said that, some techniques such as ingress network filtering can be, at present, implemented on some parts of the network (without many undesirable consequences), and thus can potentially assist the attribution problem to a certain extent. In general, it is unlikely that traceback methods alone can attribute a spear phishing email to its source; rather they should be combined with other techniques in order to infer more information about the attacker, or to narrow down his/her location.

11.2 Email authorship attribution

It is possible that a piece of email text pertaining to a spear phishing attack reflects the writing style, and possibly characteristics, of its author. This section therefore aims to present authorship attribution methods that are potentially applicable to spear phishing emails. Since email and online messages share many common features, the discussion of authorship attribution methods also includes, but is not limited to, those proposed for messages in blog posts, newsgroups, bulletin boards and chat rooms. In particular, these methods will be directly relevant in the case of Internet Relay Chat (IRC) and Short

Message Service (SMS) messages adopted as communication channels in a spear phishing attack. I also follow the convention in modern authorship attribution and hereafter interchangeably refer to a piece of text under attributional analysis as a *document*.

Whilst using text as a medium for representing and communicating information, attribution of mail messages is however not entirely the same as that of other types of writing work presented in Section 5. The reason for this is that email, as a modern form of communication, has introduced a new style of writing and a new type of language (*paralanguage*)[47], making itself an intermediate form between formal writing and informal speech. An email used for the purpose of formal and professional communication would contain many features similar to those in traditional text. However the majority of email messages in the wild would exist in their casual and informal form. As such, email messages do not often conform to rigidly syntactic and grammatical conventions. They also tend to contain simple-structured sentences, commonly used vocabularies, spelling and grammatical mistakes, and most importantly, email messages are usually short. These distinguished features render classical methods in authorship attribution less effective in the email context. For instance, the majority of the classical methods that use frequencies of word, sentences, various patterns and so forth as textual measures requires a text sample to contain at least 500 words [182] for the textual measures to be of statistical significance[48]. In reality, most email messages are far shorter than this threshold. Yet, a greater concern is a need to deal with higher-order impacts on attribution possibly caused by the combined effect of certain features of email messages. For example, the combination of adaptability and topic diversity may make email messages written by the same author vary greatly from one message to another.

Nonetheless, email text presents additional features that are not captured by the classical methods. Email text has structure (a typical email message consists of a greeting, a body and a signature) and peculiar features (e.g., the possible containment of requoted text in different positions, and the possible inclusion of disclaimers and advertisements). In particular, because of its informality and casualty, email text tends to naturally reflect an author's writing style, and the structure and layout of an email message is unlikely to be altered or modified by a (human or automatic) editor. Therefore some of the features of email, on the one hand, decrease the effectiveness of classical authorship attribution methods, whilst its additional features, on the other hand, leave room for the development and enhancement of methods to attribute email messages and their alike.

Authorship attribution methods presented in Section 5 primarily address the problem of *author identification* — the task of determining the most plausible author for an anonymous text from a group of potential authors, or suspects. They also work on a presumption that the anonymous text is of considerable length (thousands of words) and that there is a large number of sample documents for each known author. This section discusses authorship attribution methods that build on those discussed in Section 5 but can be applied to situations where these assumptions do not hold (e.g., attribution of email and online messages). Apart from author identification, these methods also address a wider range of tasks, including:

[47] Paralanguage involves the use of non-verbal elements to convey emotions (e.g., emoticons, bold/italic text, and extra symbols such as asterisks and underscores).

[48] Forsyth and Holmes [97] also found in their review that it was very difficult to attribute of a piece of text of less than 250 words to an author.

- *author characterisation*[49] which involves collecting sociolinguistic attributes, such as gender, age, occupation, and educational level, of the potential author of an anonymous document;
- *author similarity detection* which involves determining whether any two given documents are produced by the same author, without knowing who the author is; and
- *author verification* (closely related to authorship similarity) aims to answer the question of whether a given document is written by a specific author.

Attribution techniques for email analysis are discussed below. In the discussion, I treat authorship similarity detection and author verification as special cases of author identification.

11.2.1 Author identification

Pioneering efforts in determining email authorship are presented in [70, 4, 71]. More specifically, the initial experiment on author identification was conducted by de Vel [70] using a collection of 240 emails by 5 authors. De Vel used support vector machines (see Section 7.5.2) as a classification technique, and utilised a set of 38 features of various types — lexical, character-based, syntactic (e.g., function words), structural (e.g., number of tabs) and application-specific (e.g., the reply status). De Vel found that the features that effectively discriminated the different authors included the existence of tabs, number of lines in the email text and function word attributes. In subsequent work, Anderson *et al.* [4] and de Vel *et al.* [71] studied author identification across multiple email topic categories. For instance, de Vel *et al.* [71] used a total of 156 email documents by three authors discussing three independent topics (movies, food and travel). The researchers also used a larger set of features (170 features in total) containing up to 21 structural features, e.g., the containment of a greeting/farewell acknowledgment, containment of a signature text, number of attachments, positions of requoted text, HTML tag frequency distribution and total number of HTML tags. The experimental results demonstrated the combination of linguistic and structural features enabled the researchers to effectively discriminate the authors even when multiple topic categories are involved. However, according to de Vel and colleagues, since some of the features used in the experiments have a 'content-based bias' [71], the accuracy of the results would diminish if there was a content overlap between the topic classes. A more comprehensive investigation of email authorship analysis is presented by Corney in [65], the results of which both consolidated previous findings and provided important information. Corney's findings include: (i) though the use of linguistic features by themselves was not sufficient, the combination of linguistic and structural features, together with the use of support vector machines, was demonstrated to successfully attribute an email document to its author, (ii) due to the small number of words in a typical email message, the use of word collocations (word-based ngrams) was unsuccessful, and (iii) it was possible to determine authorship for email documents containing as few as 200 to 250 words. A full presentation of the feature sets used in Corney's work can be consulted in appendix A of [65].

[49] Author characterisation is also referred to as *author profiling* in the literature.

Motivated by the encouraging results achieved in the initial investigations of email authorship, a series of research studies have been undertaken in an effort to advance the state of the art of what is known as *modern authorship attribution*. The research studies addressed overlapping problems and achieved diverging results. To untangle the relationship among the studies as well as to present their contribution in a coherent manner, I will present them according to the major types of problems they attempted to address as follows: (i) studying of features that represent writing style in modern settings, (ii) investigation of custom feature sets that are either optimised for a domain or best represent the authors, (iii) study of author attribution in a multi-lingual context, (iv) development of methods for author verification/similarity detection, (v) devising solutions for a specific practical problem possibly encountered in modern authorship attribution (i.e., there is a very large number of authors with small samples of text), and (vi) evaluation of different techniques for attributional analysis.

11.2.1.1 Study of features that represent writing style in modern settings Zheng and colleagues [322] conducted a comprehensive study on writing-style features and classification techniques for online messages. They conducted experiments with English and Chinese 'for sale' online messages, in which content-specific features were used in conjunction with other content-free features (e.g., lexical, character-based and syntactic features). According to Zheng and colleagues, content-specific features are important discriminating features for online messages on the grounds that a specific user may often post online messages involving a relatively small range of topics whereas different users may distribute messages on different topics. For example, a criminal selling pirate software may use such words as *obo* (or best offer), for sale, whereas a criminal distributing pornography materials may frequently use words such as *sexy*. Therefore, choice of words or characters for similar concepts in specific topics may provide some clue about the identity of the author. Zheng and colleagues extracted a total of 270 features which consisted of 53 character-based features, 34 lexical features, 158 syntactic features (frequencies of punctuation and function words), 14 structural features, and 10 content-specific features (such as *deal*, *obo*, *sale*, *wtb*, *thx*, *paypal*, *check*, *windows*, *software*, *offer*, *Microsoft* which are specific in the 'for sale' context). Several important remarks were made on the experimental results: (i) lexical features themselves are not effective for authorship identification for Chinese online messages (this is probably due to Chinese having different word-segmentation to English), (ii) adding syntactic features to the feature set did not result in a significant accuracy improvement for the English dataset (since the online messages were too short and thus the frequencies of the function words are not of statistical significance), (iii) adding structural features improved the performance significantly for the English dataset (this suggested that structural features appeared to be a good discriminator among authors for online messages), and (iv) content-specific features improved the performance for the English datasets.

In an effort to improve the representative power of a feature set, Koppel and colleagues [163] proposed a new feature selection technique which is based on a *stability* criterion. Stability is defined by Koppel and colleagues as a measure that captures the extent to which a language element, such as a word or a syntactic construct, is replaceable by semantically equivalent elements (e.g., *smart* can be replaced by *clever*, *bright* and *intelligent* without introducing a significant change in its meaning). According to this definition, words such

as *or* and *the* are stable because they do not have semantically equivalent alternatives, whereas the word *be* is more unstable because it can be replaced by *being*, *been*, *was*, *were*, *is*, *are* and *am*, depending on the tense and the subject of the sentence. The idea underpinning the stability criterion is that linguistic features that are both *unstable* and *frequent* are most likely to reflect a stylistic choice, versus frequent but stable features such as proper nouns, or infrequent words [163]. In some aspects, the rationale behind unstable and frequent features is similar to that of function words; however, unstable and frequent features have an advantage over the use of function words in that the stability criterion can be applied to any type of linguistic feature — whether it is lexical, syntactic or complexity-based.

11.2.1.2 Investigation of custom feature sets All the research work presented above used a 'universal' fixed set of features constructed using features that promise to discriminate between the writing styles of different authors. This universal set of features is then instantiated to capture the writing style of authors. There is a movement toward the development of what I refer to as *custom* feature sets. The goal of a custom feature set is to either best represent documents in a specific domain (*domain-specific feature set*) or best represent an individual author (*author-specific writeprint*).

Author analysis using domain-specific writeprints Li and colleagues [186] devised a feature selection technique based on genetic algorithms (GA)[50] to generate an optimal set of features for online messages. To utilise the optimisation function of genetic algorithms, they represent a set of features as a set of binary values (e.g., '10011') in which 1 represents a selected feature and 0 represents a discarded one. Different subsets of the feature set are in the form of chromosomes, each of which has its length as the total number of candidate features, and the sum of all 1 digits as the actual number of features in the subset. The fitness value of each chromosome is defined as the accuracy of the prediction results. The selected features of the optimum chromosome at the end of the evolutionary process (the chromosome that achieved the highest accuracy of classification among all the generations) would correspond to the key writeprint features. To empirically validate their hypothesis, Li and colleagues conducted an experiment with a collection of online messages from `misc.forsale.computers` as the English dataset and online messages from the two most popular Chinese bulletin Board systems as the Chinese dataset. First the optimal feature set was constructed from the full set containing lexical, syntactic, structural and content-specific features, using the proposed GA-based feature selection algorithm. Examples of features selected in the optimal set included: (i) *@* and *$* for lexical features, (ii) *!* and *:* for punctuations, and, *if* and *can* for function words (both syntactic features), (iii) *number of sentences per paragraph* and *has separator* for structural feature, and (iv) frequencies of *check* and *sale* for content-specific features. The optimal set was then used for author identification using a support vector machine algorithms. The results demonstrated a higher accuracy result when using the optimal feature set for both the English and Chinese datasets, (99.01% and 93.56%, respectively) in comparison to those when using the full feature set (97.85% and 92.42%, respectively).

[50] Genetic algorithms refer to a class of probabilistic optimisation algorithms inspired by Darwin's theory about evolution. Genetic algorithms are suitable in finding solution for hard problems, especially where little is known about the underlying search space.

It was also noticed that greater improvement between the adoption of the optimal set and full set was achieved for the English dataset.

Author analysis using author-specific writeprints Iqba and colleagues [135] proposed a method to construct author-specific writeprints for the purpose of author verification (see below). The novelty of their approach is that the constructed writeprint can uniquely represent a specific author (in a closed set of authors). The idea underpinning this novelty involves representing the authors' writing styles as sets of *frequent patterns* extracted from his/her email messages, and eliminating any common patterns between the sets of frequent patterns. As the result of this process, each author-specific writeprint best represents the writing style of its authors, but does not represent the writing style of any other authors. To facilitate an efficient and easy construction of an author's writeprint, the researchers discretised the frequency of each stylistic feature into multiple intervals, called *feature items*. A feature item that contains a normalised feature frequency would have value 1, or have value 0 otherwise. In this way, each email document is represented by a feature vector of binary values. A *frequent pattern* is a feature item, or a set of feature items, that exists in a number of emails by an author which is greater than a user-defined *minimum support value*. A vector of such frequent patterns (a *writeprint*) is used to represent an author's writing style, and is made unique by filtering out the common frequent patterns. By matching write-prints with the malicious email, the true author can be identified. Iqba and colleagues reported that their writeprint method achieved a classification accuracy of 90% (with 0.1 as the minimum support value) when tested with 120 real-life emails by 6 authors.

11.2.1.3 Authorship attribution in a multi-lingual context Authorship attribution has also been studied across different languages (c.f., [241, 243, 186]). More specifically, Peng and colleagues [241] conducted experiments on Greek, English, and Chinese datasets. They studied the linguistic characteristics of each of the languages, and suggested the use of character-based features such as n-grams to avoid word-segmentation which is a difficult problem for many Asian languages including Chinese. Peng and colleagues built an authorship analysis model based entirely on n-grams, and used the model to identify authors of Greek newspaper articles, English documents, and Chinese novels using the Bayesian classification technique (see Section 7.5.1). They achieved high accuracy in each language, with the best accuracy achieved in each language being: English (98% using a 6-gram model), Chinese (94% using a 3-gram model) and Greek (90% using a 3-gram model). They also noted that the Chinese vocabulary is much larger than the English vocabulary, which may give rise to sparse data problems and necessitated the use of some feature reduction/transformation techniques (see Section 6.2.1). In the same vein, Li and colleagues [186] devised an optimal set of features, consisting of lexical, syntactic, structural and content-specific features, and applied it to authorship identification of both English and Chinese online messages. The researchers reported the accuracy of the results (produced by a support vector machine classifier) which were as high as 99.01% for the English dataset and 93.45% for the Chinese dataset.

11.2.1.4 Authorship verification/similarity detection Apart from the author-specific writeprint method presented above, the research work discussed so far has focused

on identifying the author for an anonymous document from a list of suspects, none of them have addressed the problem of verification and similarity detection. To address this problem, an intuitive solution is to divide the group of suspects into two groups: A or $\neg A$ (where A is the suspect who needs to be verified if he/she is the author of an anonymous document), and use an existing author identification method to determine whether the anonymous document belongs to A and $\neg A$. However, this approach has a fundamental flaw in the fact that other authors in the group (excluding the suspect) are treated as negative examples, and thus fails when the document is not written by any of the authors. This motivates Iqba and colleagues (see above) and other researchers [2, 164] to devise a new range of techniques that are able to determine whether a document is written, with a certain degree of certainty, by a suspect, rather than simply assigning the document to the most probable author in a closed set.

Research work by Abbasi and Chen [2] targets both author identification and similarity detection. They first devised an extensive base collection of features (tens of thousands), primarily consisting of different types of n-grams and including 5,513 common word misspellings — expecting that it would better capture all possible variations in the language used among the authors. Next, a writeprint was created for each author using the Karhunen-Leove transform[51] (or K-L transform) on the base collection of features and the author text. Then the dissimilarity in the writing styles of the suspect and the author of the anonymous document was measured through the usage of certain features, or *disrupted patterns* (e.g., there are features that the suspect never uses — the features each has a frequency of zero in the suspect's writeprint, but exist in the anonymous document, and vice versa. The comparison and similarity detection techniques used were based on K-L transform, instead of mainstream statistical and machine learning techniques. The researchers carried out an empirical comparison of their approach and existing benchmark methods based on support vector machine and principal component analysis (see Section 6.2.2), and across multiple domains including different types of online messages and programming code. The feature vectors used in the experiment were derived from non-overlapping blocks of text of approximately 250 words[52] from 100 authors. The result demonstrated that their technique significantly outperformed the other techniques in author identification for email messages and chat-room postings, and significantly outperformed the other technique for all datasets in similarity detection. Overall, the empirical experiment demonstrated the success of their approach for similarity detection with small sample of texts, and indicated the usefulness of author-specific feature sets when dealing with of large number of authors.

Abbasi and Chen used disrupted patterns (e.g., features with zero frequency in one document are used in the other documents) to determine the dissimilarity between documents belonging to different authors. Although this idea is intuitive and effective in many cases, it does not work if disrupted patterns are the results of the documents discussing different topics, or the author unintentionally/intentionally changing his/her writing style. To this end, Koppel and Schler [164, 166, 168] took a rather different approach. Instead of focusing on strongly discriminating features among documents across authors as important clues to authorship, the researchers identified features that reflect the dissimilarity

[51] Karhunen-Leove transform is an equivalence of PCA (see Section 6.2.2) in the pattern recognition paradigm (c.f., [303, 304]).

[52] 250-word is the minimum requirement for a piece of text to be effective in authorship analysis.

among documents by the *same* author and subsequently *eliminated* them. This approach is based on the hypothesis that no matter how different the two documents are, the differences between them will be captured in *only a relatively small number of features* if they are written by the same author, despite possible differences in theme, genre and the alikes. Once this small set of features is removed, it becomes significantly harder to distinguish between the two documents. Koppel and Schler described the operation of their method, named the *unmasking* technique, as follows. Based on an initial feature set containing words with the highest average frequency in the documents, the unmasking technique is iteratively applied to different portions of the two documents through two steps: (i) determine the classification accuracy results using a support vector machine with linear kernel (see Section 7.5.2), and (ii) eliminate the k most strongly weighted positive features and the k most strongly weighted negative features. With the masking technique, documents written by the same author become indistinguishable much faster (after a few iterations) than documents written by different authors. An experiment conducted on literary texts using the unmasking technique was reported to achieve consistently high classification accuracy (from 80% to 90%). It was also noted by Koppel and Schler that the unmasking technique is more effective at discriminating different authors (even if they are on the same topic) than distinguishing the same author on different topics [164]. The unmasking technique is also robust to a lack of data: it is capable of verifying that a document is authored by a suspect when writing examples of only that suspect are available to attribution practitioners (the *one-class classification problem*). In general, the effectiveness of the unmasking technique is appealing, however the author's sample text is required to be sufficiently long for the technique to be effective. As stated earlier, another major contribution of the unmasking technique is that by capturing an author's writing style at the 'submerged level of consciousness', the unmasking approach is able to address (to a certain degree) the problems of an evolving or deceiving writing style. Whilst it is possible for writing style to change for the purpose of deception, it is also apparent that human writing style evolves over time. There are many possible reasons for this. For instance, according to Pennebaker and Stone ([240] cited in [40]), people tend to 'use more positive and fewer negative affect words, use fewer self-references, use more future-tense and fewer past-tense verbs, and demonstrate a general pattern of increasing cognitive complexity' as they get older.

11.2.1.5 Authorship identification/verification for a large number of authors

The majority of research on authorship attribution deals with a small number of authors, with some researchers reporting a decrease in classification performance when attempting to increase the number of authors in their experiments. Zheng and colleagues [322] observed a 14% drop in accuracy when increasing the number of authors from 5 to 20, and Argamon and colleagues [8] experienced a 23% drop in message classification accuracy when increasing the number of authors from 5 to 20. Realising that scalability with respect to the number of authors is important for the practical viability of an authorship attribution method, Madigan and colleagues [195] and Luykx and Daelemans [190] studied authorship analysis on sets of 114 and 145 authors, respectively. Most recently, Koppel and colleagues [168, 169] performed authorship attribution in situations where there are thousands of known candidate authors (more specifically, 10,000 blog authors) with known documents of limited size (i.e., 2000 words in the known document for each

author). To handle a large number of authors (i.e., classes) — a situation in which the construction of a learning classifier would not be practical, the researchers made use of a simple statistical similarity-based measure *cosine similarity*, and represented each document as a large vector of space-free character 4-gram frequencies. Based on a presumption that two documents belonging to the same author would consistently exhibit similarity on various feature sets, matching each author's known document with the anonymous document (using the cosine similarity technique) is repeatedly carried out on a number of different subsets of the full feature set. The known document most often recorded as 'top match' will have its author assigned to the anonymous document. In general, the attribution method proposed by Koppel and colleagues achieved satisfying results. For a comprehensive evaluation of the method, please refer to [168]. An interesting result worth noticing from the experiment is that the fewer the number of authors, the less accurate the classification results are. *This result suggested that while authorship attribution involving a large number of authors with known documents of limited size can be solved, verification of authorship where both the number of authors and the size of known documents are small remains open for future research.*

11.2.1.6 Attributional analysis methods A variety of techniques have been applied to attributional analysis, including Naïve Bayes (c.f., [218, 160, 133, 242]), principle component analysis (PCA) (c.f., [39]), rule construction (c.f., [124, 126, 6, 165, 1, 322]), Winnow (c.f., [161, 7, 163]), Baysian regression (c.f., [107, 195, 5]), cluster analysis and discriminant analysis (c.f., [125, 182]), multilayer perceptron (c.f., [292]), radial basis function (RBF) networks (c.f., [189]), and especially support vector machines which are adopted by the majority of researchers in modern authorship attribution (c.f., [70, 71, 75, 165, 167, 1]). Please refer to Sections 6 and 7.5 for a description of the mentioned methods.

It is collectively revealed from individual empirical results that support vector machines, in general, provide the greatest accuracy results. This finding was also confirmed by a number of experiments conducting comparisons of different attributional analysis techniques on the same dataset. For example, Zheng and colleagues [323] reported in their published work that support vector machines achieved better performance than neural networks which, in turn, gained higher performance than the C4.5 decision tree algorithm in terms of precision, recall and accuracy; and Koppel and colleagues stressed that support vector machines and Bayesian regression are far superior to the other learning algorithms. However, when the number of classes (authors) involved is exceptionally large, simple statistical similarity-based methods such as the cosine similarity are considered more appropriate than machine-learning methods [168, 190].

11.2.2 Author characterisation

When identification of an individual author is infeasible (e.g., there are no known authors for a collection of emails), predicting characteristics, or traits, of authors can be a good alternative. By providing certain clues about an author's identity, author characteristics help narrow down a group of suspects, rather than pinpointing the actual author. Classification of author characteristics investigated to date include: gender and/or age (c.f., [7, 36, 37, 65, 93, 92, 162, 246, 264, 290]), education (c.f., [65, 93, 92, 246]), location (c.f.,

[246]), language background (c.f., [65, 167]) and psychometric traits (c.f., [93, 92, 244, 245]). For instance, Corney [65] built sociolinguistic profiles for authors which can be used to identify the gender, language background, age group and education level of email authors. Estival [93, 92] implemented TAT — an author profiling tool with application to Arabic emails demographic traits (age, gender and education), and psychometric traits (extraversion, lying, neuroticism and psychoticism). Thomson and colleagues [290] investigated the existence of gender-preferential language styles in email communication. The types of styles investigated included references to emotion, provision of personal information, use of intensive adverbs, the frequency of insults and opinions. The researchers hypothesised that the first three of these features are characteristics of female authors, whereas the last set of features are male-preferential. Using manual feature extraction and discriminant analysis (see Section 7.5.2), Thomson and colleagues claimed that they were able to predict the gender of email authors.

In particular, Koppel and colleagues [168] conducted a most comprehensive experiments on author characterisation which investigated all profiling dimensions presented above: gender, age and personality (more specifically, psychometric traits). For gender and age classification experiments, Koppel and colleagues conducted their experiment with a corpus containing a full set of postings of 19,320 blog authors (who provided self-reported age and gender); with each author document (which comprises all the postings by the author) containing from several hundreds to tens of thousands of words. The documents in the corpus were first classified into two categories of gender (male and female), and then grouped into three classes of age (13-17, 23-27 and 33-47). Important findings from the experiment are listed below.

- With respect to gender classification, content-specific features are slightly better than linguistic features.
- The linguistic features most useful for gender discrimination are *determiners* and *prepositions* (features of male writing) and *pronouns* (features of female writing).
- The content-specific features most useful for gender discrimination are words related to technology (male) and words related to personal life and relationships (female).
- Regarding age classification, the linguistic features most useful for discrimination are *contractions without apostrophes* (features of younger writing), and *determiners* and *prepositions* (features of older writing).
- The content-specific features that proved to be most useful for age classification are words related to school and mood for teens, work and social life for the 20s, and family life for the 30s.

Regarding document classification according to native language, Koppel and colleagues collected from the International Corpus of Learner English [105] 1290 documents (579-846 words long each) by 258 writers from Russia, the Czech Republic, Bulgaria, France, and Spain. Based on linguistic features that are specific to a particular language, the researchers classified documents into classes of authors with different native language. For example, authors with Russian, Bulgarian and Czech as their native language tend to omit the article *the*, and English words with analogs of common use in a native language,

such as *indeed* (French), *over* (Russian) and *however* (Bulgarian) tend to be used with higher frequency. Koppel and colleagues [168] observed that the classification accuracy significantly increased when linguistic features are used in conjunction with features that measure linguistic idiosyncracies and errors.

Toward classification of personality, Koppel and colleagues [168] made use of a corpus of essays written by psychology undergraduates at the University of Texas. The essays range in length from 251 to 1951 words and contain expressions of thoughts and feelings of the authors. Each author also provided a self-report for the five personality dimensions: *neuroticism*, *extraversion*, *openness*, *conscientiousness*, and *agreeableness*. The results demonstrated that while content words are, unexpectedly, useless for classifying documents by neuroticism, linguistic features give a great deal of information about personality. For instance, the experimental results suggested that neurotics tend to refer to themselves, to use pronouns for subjects rather than as objects in a clause, to use reflexive pronouns, and to consider explicitly who benefits from some action (e.g., through the use of prepositional phrases containing *for* and *in order to*); non-neurotics, on the other hand, tend to be less concrete, to use less precise specification of objects or events (e.g., through the use of determiners and adjectives such as *a* or *little*), and to show more concern with how things are or should be done (e.g., via the use of prepositions such as *by* or *with* and modals such as *ought to* or *should*).

11.2.3 Author clustering

To attribute a set of documents by unknown authors, instead of learning about the document authorship by inferring as many characteristics about the authors as possible (as discussed above), one can attempt to predict if the documents possibly have common origins (e.g., some documents are possibly authored by the same individual, or the same type of individuals) by grouping together closely similar documents. In general, author clustering is a straight-forward task which makes use of the instance-based representation (see Section 7.5.5) and clustering algorithms (see Section 7.5.6). When the dataset involved is large, one can capitalise on techniques for information storage and retrieval studied in text categorisation (see Section 6) to achieve an efficient implementation. It is also possible to merge multiple aliases belonging to an individual based on the documents authored by the aliases. For instance, Rao and Rohatgi [252] used principal component analysis and a nearest-neighbor-based clustering algorithm to merge 117 news posting users with two aliases; and Noval and colleagues [223] performed a similar alias merging task using their proposed similarity-based clustering method on the set of features (distribution of words/function words, misspellings, punctuations and emoticons) that represented the documents being clustered.

11.2.4 Discussion

The methods presented in this section are aimed at determining authorship for general emails and online messages, based on linguistic, structural and other features of the emails' textual content. Since a spear phishing email is distinguished from a general email *not* in its (textual) content, but in its intent, the discussed methods can be adapted for use in the context of spear phishing. Some remarks regarding this prospect are included below.

- With respect to stylistic features, n-grams and function words are among the most effective linguistic features for the discrimination of spear phishing emails by different authors. Since the content of spear phishing emails by the same author can vary significantly in different contexts, general word frequencies, if to be used, should be handled with caution due to their potential capturing of content words[53]. Structural features are less significant for spear phishing emails since many spear phishing emails tend to be presented in a formal manner. However they may be combined with linguistic features to improve the accuracy of the prediction results. Syntactic/semantic features (except for function words), on the other hand, are not very effective due to the relatively short length of spear phishing emails. In general, the combination of different types of features is likely to improve the overall accuracy of the authorship prediction results.

- Errors, as linguistic features, are less important in the context of spear phishing. While errors are among the discriminative features of phishing emails, it is unlikely to be the case for spear phishing emails. This is due to the fact that spear phishing emails tend to be carefully crafted (and thus unintentional errors are often avoided), and target a small number of victims (and thus intentional errors, e.g., those resulted from application of homograph techniques to evade detection (see Section 2.1), are less likely to exist). However, due to the very diverse backgrounds of spear phishers, unique terminology stills play a role in determining authorship of spear phishing emails.

- Content words should be included as features when the collection of spear phishing emails belong to a specific domain (e.g., the emails claim to be sent from a financial institution for the purpose of identify theft, or all the emails are part of attacks mounted on a specific organisation to exfiltrate sensitive data). In this case, information retrieval methods (such as Latent Semantic Analysis, see Section 6.2.2) can be applied to the collection of emails in order to obtain relevant content words.

- The feature set used for representing the writing style of the authors of spear phishing emails used can be static or dynamically tailored to the dataset under investigation. Since each type has its own merits and drawbacks, which type to employ depends on preference. A universal static feature set is likely to produce less accurate results when dealing with evolving datasets. Conversely, custom feature sets promise more accurate results, however they need to be regenerated with each change introduced in the dataset. So there is a trade-off between accuracy and performance overhead, which needs to be taken into consideration.

- Machine learning algorithms (especially, support vector machines) have been demonstrated to be effective and efficient for attributional analysis, and so they should be considered for classification techniques for spear phishing emails. However, when the number of spear phishing email authors is substantially large, attribution practitioners should consider utilising simple statistical similarity-based methods, instead of machine learning algorithms.

- In general, the challenges of authorship attribution in the context of spear phishing

[53] Features based on n-grams also face the risk of capturing content words but to a much lesser extent.

do not lie in the existing thematic methods themselves, but are rather rooted in a number of factors, some of which presented below.

  - An organisation has a collection of spear phishing emails, but with unknown authors. In this case, if the organisation has email samples of a particular suspect, authorship verification/similarity detection methods (such as the method using disrupted patterns [2] and the unmasking technique [164, 166, 168]) can be used to verify if the email document under investigation is written by the suspect. Otherwise, before author classification can proceed, it is necessary to label the collection, perhaps by performing unsupervised learning (such as clustering[54]) with the collection of emails, either based on (i) their textual content to group emails that might belong to an author or on (ii) metadata (possibly combined with some content) to identify classes of responsible authors (see Section 11.3).
  - A spear phisher may change his/her writing style on a regular basis to confound attribution. This is an intuitive and plausible expectation. However, identification of an offender solely based on the writing style reflected in an email is not likely to be considered admissible evidence in court, rather it tends to complement other types of evidence. Therefore, some spear phishers — especially those who reside in nation states where there is no or weak cyber legislation — have much less motivation to implement this practice. Having said that, current authorship attribution methods are not able to resolve the problem of deception[55]. The problem may be addressed to a certain extent using the unmasking technique proposed by Koppel and Schler [164, 166, 168]. Deception within text might also be addressed by attribution based on metadata drawn from the lure, hook or catch components[56].

- Regarding implementing authorship attribution in practice, email messages often need to be preprocessed to be in a suitable form for the attribution task. Except for the use of general natural processing language tools to extract tokens and phrases, specific methods and tools devoted to cleaning and extracting features of email messages can be adopted/adapted for use. To name a few, Carvalho and Cohen [47] proposed a technique to extract signature and reply lines from email; Tang and colleagues [287] devised a method for email data cleaning which involves non-text filtering and text normalisation; and Lampert and colleagues [178] implemented Zebra, a system based on support vector machines for segmenting the body text of email messages into nine zone types (*author*, *greeting*, *signoff*, *reply*, *forward*, *signature*, *advertising*, *disclaimer*, and *attachment*) based on graphic, orthographic and lexical cues.

[54] A number of clustering methods exist in the literature that can assist this task, a brief description of some of the methods are given in Section 7.5.6.

[55] Brennan and Greenstadt [34] demonstrated that current authorship attribution methods are quite susceptible to both *obfuscation attacks* (where the writing style is changed while the content is preserved) and *imitation attacks* (where the authors attempt to imitate the writing style of a specific author.

[56] Generally, deception within text is more easily achieved (e.g., copy-paste) than deception in metadata (e.g., deceptive malware metadata might require the use of different attack techniques and tools).

11.3 Adversary class identification and characterisation

Apart from providing ownership evidence such as IP/email addresses, *email metadata* can serve as class characteristic evidence, based on which a class of adversaries[57] that are likely to be responsible for a given spear phishing email can be identified. On the one hand, metadata can be associated with other pieces of evidence contained in a spear phishing email to reveal more information about the actual attacker. On the other hand, metadata of emails can be correlated in order to identify common adversary classes responsible for a collection of spear phishing email messages, as well as to determine the class associated with a specific email. Attribution methods in the latter category are similar to those described in Sections 5 and 11.2 except for (i) linguistic and structural features being replaced/complemented with metadata-based features, and (ii) identification of an individual author being replaced with the identification of a class of adversaries. In addition, email metadata plays a particularly important role when dealing with malicious emails containing text in images in order to evade detection and attribution where conventional attribution methods become ineffective.

Research work directed toward determining the origins of emails overwhelmingly uses text rather than metadata. However, one attribution method in this direction was proposed by Wei and colleagues [306] who conjectured that the relationships between the metadata-based features across a set of email messages can potentially reveal common origins of the emails (e.g., if different web domains referenced in the emails point to the same IP address, then they are related; likewise, if two IP addresses host the same web pages, then the two IP addresses are related). To verify the idea, the researchers attempted to cluster spam emails into groups of common origins, using a set of features either derived directly from the emails, or by looking up additional sources such as WHOIS data from a domain name registra. The set of features used in their clustering of spam emails largely consists of metadata-based features including `message_id`, `sender_IP_address`, `sender_email`, `subject`, `body_length`, `word_count`, `attachment_filename`, `attachment_MD5`, `attachment_size`, `body_URL` and `body_URL_domain`. These features have varying degrees of influence on the clustering task, for example, `sender's IP address` is considered weak evidence that does not prove two email are related, while `body_URL_domain` is suitable for clustering specific sub-groups of spam emails (i.e., spam emails that contain URLs), and a domain name (especially, a spam domain) is a very strong indication that the two emails are related. Wei and colleagues carried out their initial experiments on a dataset of 211,000 messages using hierarchical clustering algorithms. The emails that were predicted to have a common origin (i.e., belonging to the same cluster) were then verified through one or both of the two following methods: (i) the images associated with the emails' webpages were compared to determine if they are identical or nearly identical, and (ii) a 'WHOIS' and Host resolution was performed on each of the domains associated with the emails to find out if they resolved to the same IP address. Despite its promising prospect, the research is still in its preliminary phase and requires further study.

In a related approach, McCombie and colleagues [203] used email headers in an attempt to group phishing attacks into groups with common *modus operandi* or with the same level of criminal organisation. McCombie and colleagues discovered that IP addresses,

[57] A class of adversaries can also be referred to as a type of adversary, or an *abstract* adversary with certain characteristics.

though interesting, did not prove to be useful for classifying groups. Instead, less obvious features turned out to be more useful for this task. In particular, the researchers made use of the following features, in addition to those based on spelling, typographic errors and unusual terminology, to categorise the attackers: X-Mailer (e.g., Microsoft Outlook Express, Internet Mail Service and SquiirelMail), date[58] (as recorded in the email header), commonly used sender addresses (e.g., *Support@*, *Security@*, *Account@* and *access@*), commonly used word in subject lines (e.g., *Update*, *Access*, *Agreement* and *Security*), targeted victims and targeted financial institutions, and occurrence date (e.g., the phishing incidents seemed to occur at the midweek dates and peak on Tuesday). To enhance the accuracy of their classification method, McCombie and colleagues complemented email metadata-based features with features derived from the associated phishing websites (as presented in Section 13.3). On conducting experiments with the proposed method on a real-life dataset, the researchers discovered an important insight into the groups of adversaries behind the investigated phishing incidents: three out of the six identified groups of threat actors are responsible for up to 86% of the incidents.

Approaching the problem from a totally different perspective, Chen, Zhang and colleagues [54] [319] identified common origins of spam emails based on their visual features, or 'spam images'. Spam images are usually composed of foreground objects (text message and/or illustrations such as product pictures and logos), and background (e.g., color scheme and/or textures). The researchers pointed out that spam images from a common source often result from modifications to a common template, and thus it is possible to identify the origins of spam emails according to their visual similarity. Though applicable, attribution methods based on visual features tend to less useful for spear phishing emails. This is because visual features tend to be much more prevalent in spam emails (as a vehicle for advertisement) than in spear phishing emails.

Discussion Classification/clustering approaches to identify classes of adversaries associated with spear phishing emails can be built on existing methods (such as those presented above) by eliminating the irrelevant features and incorporating the relevant characteristics of spear phishing emails. Presented below are examples of metadata-based features to be (potentially) added in a spear phishing clustering scheme.

- **Deception method**: this feature indicates whether a spear phishing email is purported to install malware on a victim's machine or trick the victim into disclosing his/her personal credentials. If the former, the feature also details the malware delivery method (i.e., whether the malware is delivered via a malicious attachment or a drive-by-download), and the types of the malicious attachment (whether it is a Microsoft Office or Adobe Portable Document Format file). If the latter, the feature also indicates whether the victim is convinced to visit a fraudulent website or enter the information in an HTML form embedded in the email.
- **Email theme**: this feature captures the 'theme' of the spear phishing email; for example, whether the email claims to have come from a financial institution (associated with the victim) and asks the victim to verify his/her username and password,

[58]It is interesting to note that the X-Mailer field and the Date field time zone were observed only in phishing emails, and not in any valid email in the sample data [203].

or whether it appears to come from a senior person in the victim's organisation and asks him/her to read an important report attached to the email.

- **Social engineering technique**: this feature reveals techniques a spear phisher may use to 'lure' the victim into his/her trap. Attribution practitioners may choose to reuse/adapt a list of techniques identified by Drake [82] which include:

  - *impersonating reputable companies*: emails appear to come from a reputable company in order to gain the trust from their recipients,
  - *creating a plausible premise*: email senders claim that the recipients' account information is outdated or their credit card has expired,
  - *requires a quick response*: email senders threaten to deactivate or remove the recipients' account if they do not respond within a given period of time, and
  - *security promises*: email senders promise to keep the transaction secure, and the recipients' information confidential and private.

  Alternatively, the attribution practitioner can sort the techniques according to a more general categorisation such as the human-based social engineering techniques defined by Mitnick [267]:

  - *trapping of role*: social engineers exhibit characteristics of the role they have chosen to impersonate,
  - *credibility*: social engineers choose to act in certain way to gain credibility from their victims (e.g., warning the victims of security issues or offering help to them),
  - *altercasting*: social engineers place their victims in certain roles, and thus encourage them to perform some 'expected actions',
  - *distracting from systematic thinking*: social engineers put pressure on their victims to quickly arrive at conclusions to limit the amount of time for the victims to conduct systematic thinking,
  - *momentum of compliance*: social engineers train their victims to be compliant by putting them through a series of requests, starting with insisting the victims perform a small favour and ending with asking them to carry out the desired action,
  - *desire to help*: social engineers put themselves in positions that trigger the victims' instinct to help.
  - *attribution*: social engineers choose to act in a certain way to create false perception about who they are to their victims,
  - *liking*: social engineers influence their victim by making themselves 'likable' to the victims,
  - *fear*: social engineers use fear to compel their victims to perform an action, and
  - *reactant*: social engineers arrange situations in which their victims welcome certain requests that they normally criticise.

- **URL spoofing techniques**: this type of feature describes tricks applied to the URL contained in a malicious email. This feature can be assigned values based on the taxonomy of phishing emails [82] as presented below.

  - *Link text in email differs from link destination*: the link text seen in the email is usually different than the actual link destination (e.g., the following HTML code `<a class="m1" target="_blank" title="Update" href="http://www.memberupdating.com"> http://account.earthlink.com</a>` displays a URL as `http://account.earthlink.com`, but it is actually destined to send the user to `http://www.memberupdating.com`).
  - *Using `onMouseOver` to hide the link*: the JavaScript event handler `onMouseOver` is used by a number of threat actors to show a false URL in the status bar of the user email application (e.g., in a fraudulent PayPal email, the phisher coded the link in such a way that when a user puts the mouse over the link, the status bar will show `https://www.paypal.com/cgi-bin/webscr?cmd=_login-run` instead of its real destination `http://leasurelandscapes.com/snow/webscr.dll`).
  - *Using the IP address*: to conceal the hostname of phishing websites, phishers use the IP address (instead of the hostname) of the websites (e.g., the URL used in a fraudulent email message is `http://210.14.228.66/sr/`, instead of `www.memberupdating.com`).
  - *Using the @ symbol to confuse*: some phishers insert the symbol @ into URLs for the purpose of providing false and misleading information. The use of @ in an http link essentially renders all text before the symbol @ to be discounted (e.g., in the following example, the fraudulent Web site IP address `210.93.131.250/my/index.htm` is embedded in a URL that appears to be that of an eBay web page: `http://cgi1.ebay.com.aw-cgiebayISAPI.dll%00@210.93.131.250/my/index.htm`.
  - *Hiding the host information*[59]: taking advantage of the fact that some versions of Microsoft Internet Explorer do not display text after null or unprintable characters in a URL, some threat actors insert these characters in their URLs to conceal the actual hostnames. (e.g., the following link `http://cgi1.ebay.com.aw-cgiebayISAPI.dll%00@210.93.131.250/my/index.htm` will be displayed in the browser address bar as `http://cgi1.ebay.com.aw-cgiebayISAPI.dll`.
  - *Using hexadecimal character codes* fraudsters can also hide URLs by using dword/octal format, or hexadecimal character codes to represent the numbers in the IP address (e.g., the IP address `http://220.68.214.213` can be represented using hexadecimal character codes (each of which begins with '%') as `http://www.visa.com%00@%32%32%30%2E%36%38%2E%32%31%34%2E%32%31%33`.
  - *Redirecting the URL*: some companies provide a redirection service where the provided link takes the user to the service site and the service site then forwards the user to the intended site. Whereas redirection services are convenient for URL shortening, this service is abused by a number of phishers to confound attribution. Some phisher even employ a 'double-redirect' technique where the URL is redirected twice [82].

[59] This vulnerability has been fixed in a patch for Internet Explorer version 6 [82].

  - *Switching ports*: some phishers host their website on a compromised server (of a legitimate company) and hide the existence of their website by using a higher numbered port (e.g., the specific port 8034 is used in the IP address embedded the following link `http://www.citibankonline.com:ac-KTtF4BD6y4TZlcv6GT5D@-64.29.173.91:8034/`).

- **Victim**: a spear phishing email is distinguished from a phishing email in that it targets a few victims and tends to adopt more sophisticated technical/deception techniques. Therefore, in the case of corporate and espionage spear phishing attacks, *victimology* should be taken in account[60], and attackers can be grouped according to the types of victim they target. Features of this type can be specialised into *role* (individuals in certain roles may have access to sensitive data of interest to a spear phisher) and *relational* (relation between the sender and the victim) [3].

- **Locale**: the spear phisher's locale can be inferred through language settings, character encoding, time zone settings, Internet Protocol (IP) addresses and system host names [3]. For instance, Big5, GB2312 and GBK are all character encoding sets[61] used primarily in the far east region of the world, and +0800 and +0900 time zones cover China to Japan [3].

- **Tool**: spear phishers may prefer to use certain custom tools to create and send emails rather than standard tools (e.g., for the ease of spoofing email address or automatically tailoring the email, to a certain extent, to individuals). For instance, many email clients leave identification in sent emails via the `X-Mailer` header. A list of values for `X-Mailer` associated with spear phishing emails include `aspnet`, `blat`, `dreammail`, `extreme mail`, `foxmail`, `ghostmail` and `outlook express` [3].

11.4 Identity resolution

A *clustering-based* counterpart of authorship attribution (which is a classification problem), identity resolution has long been practised in criminal and terrorism-related investigations. Its goal is to resolve the problem of intentional identity deception by matching/merging references to an identity from different information sources (c.f., [146], [253], [297], [298], and [299]). Identity resolution has recently found applications in cyberspace, especially in the email domain, to merge different references belonging to a cyberspace identity (or *pseudonym*). For instance, efforts to resolve identities in cyberspace include: extraction of contact information (from the Web) for those whose names and email addresses are found in email headers [67]; categorisation of different email addresses corresponding to the same identity [129]; recognition of names referenced to the same identity in an email collection [210]; revelation of relationships between people based on their email communications [199, 200]; analysis/ranking of identities in a collection of emails according to level of expertise on discussion topics [79], and the design and implementation of a

[60] The degree of sophistication of the attacker is mostly revealed through the hook and catch components (as presented in the subsequent sections).

[61] Character encoding is concerned with mapping a character to a form that is suitable for transmission and display on a computing system. Character encoding is usually set by the email client or tools used to create the email.

computational model of identity that can be used for resolving mentions (i.e., references such as names, words, and phrases) in email [90].

Identity resolution plays an important role in identification and apprehension of criminals. In the context of spear phishing attacks, the efficacy of identity resolution approaches diminishes due to the combination of the following factors: (i) methods devised for cyberspace identity resolution are usually applied in situations where there is a large collection of emails (and/or other sources of information) containing many references about an individual, which is unlikely to be the case for spear phishing emails, and (ii) the methods also require the existence of at least one stable attribute that is unique to an identity (i.e., names and email addresses) to serve as a 'pivot' (or 'seed') for merging all other references about the identity, which is also unlikely to be the case for spear phishing emails where names and email addresses are the most likely spoofed items. However, these pitfalls do not entirely invalidate the potential application of identity resolution techniques in spear phishing attribution. If one is able to get hold of a mailbox of a suspect, to state an example, identity resolution techniques and the alike can be applied to the suspect's mailbox to reconstruct his/her social networks. The results of this might provide useful clues as to whether the suspect is connected to the spear phishing attack under investigation. To illustrate another example, if a piece of evidence that is unique and strongly connected to the threat actor is obtained (e.g., an author-specific watermark is found in an attached malicious code), techniques in identity resolution can be applied that use that piece of evidence to merge all the references belonging to the actor.

12 Attribution of evidence in the hook component — malicious software

The hook component of a spear phishing attack is purported to collect sensitive information from those who fall victims to the attack. In deception-based spear phishing, the hook refers to the illegitimate website referenced in a spear phishing email, while in malware-based attacks, the hook is the malicious code either embedded in an email attachment or delivered via a drive-by-download. Obviously, being able to identify or obtain any information about the developers of the hook component is very important for the overall attribution task. Techniques of authorship analysis potentially applicable to attribution of these two modalities are presented in this section and the next section.

The most direct and efficient way to determine the author of a piece of malicious code is via detection and retrieval of (i) explicit authorship information[62] or (ii) a watermark, if any, embedded in the code. Watermarking is a technique aimed at hiding authorship information in a piece of software. To legitimate software, watermarking is intended to protect the copyright and authorship, and to defend against illegal use, of the software. A watermark is also possibly embedded in a piece of malicious code, mostly out of fame and personal satisfaction. Therefore watermarks can serve as ownership evidence — if a watermark is found, a direct link between the code and its author can be established, otherwise techniques based on other features of the code need to be conducted. This section is devoted to a discussion of the latter group of techniques[63].

Given an anonymous piece of code and the task to identify its author, perhaps the most intuitive approach one may take is to compare the piece of code with those by the suspected authors. In the case where no information about the suspects is available, scrutinising the code in order to infer as much information about the author as possible seems the most likely alternative option. In fact, this is *exactly* how the authorship problem is approached by the researchers in the community of *software forensics*. A relatively young research area, software forensics involves analysis of software code for evidence of activity, function, intention and authorship [268]. Of particular interest to this section is the portion of software forensics devoted to the study of the link between a software program and its author(s) (henceforth, referred to as *software authorship*). There are many reasons for this connection to be investigated, ranging from resolution of intellectual property disputes, defense of cyberspace attacks to tracking of source code for project management. Within the software authorship discipline, research has specialised in multiple sub-areas, namely, *author identification*, *author characterisation*, *author discrimination* and *author intent determination* [112]. The goals of author identification and author characterisation in the context of software are almost identical to those studied in natural language (see Sections 5 and 11.2). The goal of author discrimination is to determine whether one or more authors are involved in the creation of the program (c.f., [188], [35]), and that of author intent determination is to clarify whether the destructive code is deliberate or accidental, i.e., whether the code is a result of a malicious intent or a software fault (c.f., [272], [274]). Research directly relevant to attribution of malicious code focused on author identification

[62] Some viruses contain names, addresses, company names, email addresses and Websites either in plain text or in an encrypted form [268].

[63] This group of techniques can also capture watermarks, if any, exist in pieces of code.

— the topic to which the majority of the section is devoted. Characterisation of the likely author of a piece of software code is also relevant, but very little effort has been made on the topic. This is partly due to the fact that author characterisation is considered a topic pertaining to psychology in the software forensic community [173]. The other two areas of software authorship —author discrimination and author intent determination — are not directly relevant to the attribution task, and thus are not discussed further in this section. A closely related topic to the attribution problem under discussion is source code plagiarism. However, according to Krsul, traditional methods for plagiarism detection are not suitable for authorship analysis: metrics calculated for two functions that are both structurally and logically equivalent but stylistically different would be similar in plagiarism, but dissimilar in authorship analysis. For the reader's interest, information about plagiarism detection can be consulted at [220, 221, 32, 149, 117, 86, 282, 63, 119, 295].

The following two sections discuss methods to identify single adversaries (based on the programming style reflected in a piece of malware) and predict classes of adversaries (based on the tools, techniques and behaviours associated with a piece of malware).

12.1 Malware authorship attribution

As a piece of software code, or a *program*, is presented in a textual form and specified by a language, researchers in the field have capitalised on the experience and methods in natural language authorship attribution. Here, a piece of code is in place of a piece of text, a programming language in place of a natural language, and *programming features* replace/complement linguistic and structural features. However, unlike authorship attribution for natural languages which have been studied for almost two centuries, research on attribution of source code have received attention for only the last two decades[64].

Another difference between the two types of authorship attribution is the degree of hardness associated with the attribution problem of software code. Having many similarities with natural language authorship attribution, identifying the likely author of a piece of software, however, faces more difficulties. This is due to the lower degree of flexibility of programming styles in comparison to that of writing styles in natural languages. The flexibility constraints of programming styles come from the following sources:

- software compilers mandate a piece of code to follow very rigid programming rules for it to be successfully compiled and executed,
- it is highly expected that the code follow guidelines and conventions for the purpose of efficiency and readability,
- some development tools automatically generate code templates and prototypes, or automatically format the code for the programmers' convenience, and
- a portion of the code, due to libraries and with the advent of object-oriented languages, may be reused from other sources.

[64]This is probably due to the fact that software authorship is built on the corresponding field in natural language which is already mature, and requires researchers to have sufficient knowledge about programming.

In addition, there is a significant difference in the vocabulary size in natural language and the number of constructs in code: whereas the number of English words is around one million[65], there are, for example, about thirty-two keywords, thirty-nine operators, and fifteen standard header files available in the C programming language [40]. Furthermore as time passes, programmers vary their programming habits and their choice of programming languages [173]. To make the attribution task even more challenging, the remnant of a cyberspace attack is often only a piece of executable code [112]. Source code can be recovered from executable code, but with information loss. This is because many programming features useful for attribution are stripped away or modified in the executable code (for example, comments and indentations are removed, identifiers are replaced by memory addresses or register names, and optimisations may be performed on the code), possibly giving the executable code a very different structure than the original source. However, according to Spafford and Weber [273], and Gray [112], important information for attribution still remains: data structure and algorithms, compiler and system information, level of programming skill and area of knowledge, use of system and library calls, errors present in the code, and symbol tables. Despite the presented obstacles, software forensic researchers conjecture that programming, especially in a language rich in data types and control structures, still has considerable room for variation, innovation and personalisation [273]. Also, humans are perceived as creatures of habit [175], who often opt for, either consciously or subconsciously, following certain habitual patterns when engaging in activities. With this prospect, researchers in the community have strived to identify sets of features that may remain constant for a significant portion of the programs that a programmer might produce.

A rich source of programming features is *software metrics* which are organised by Burrows [40] into eight categories: *size metrics* (e.g., lines of code, token counts, and function counts), *data structure metrics* (e.g., amount of input/output, variable spans and data sharing between modules), *logic structure metrics* (e.g., a number of paths and nesting levels of a looping construct, the use of goto statements), *composite metrics*, *software science composite metrics*, *effort and cost metrics*, *defect and reliability metrics* and *design metrics*. Among the vast number of metrics that can be extracted from source code, the metrics that may assist in authorship analysis are either selected or extracted with an expectation that a given set of programs from one author would be more similar in terms of these selected/extracted features than a set of programs from a variety of authors. Once a set of features is compiled, the procedure to determine program authorship is similar to that of authorship attribution of natural language texts (see Section 5).

12.1.1 Review of software authorship methods

Author identification involves matching the authors via the programming style reflected in the software code. A number of research efforts have been attempted to identify program authorship, representatives with reported empirical results are given below.

12.1.1.1 Author analysis of Pascal programs Oman [226] conducted their experiments on authorship analysis using eighteen short Pascal programs by six authors. Each

[65]Data is obtained from `http://hypertextbook.com/facts/2001/JohnnyLing.shtml`.

program implements a simple algorithm based on textbook example code for three tree traversal and one sorting algorithms. To describe programming style, Cook and Oman used a set of fifteen features based purely on typographic characteristics such as `inline comments on the same line as source code`, `keywords followed by comments` and `one or two space indentation occurred more frequently`. Cook and Oman then transformed each program into a vector containing Boolean values for these features and utilised clustering for authorship analysis. Though accurate results were reported, the adequacy of their approach was strongly criticised by Krsul [173] on the following points: (i) the implementation of the programs, by their nature, tend to be similar, (ii) the features used are specific to Pascal programming language, and most critically (iii) a Boolean measurement is not suitable for many programming style features.

12.1.1.2 Authorship analysis of C programs Krsul [173, 174, 175] investigated the problem of program authorship by looking at stylistic characteristics of C source code. To improve on the simple and inexpensive set of features used in the previous attribution work on Pascal programs, Krsul devised a comprehensive and expressive collection of fifty features which are categorised into three types: *programming layout metrics*, *programming style metrics*, and *programming structure metrics*. Programming layout metrics measure layout features such as indentation, placement of comments and placement of brackets. These metrics was considered as fragile because they are easily altered by tools with automatic formatting capabilities. Programming style metrics measure characteristics that are less susceptible for modification by formatting tools, such as means variable length and mean comment length. Programming structure metrics measure those that are hypothesised to reflect the programming experience and ability of the programmer. Krsul selected a small but most useful subset from the collection (via an empirical analysis) to take part in a programming authorship experiment which involved 29 authors with various programming experience and expertise, and 88 programs from a wide variety of programming styles and for different problem domains. Krsul used a discriminant analysis method (see Section 7.5.2) and achieved a success rate of 73%. It was concluded from the experiment that though it is possible to find a set of metrics that can be used to classify programmers correctly, it is not possible for a small and fixed set of metrics to be able to classify all possible programmers successfully. The final speculation from this project is that it is feasible to identify a particular programmer of a piece of source code, for a specific set of programmers within a closed environment. Krsul suggested observations about the suspect's environment (e.g., the preference of revision control system, integrated development environments and the educational background of a programmer) could be used to infer whether or not the suspect is the author of the anonymous program as well as the exploration of other statistical methods for authorship analysis. This speculation has been validated by subsequent work by other researchers in the community.

12.1.1.3 Authorship analysis of C++ programs Different from Krsul's work which focused on identifying a set a features to allow one to distinguish between authors, MacDonell and colleagues [192, 193] aimed at examining the efficiency of different classification techniques in software authorship analysis. MacDonell and colleagues used a dictionary-based system called IDENTIFIED (Integrated Dictionary-based Extraction of Non-language-dependent Token Information for Forensic Identification, Examination,

and Discrimination) to extract source code metrics for authorship analysis. In total, the researchers extracted a set of 26 standard authorship metrics from 351 programs by seven authors. The programs are either based on programming books or examples provided by a C++ compiler, or written by experienced commercial programmers. Three different classification techniques were used in the experiment: neural network, multi discriminant analysis and case-based reasoning. The results demonstrated a high overall classification rate, with case-based reasoning being the best performing technique (achieving a classification rate of 88.0%), which is followed by neural network and multiple discriminant analysis (achieving an identical classification rate of 81.1%).

12.1.1.4 Authorship analysis of Java programs Claiming that little had been done about authorship identification of Java source code, Ding and Samadzadeh [76] studied software metrics from Java source code for authorship identification. The set of extracted metrics for Java program authorship analysis were adapted from metrics developed for authorship analysis of Pascal, C, C++ programs as presented above. The researchers extracted 56 metrics (16 programming layout metrics, 13 programming style metrics and 27 programming structure metrics) from 46 groups of Java programs — each group containing 4 to 10 programs — collected from Internet shareware, computer science classes, and fellow graduate students. The researchers then used *stepwise discriminant analysis* to remove redundant metrics (8 metrics were removed in this step), and *canonical discriminant analysis*[66] as a classification technique. Ding and Samadzadeh reported that 62.6 – 67.2% of Java source files could be assigned correctly to their 46 original authors. In the case of derived canonical variates (the linear combinations of metrics derived from the metrics available in canonical discriminant analysis), the percentage were much higher (up to 85.5%). The result also revealed that layout metrics played a more important role in authorship identification than the other metrics[67].

12.1.1.5 Authorship analysis of Java and C++ programs using n-grams An obvious limitation of the above approaches is that the programming metrics are language-dependent. Also, the process of metric selection is not trivial, which involved either manual or automated (using statistical methods) filtering out uncontributive metrics in the initial set. Frantzekous [98, 100, 101, 102, 103, 99] attempted to advance the current state of program authorship analysis by devising a method that does not bear these drawbacks. The novelty of their approach involves the use of n-grams, a feature in natural language rather software metrics. As shown in Sections 5.2.1 and 11.2.1, n-grams were demonstrated to be very successful in identifying the authors of written work, and do not require multiple training samples for each author because n-grams are directly generated for an author profile. However, the commonly used similarity measure to compare n-gram profiles is not suitable for short pieces of source code, so the researchers applied n-gram methods for software authorship analysis with a simplified version of the similarity measure. The n-gram method was shown to achieve impressive accuracy results: the classification accuracy reached 100% in most cases when tested on the test set used in authorship analysis of C++

[66] See Section 7.5.2 for a brief explanation of discriminant analysis.

[67] Please note that this result is not necessarily contradictory to Krsul's view about layout metrics. Layout metrics are only fragile with the presence of an automatic formatter. In the absence of such tools, layout metrics can be expected to reflect the programming style of a programmer.

program by MacDonnel [192, 193]. To construct a more realistic test case, the researchers used open source code by eight authors each of whom developed from 4 to 30 programs, and the size of each program ranged from 23 to 760 lines of code. The n-gram method was shown to maintain high accuracy results for the realistic test set, even when the number of authors increased to 30, the accuracy still exceeded 95% in most cases. The experiments also indicated that the best classification was achieved for n-grams of size 6 or 7 and for profiles of size 1500 to 2000 (n-grams).

12.1.1.6 Authorship analysis using histograms Histograms were used to present values for programming metrics in [170] and [179], but these two research efforts are very different. Kothari and colleagues [170] attempted to construct a distinct set of programming metrics for each author (a kind of programmer-specific *code-print* in reference to author-specific writeprint in natural language). To construct a profile, the researchers computed for each programmer a histogram for the distribution of each style-based metric (e.g., distribution of line size, leading spaces and semicolons), and a histogram for frequencies for all possible 4-grams extracted from the author's source code. To make the profile of each programmer author-specific, the researchers made use of Shannon information entropy (see Section 6.2.1) to compute for each metric two entropy values: *individual consistency entropy* and *population consistency entropy*. Conceptually, the individual consistency entropy allows one to measure the consistency of a specific metric among pieces of source code by a single developer, whereas the population consistency entropy measures the consistency of a specific metric among pieces of source code across different developers. The researchers selected, for each author, 50 top metrics with high individual consistency entropy values and low population consistency entropy value, to construct a profile for the author. The resulting author profile, in effect, contains metrics that represent the author's programming style, and at the same time, sufficiently distinguishes itself from the rest of the authors. A case study was conducted on groups of 8 and 12 developers with a number of source code files, 1287 and 108 respectively. The results revealed that 4-gram metrics significantly outperformed style-based metrics (accuracy achieved for 4-grams using the Bayesian algorithm was from 61% to 95%, in comparison to 18% and 63% achieved by style-based metrics). Also making use of histograms, Lange and Mancodiris [179] aimed to construct a 'universal' effective set of programming metrics for discrimination among the authors using a genetic algorithm rather than Shannon information entropy. The researchers devised a set of programming metrics based entirely on programming style (no n-grams included). The genetic algorithm was applied to this set to select a good metric combination for author identification. They used this method to classify 40 projects among 20 developers and achieved accuracy results ranging from 53% to 75%.

12.1.1.7 Software authorship using writeprints Recall that Abbasi and Chen [2] proposed the Writeprints technique that incorporated a rich set of stylistic features for the purpose of author identification and similarity detection. Their technique is applicable to a range of domains including email, instant messaging, feedback comments and program code, and was empirically demonstrated to outperform a number of existing thematic methods [2]. However, the researchers suggested that the Writeprints technique should be extended to incorporate program-specific features (such as those studied in [174, 226]) in order to enhance its performance. For more information about the discussed research

work, the reader is advised to refer to Section 11.2.1.

The representative work in software authorship presented above reveals one interesting fact: the most successful method is the one that makes use of a single linguistic feature (i.e., n-grams) without the use of any programming metrics. This fact was also confirmed in the research work by Burrows [40] that follows.

12.1.1.8 Software authorship analysis by Burrows Burrows [40] evaluated the authorship analysis methods presented in the previous research work by reimplementing them using the same or equivalent algorithms provided by the Weka machine learning toolkit and executing the algorithms using the same set of metrics on the same dataset. Though the achieved accuracy results for all the methods were much lower than those reported in the original papers, the authorship analysis method based on n-grams was still demonstrated as the most effective. Burrows conjectured that the accuracy result of n-gram methods may even be improved if additional linguistic features are incorporated. As such, Burrows replaced character-base n-grams with token-level n-grams where a token can be a lexical or character-based feature. Burrows found from his experiment that the types of token captured in token-level n-grams that provided the best classification result and significantly outperformed the character-based n-gram method, are *white spaces*, *operators* and *key words*. In addition, Burrows also conducted a comprehensive study of software authorship problems with variations in the number of authors, number of samples per author, sample size, author stylistic strength, and sample timestamp (to deal with programming style changes). Detailed empirical results reported in the study should be consulted at [40].

12.2 Adversary class identification and characterisation

Apart from evidence directly relevant to authorship of a piece of malicious code (i.e., answering the *Who* question), information pertaining to the *What*, *How*, *When* and *Why* questions obviously provides a valuable insight into the characteristics of the class of attackers that are likely to be involved in the commission of an attack. For example, an analysis of a malicious piece of code can report that the malicious code is a type of keylogger/screencapture that aims to capture personal credentials, or a piece of custom malware designed specifically to exfiltrate certain sensitive documents. While the former case represents a general case of a spear phishing attack where a spear phisher is likely to commit a financial fraud, the latter case seems more specific to an espionage scheme, either industrial or political, where a spear phisher is likely seeking the intellectual properties of an organisation or classified data relating to national security. This demonstrates the possible contribution of malware analysis and characterisation to the attribution task of determining the class of adversaries likely responsible for a spear phishing attack. Software/malware analysis is in the realm of software engineers and security experts, rather than attribution practitioners. As such, this survey is not intended to describe techniques to perform software/malware analysis — there are, in fact, a wide range of tools publicly and commercially available to assist this task — it is instead focused on presenting ideas and methods to *use* the malware analysis results for the purpose of attribution. In particular, methods to determine a class of adversaries associated with a piece of malware is

presented in the following two perspectives: (i) prediction of certain characteristics (e.g., capabilities, connections and professionalism) of the adversary class based on the characteristics of a piece of malware, and (ii) identification of the intents as well as common sources of a collection of malware via malware classification.

12.2.1 Malware-based adversary characterisation

The adversary class discussed herein can be understood either as the class of individuals likely authoring the malware sample, or a class of attackers likely committing the malware-based attack. The classes defined in these two points of view may be or may not be referenced to the same group of individuals.

Regarding the first point of view, classes of adversaries can be revealed through a classification of malware samples based on metadata, or characterised based on the programming styles reflected in the malware samples. In the former case, metadata that is useful in discovering classes of adversaries include: development tools (such as compilers, text editors and databases), programming languages, and cultural influences in programming [268]. An example of cultural influence is the difference between program designs in the Windows environment (Windows programs are usually large and monolithic) and those in the UNIX environment (Unix programs tend to contain small individual modules making calls to a number of single-function utilities) [268]. In the latter case, some background information about the class of authors associated with a piece of malware can be determined by examining the programming style. For example, Java programmers tend to use more descriptive or verbose variable and method names, and 'undergraduate students and electrical engineers are notorious for abusing hashing' [173], 'Lisp programmers are notorious for abusing linked-list data structures' [173], and 'native Fortran programmers prefer using short lines' [173]. However, as mentioned in Section 12.1, there are very few, if any, efforts on deriving the characteristics of the author of a piece of program code. Apart from the general guidelines recommended by Gray [112] displayed in Table 4, it appears there is no attempt to pursue this research direction any further.

Table 4: *Software metrics and their clues to author characterisation.*

Software metrics	**Provided clue to author characterisation**
Number of each type of data structure	Background and sophistication of a program author
Cyclomatic complexity of the control flow of the program	The manner in which the code was written, the characteristic style of a programmer
Quantity and quality of comments	Linguistic characteristics
Types of variable names	Background and personality of the author
Layout (indentation and modules)	Background and the author s personality

Regarding the second point of view, perhaps the only relevant and comprehensive work in the literature is the study by Parker and colleagues [231] who attempted to characterise cyber adversaries based on certain characteristics of the automated attack tools utilised by the adversaries[68]. Since malware accounts for a subset of such tools, ideas and findings contributed by the researchers can be potentially used, or built on, to characterise the adversary class in the spear phishing context. A relevant portion of the research work is presented below — throughout the following discussion, the reader may choose to interpret an attack tool as a piece of malware, and attack techniques as the techniques employed by the malware.

12.2.1.1 Characterisation of cyber adversaries by Parker and colleagues To characterise cyber adversaries, Parker and colleagues [231] defined a collection of metrics which can, once assigned with real values, suggest certain characteristics of the adversaries. With respect to an attack itself, the metrics are grouped into two categories: *attack tools* (the tools used during a single attack) and the *attack techniques* (the techniques used during a single attack).

Attack tools Knowledge of the attack tools used in an attack (or the malware samples in the context of spear phishing) potentially reveals various information about the attacker — not only does it clearly indicate the tools the attacker has at his/her disposal (since there exist a large number of attack tools, some of which are available in the public domain, many of which are not), but it also provides information surrounding the ways the tools are used. The researchers considered the set of following metrics for attack tools:

- *The ease with which the attack tool is used*: this metric indicates the skills required to use an attack tool in order for the attack to be successful. Naturally, available exploits require the users to possess a varying level of technical understanding of the vulnerabilities that the exploits are leveraging for them to be used effectively. Therefore knowledge of the attack tools enables attribution practitioners to determine not only the types of skill an attacker possess, but also the levels of said skills.
- *The availability of the attack tool*: this metric informs whether or not an attack tool is available in the public domain, and subsequently reveals information relating to various aspects of the attacker's professionalism such as finance, resource and connection.
- *Nontechnical skill prerequisites*: this metric refers to any nontechnical-related prerequisites an adversary must possess to be able to use a specific tool (e.g., local area network access or physical access)

Figure 5 summarises an example set of attack tools together with their associated metric values and information about the unknown adversaries presented in [231].

[68] The research work by Parker and colleagues will be discussed further in Section 18.2.

Characterising cyber adversaries based on attack tools alone is, in many cases, insufficient. It is not always possible to determine the utilised attack tools (as in the case of general network attacks), and if the attack tools are able to be determined (as in the case of spear-phishing attacks), the same tool (or malware) can be implemented using different techniques, reflecting varying levels of skill and sophistication of the respective attackers. For this reason, analysis of the attack tools should be conducted in conjunction with examination of the attack techniques in order to obtain more complete knowledge about the adversary.

Attack techniques Attack techniques refer to the specific technologies and attack methodologies leveraged to perform a given attack. The set of following metrics considered by the researchers in assessing an attack technique is presented below.

- *Nontechnological resources required*: this metric presents the nontechnological resources that are required to leverage an attack technique, e.g., time, finance and the level of initial access.
- *The distribution level of the attack technique*: this metric indicates how well known the attack technique is, the knowledge of which potentially reveals a great deal of information about the capability and professionalism of an attacker.
  - One approach to identify the technique's distribution level is to determine the distribution status of the technique in the *vulnerability disclosure procedure* via the use of the *vulnerability disclosure pyramid metric* (see Figure 6) or *disclosure food chain* (see Figure 7). To compromise a target host, an attacker generally needs to acquire knowledge of a specific vulnerability affecting the host, and (if possible) possess an exploit code for the vulnerability. This turns out to be not a significantly challenge nowadays where there are hundreds of new vulnerabilities discovered and published every month, often coupled with exploit programs [231]. Once the information is released to the public domain, knowledge of software vulnerabilities and the accompanying exploit programs become accessible to everyone, including adversaries. Therefore although vulnerability information is published with good intention, it is often used for adversarial purposes. In response to the security concerns regarding the damage that could occur through vulnerability disclosure, various models of vulnerability disclosure procedure have been proposed and implemented in practice [231]:
    * *full disclosure*: all information pertaining to a vulnerability should be publicly available to all parties,
    * *responsible restricted 'need to know' disclosure*: only partial information is released, while giving an opportunity for software/ service vendors to fix the problems before a disclosure is made,
    * *responsible full disclosure*: all information is provided to security communities, and the respective problems are provided to the relevant software or service vendors/providers,
    * *security firm 'value added' disclosure model*: the disclosure of a vulnerability is made in the form of an alert or full advisory, as part of a 'value

Table 5: *An illustration of an example set of attack tools together with their associated metric values and information about the associated adversaries. All of the attack tools presented above do not require nontechnical skill prerequisites and this metric is omitted from the table.*

Attack tools	**Skill level required**	**Availability**	**Adversary profile**
Mass rooters	Typical skill level required is generally low	Mostly available within the public domain	Those who tend to compromise any vulnerable hosts on public networks happening to fall into their path
Port-scanning tools	Typical skill level required to execute the tools is extremely low but in-depth knowledge of network technologies is required in order to attain the desired goal	Mostly available within the public domain	Infeasible to determine without considering the attack techniques
Operating system enumeration tools	Typical skill level required to execute the tools is low but higher skill levels are required in order to attain the desired goal	Mostly available within the public domain	Infeasible to determine without considering the attack techniques
Software exploits	Skill level is reflected through the way the adversary uses a software exploit (e.g., does the exploit require detailed knowledge of the respective vulnerability or does the exploit require any modification for it to function?)	Mostly available within the public domain	Infeasible to determine due to the broad nature of software exploits
Commercial attack tools	Typical skills required to execute the tools is low	Far less available on the Internet — less accessible to general adversaries (due to their high cost) and only available to adversaries sufficiently well connected to the underground economy	Those who are likely to compromise the organisation's system, sometimes using tools stolen from the organisation (since commercial tools are often watermarked to the purchaser).

Table 6: *An illustration of different techniques to leverage flaws in web application, Linux 2.4 user space software, and Linux 2.4 kernel software together with the required skill levels which are scored according to the difficulty degrees associated with the respective attack techniques [231].*

Types of exploit	**Attack techniques**	**Public**	**Private**
Web application exploitation techniques	Proprietary application penetration: *SQL Injection*	3	5
	Open source application penetration: *SQL Injection*	3	5
	Proprietary application penetration: *Arbitrary Code Injection*	2	4
	Open source application penetration: *Arbitrary Code Injection*	2	4
	Proprietary application penetration: *OS command execution using SQL Injection (MS SQL)*	3	5
	Proprietary application penetration: *OS command execution using SQL Injection (Sybase)*	3	5
	Proprietary application penetration: *SQL Injection only (MS SQL)*	4	6
	Proprietary application penetration: *SQL Injection only (IBM DB2)*	6	8
	Proprietary application penetration: *SQL Injection only (Oracle)*	6	8
User-land software exploits	Local Privilege Escalation (Linux 2.4)	3	6
	Remote Deamon exploitation (Linux 2.4)	4	8
	Exploitation via Mass Rooter (Linux 2.4)	1	3
Kernel software exploits	Remote kernel space overflow mass (Linux 2.4)	1	3
	Remote kernel space overflow (Linux 2.4)	5	9
	Local kernel space overflow (Linux 2.4)	3	6

added' service, to a closed group of private and/or public 'customers' who subscribe to the service, and

* *non-disclosure*: vulnerabilities are fixed by software/service vendors without informing security communities, the public or customers.

Figure 6 displays the pyramid model that depicts responsible full disclosure. In the figure, time is represented by the Y axis, and the distribution of information pertaining to a vulnerability is represented by the X axis. At the top of the pyramid, knowledge about the vulnerability is limited to those who discovered the problem, and at the bottom of the pyramid the knowledge of the vulnerability is widely available in public. The two axes are strongly related: as time goes by, more people get to know about the vulnerability. The pyramid allows one to infer the following information about an adversary with respect to the exploit he/she uses in an attack: (i) the *degree of interaction/involvement* the adversary has with security communities, (ii) the *probability of success* (the higher the adversary resides in the pyramid, (a) the more time he/she has to perform an adversarial act prior to the availability of any vulnerability 'patch',

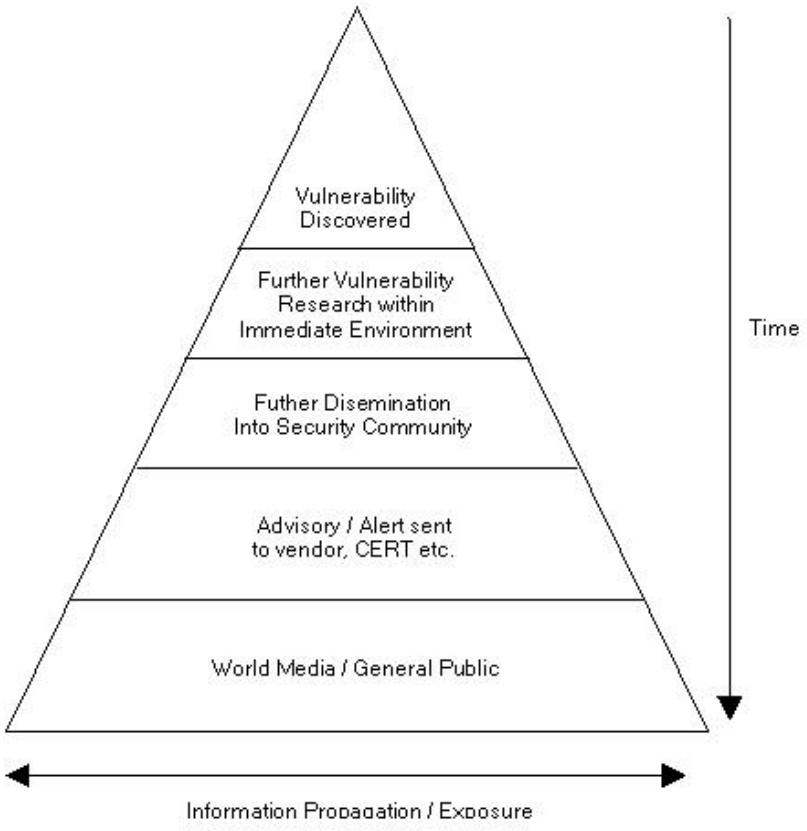


Figure 6: *A graphical illustration of the vulnerability disclosure pyramid metric (adapted from [231]).*

and (b) the less people know about the vulnerability — the lesser the likelihood of being detected), and (iii) *the capability of the adversary* (the higher the adversary is in the pyramid, the more likely, it is perhaps, that he/she needs to develop his/her own exploit to leverage the vulnerability).

- *Any attack inhibitors reduced through the use of the attack technique*: this metric informs whether the attack technique has any specific intention to reduce attack inhibitors such as the likelihood of detection, the likelihood of attribution, and the likelihood of failure to achieve a desired objective.
- *The ease with which the attack technique is implemented*: this metric aims to measure how easy it was for an attacker to implement a specific technique. This metric is only considered when the distribution level of an attack technique does not include the public domain.

Figure 6 illustrates different techniques to leverage software flaws together with the associated skill level scores. It should (again) be noted that this type of inference can fall prey to deception: if a sophisticated attacker intentionally masquerades as an unskilled adversary, the validity of the predicted results in this context would diminish.

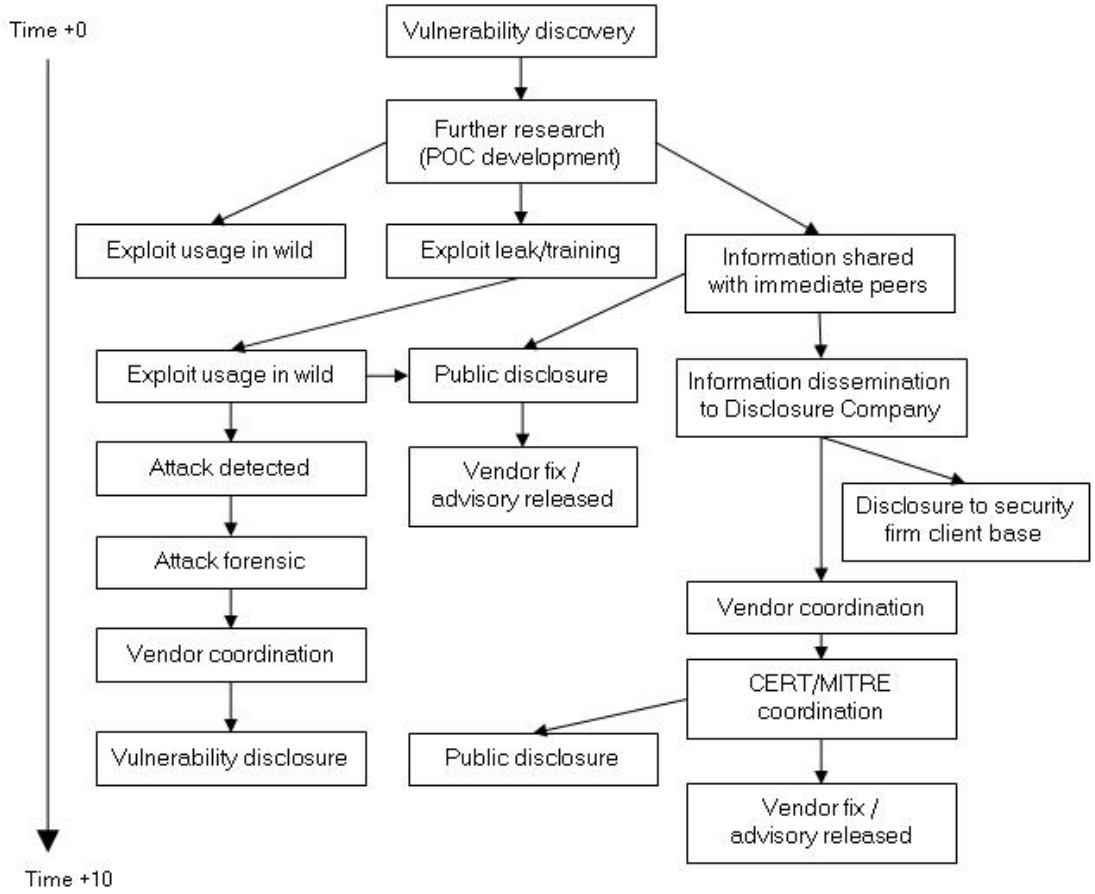


Figure 7: *A graphical illustration of the disclosure food chain (adapted from [231]).*

Parker and colleagues proposed a set of metrics for attack (or malware) tools and techniques for the purpose of characterisation of cyber adversaries. In some cases, attribution practitioners do not aim to acquire this type of knowledge, instead they are interested in determining if the malware samples have common sources as well as what their intents are. To this end, attribution can be accomplished via malware behaviour analysis and classification as presented below.

12.2.2 Malware behaviour analysis and classification

Analysis of a sample of malware in order to learn about its behaviour is among the core activities of security researchers and practitioners. Malware analysis is mostly carried out as part of the development and/or advancement of anti-virus solutions. A particularly relevant task of malware analysis is malware classification whose primary goal is to allow security analysts to skip variants of known malware (i.e., whether a malware sample exists in other versions, or belongs to a specific family [268]) and, instead, focus on new and emergent threats. Though not explicitly documented, methods in malware behaviour analysis and classification can be leveraged to assist the attribution task in the two following

perspectives:

- *intent inference*: analysis/classification of malware behaviours ultimately reveals the intent of the malware sample under investigation (and thus, of the threat actor) through an identification of (i) the type of malware sample (e.g., *virus* or *worm*, *backdoor* or *keylogger*) or (ii) the existing malware samples to which the malware sample is closely related; and
- *origin classification*: grouping malware samples that exhibit similar behaviours potentially identifies those with common origins, based on which the possible class of adversaries behind a spear phishing attack can be inferred.

Approaches to malware behaviour analysis and classification that are potentially applicable in the attribution task are presented below. Different from malware classification in the context of malware detection where a classification task is carried out to discriminate between malicious code and benign code, the methods to be presented are devised for the purpose of classification among malware samples — based on either the structural information (as in static methods) or behavioural information (as in dynamic methods) of the malware.

12.2.2.1 Static malware analysis and classification Static malware analysis involves examining the content of the executables, or the source code, of malware samples in order to derive patterns of their behaviour, based on which malware classification is performed. Surprisingly, malware classification has long been practised in a manual fashion (e.g., via memorising, looking up description libraries, or searching sample collections in databases [183]). It was not until recently, due to the number and diversity of Internet malware[69], efforts to automate this process in a large-scale manner have been made. Some representative work in automated malware classification based on static analysis are discussed in this section.

Gheorghescu and colleagues [110] implemented an automated malware classification system which is independent of the malware class and language. The classification system is purported to deliver results in a reasonable amount of time. To facilitate classification, the researchers first filtered static library code and expressed each malware sample as a control flow graph (CFG) in which nodes represent basic blocks (a continuous sequence of instructions that contains no jumps or jump targets) and arcs represent execution flow between the basic blocks. To simplify the comparison process, the researchers took into account only nodes in CFGs to calculate the distance between two malware samples and ignored the arcs between them. In this way, matching two malware samples essentially involves conducting a comparison between the basic blocks in the two respective CFGs to determine if they are identical in functionality. For an efficient classification of malware samples, Gheorghescu and colleagues investigated three approximate matching methods: *edit distance* [216, 293] (a method which aims to minimise the number of symbol insertions, deletions and replacements that is required to transform one string into another), *inverted*

[69]For example, the malware Agobot has been observed to have more than 580 variants since its initial release in 2002, and 43,000 new variants of backdoor trojans and bots were found during the first half of 2006 [16].

index (an efficient information retrieval method based on a set of key words each of which has an associated list of pointers to the origin, such as a document or paragraph, of the word), and *Bloom filters* [27] (a method for membership queries with efficient storage and a search time that is linear on the collection size). The researchers evaluated the three methods from the perspective of run time, storage space, and classification accuracy and found that the three matching algorithms all return similar classification results within a reasonable time.

Similar to the research work by Frantzeskou and colleagues [100, 101, 102, 103] in software authorship (see Section 12.1.1), Kolter and Maloof also adopted the n-grams method to represent malware samples, and used machine learning methods such as Naïve Bayes, decision trees and support vector machines for classification. However, Kolter and Maloof [158], rather than determining authorship, aimed to discriminate malware code from benign code, and subsequently group the identified malware samples into different categories. The researchers first converted each executable to hexadecimal code in an ASCII format, then constructed n-grams from the hexadecimal code by combining each four-byte sequence into each n-gram. This classification method was later improved by a variable-length n-gram approach by Reddy and colleagues (c.f., [254]) who determined to use 'dynamic' n-grams to capture meaningful sequences of different length. It was reported that the classification method using variable-length n-grams empirically outperformed the precedent fixed-length n-gram approach [254].

The obvious merits of static malware analysis lie in its simplicity (an actual execution of the malware is not needed) and completeness (different execution paths are covered). However, since static malware analysis generally captures the structural information expressed in the malicious code, rather than the actual behaviour of the code, it is well-known for its susceptibility to inaccuracies when dealing with obfuscated[70] and *polymorphic*[71] malware. As a result, malware samples that essentially perform the same function can possess very different signatures in their binary code. Various obfuscation techniques have been reported to be able to defeat many static malware analysis [187, 215, 248], including n-gram methods [150, 158]. The development of obfuscation techniques and polymorphic malware play an important part in the impetus behind the paradigm of dynamic malware analysis.

12.2.2.2 Dynamic malware analysis and classification Dynamic malware analysis involves executing a malware sample in a controlled environment, and observing and recording the behaviour of the malware. There exists a range of dynamic malware analysis tools that assist in this task (c.f., [15, 20, 145, 185, 214, 296]), and the tools that increasingly gain popularity in practice include CWSandbox[72], Norman Sandbox[73], and ANUBIS [19]. Reports produced from these tools usually provide low-level and detailed descriptions of malware behaviour which allow attribution practitioners to perform classification to determine the class, and infer the intent, of the malware samples of interest. For

[70] To avoid detection, a malware author often introduces small changes to the source code of a piece of malware, or modify the binary (using techniques such as obfuscation) in order to produce multiple variants of the malware.

[71] Some types of malware are also designed to dynamically evolve through modifying their own code (c.f., [312]).

[72] CWSandbox. `http://www.cwsandbox.org/`.

[73] Norman Sandbox. `http://www.norman.com/microsites/nsic/`.

example, if a malware sample is classified as a keylogger, it is plausible for an attribution practitioner to infer that the malware (and, thus the attacker) probably aims to capture personal credentials. Classification methods for dynamic malware analysis published in the literature mostly address two major issues: (i) formulating an efficient way to represent malware (run-time) behaviour (e.g., malware behaviour has been represented in an *opaque object* [183], feature vector [16] and behaviour profile [21]) based on the low-level behaviour reported by malware analysis tools, and (ii) devising an appropriate and efficient method to classify the represented malware behaviour.

Motivated by the need to tackle obfuscated, self-modifying and complex malware samples increasingly encountered in the wild, Lee and Mody [183] proposed a behavior-based automated classification method based on distance measure and machine learning. Observing that prior efforts to represent program behaviour based on API call history and high level functional attributes are inadequate due to their considerable information loss, Lee and Mody identified major properties of malware behaviour that need to be captured: ordering with respect to time, actions taken by the malware, and subsequent states of the environment in which the actions take place. In order to capture such a rich semantic representation that is able to scale to account for unknown behaviour types, Lee and Mody used *opaque objects*[74], instead of the classical feature vectors, to represent malware behaviour. The researchers devised a custom distance function (adapted from string edit distance) to measure similarity between objects, and categorised malware samples using *k-medoid* clustering[75]. The researchers conducted their experiments in a virtual network where all occurring events are captured by monitor agents and recorded to a log file. Lee and Mody concluded that it is important to combine dynamic and static analysis in order to improve the classification accuracy.

Bailey and colleagues [16] studied and identified the limitations of anti-virus systems. They found that existing anti-virus systems failed to deliver satisfactory classification results. More specifically, they did not provide:

- a consistent classification (e.g., different anti-virus systems produced different classification results for the same collection of malware samples),
- a complete classification (e.g., a portion of malware collection went undetected and unclassified), and
- a concise representation of malware (e.g., existing anti-virus systems do not provide sufficient (if any) explanation for their classification as well as meaning of the class labels).

To address these limitations of anti-virus systems, Bailey and colleagues introduced a behaviour-based malware clustering system in which malware behavior is described in terms of *system state changes* (e.g., files written and processes created) rather than in terms of sequences or patterns of system calls. *Behavioral fingerprints* constructed by

[74]Opaque objects offer rich syntax, allowing for description of complex malware behaviour at the cost of losing the ability to construct statistical/probabilistic views of the data.

[75]K-medoid clustering is similar to k-means clustering except that k-medoid clustering forms clusters based on the representative objects, or *medoids*, rather than the means of the objects.

system state changes, according to Bailey and colleagues [16], are more invariant and useful. In particular, they can be directly used in assessing the potential damage incurred, enabling detection and classification of new threats, and assisting in the risk assessment of the identified threats. Once the behaviour fingerprints had been computed for all the malware samples, a pairwise single-linkage hierarchical clustering algorithm was used to group malware samples with similar behaviours. Bailey and colleagues carried out their experiments in virtual machine environments and evaluated the results from the perspective of classification consistency, completeness and conciseness. The empirical results demonstrated that the proposed dynamic malware behaviour classification system outperformed existing anti-virus systems in all the measured aspects. More specifically, the system delivered results that are consistent (all the malware samples with identical behaviours are assigned to the same clusters), and concise (a description of the threat imposed by a malware sample is provided on its successful detection). The results also achieved a much higher degree of completeness (around 91% of the malware samples were classified in comparison to 33% achieved by many anti-virus systems).

Both the research work by Lee and Mody [183] and Bailey and colleagues [16] are based on unsupervised classification, leaving room for an investigation of supervised methods to classify malware behaviour. This initiative was taken by Rieck and colleagues [130] who constructed a corpus of training samples necessary for supervised learning by using existing anti-virus systems to label malware samples *a priori*. Rieck and colleagues captured malware behaviours in feature vectors where each feature reflects a behavioral pattern (such as opening a file, locking a mutex, or setting a registry key) extracted from the malware run-time behaviour reported by CWSandBox. Support vector machines were applied to the collection of feature vectors to build a classifier for each of the malware families (e.g., Sality classifier, Doomber classifier and Allaple classifier). The researchers evaluated their method on a large corpus of malware samples obtained from honeypots and spam-traps and reported that up to 70% of malware samples that had gone undetected by anti-virus systems were correctly classified by their approach. However, it is obvious that the efficacy of the approach proposed by Rieck and colleagues depends on the accuracy of the anti-virus software used to label the malware in the first place.

Built on the idea of representing software behaviour as system state changes [16] (see above), Bayer and colleagues [19] proposed a clustering system that aimed to achieve a better precision and scalability than previous approaches. The differences between the proposed system and its precedent [16] lie in the formulation of behaviour profiles and the algorithm used to perform clustering. In more detail, Bayer and colleagues modelled the behavior of a malware sample in terms of operating system (OS) objects (i.e., resources such as a file or registry key), operations that are carried out on these objects (i.e., generalised forms of system calls, for example, the two specific processes `NtCreateProcess` and `NtCreateProcessEx` are concrete instances of the OS operation `create`), dependencies between OS objects and comparisons between OS objects. A behaviour profile defined in this way essentially trades precision for generalisation. This trade-off was justified by the fact that the constructed profiles would withstand detection evasion techniques such as those that introduce superfluous operations into the malware. To facilitate clustering, the behaviour profiles were transformed into feature vectors, and *locality sensitive hashing* (LSH)[76] was employed to compute an *approximation* of the set S of pairs of similar

[76]Locality sensitive hashing is a method that purportedly hashes a collection of points in such a way

points. Bayer and colleagues then removed all pairs in S that, in fact, did not contain similar points and sorted the remaining pairs by similarity. The result of this was an approximate, single-linkage hierarchical clustering, from which an agglomerative clustering was subsequently constructed. Employing LSH in the clustering task therefore exempts users from having to compute the distances between all pairs of points (i.e., pairs of feature vectors) in the set (as required in most clustering algorithms) as well as having to choose the number of clusters *a priori* (as required in k-means clustering). Bayer and colleagues reported that their clustering method can accurately classify malware samples which exhibit similar behaviour; notably, it was able to cluster more than 75 thousand samples in less than three hours (in comparison to an estimate of 6 months that would be required to use the clustering method proposed by Bailey and colleagues [16] on the same set of behaviour profiles).

Though behaviour classification based on dynamic malware analysis proved to be much more reliable and accurate than those based on static analysis, it inevitably results in varying degrees of time-delay. In addition, the majority of methods have a common limitation: the behaviour of malware being captured is limited to the capability of the tools and whatever can be observed in a virtual environment. For example, a malware sample potentially has multiple execution paths whose executability depends on various environment factors; as a result, a malware dynamic analysis tool is, in many cases, able to observe only one of these paths. Furthermore, some types of malware are intended to stay dormant until certain time and date, or until they successfully connect to their command and control server and receive commands from the server, or until they received user input (e.g., a user entering information into a website). Though researchers in the community have attempted to address one or more aspects of the limitations (c.f., [214]), it seems that, at present, perhaps the best approach is to address these limitations by combining dynamic and static analysis as recommended by Lee and Mody [183] or by making use of high-level interaction honeypots to observe a wider spectrum of behaviours of malware samples (see Section 15.1).

In general, malware samples can be characterised from different attribution perspectives, the results of which serve as a basis for an investigation of adversary classes potentially authoring the malware samples, by means of inference or classification. Presented above are methods to describe malware samples in some sort of malware behaviour profiles. Other researchers, such as Betjtlich [24] and Cloppert [58] also proposed to characterise malware samples based on an *explicit set of features* which either constitutes the modus operandi of a malware-based attack, or represents the malware characteristics relevant to attribution. More specifically, Bejtlich [24] identifies 20 characteristics of a piece of malware that need to be evaluated in attributing an attack: (1) *timing*, (2) *victims or targets*, (3) *attack source*, (4) *delivery mechanism*, (5) *vulnerability or exposure*, (6) *exploit or payload*, (7) *weaponisation technique*, (8) *post-exploitation activity*, (9) *command and control method*, (10) *command and control servers*, (11) *tools*, (12) *persistence mechanism*, (13) *propagation method*, (14) *data target*, (15) *data packaging*, (16) *exfiltration method*, (17) *external attribution*, (18) *professionalism*, (19) *variety of techniques*, and (20) *scope*. Cloppert [58] attempted to construct the *modus operandi* of the malware-based attack based on its *kill chain* — the sequence of events that must occur for an attack to successfully achieve its objective. In this context, the kill chain is defined to include the following

that similar points have a much higher collision probability than points that are distant.

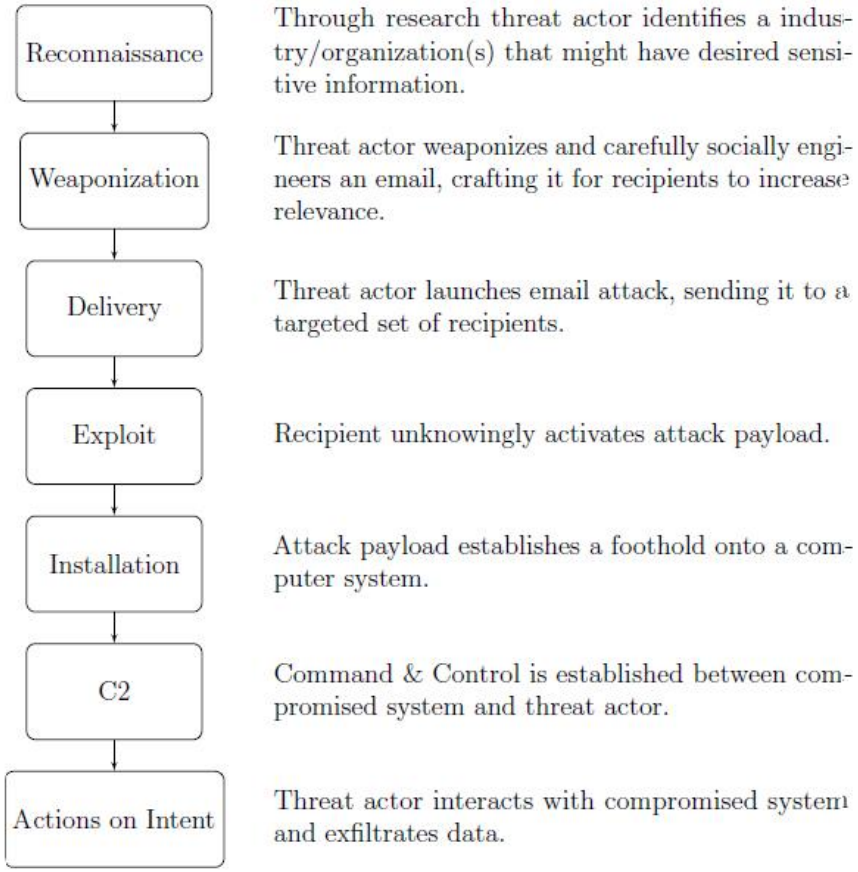


Figure 8: *An illustration of the email kill chain [58].*

sequential phases: *reconnaissance*, *weaponisation*, *delivery*, *exploitation*, *Command and Control (C2)*, and *exfiltration* [58] (see Figure 8). Clopper further hypothesised that some of the phases (or elements of the phases) of the kill chain are likely to be *immutable* (infrequently-changing) across different attacks carried out by the same adversary, and consequently culminate in his/her behaviour profile. The modus operandi of an attack constructed in this manner is, therefore, able to capture the *adversary behaviour*, allowing for an inference of common groups of adversaries behind a collection of malware-based attacks.

12.3 Discussion

Regarding software authorship, it is demonstrated in the literature that linguistic features based on n-grams tend to be more effective than programming features. However, n-grams are susceptible to obfuscation techniques (c.f., [150, 158]), and thus, it might be possible to complement n-grams with programming features to improve the overall result accuracy. It is important to note that various attribution methods pertaining to email authorship can be adapted for use in software authorship. From a larger perspective, determination of software authorship in the context of spear phishing faces a similar problem to email

authorship: there often does not exist a collection of malware samples with known authors. In this scenario, it is necessary to perform clustering on the collection of malware samples under investigation before a classification of unseen malware samples can take place.

With regard to identification and characterisation of adversary classes, results provided by malware analysis can be used to infer characteristics of the adversary class likely responsible for a malware-based attack. In this respect, attribution practitioners can capitalise on the method proposed by Parker and colleagues by replacing general attack tools with pieces of malware, attack techniques with the techniques employed in the malware, and methods to deduce information about the adversaries for specific attack tools with those for specific malware samples. By the same token, attribution practitioners may wish to perform classification of the malware behaviours produced from the analysis results in order to determine the intent and common origins of the malware samples under attribution. To this end, there exist a range of methods (the representatives of which are presented above) to assist the attribution practitioners to accomplish this task.

13 Attribution of evidence in the hook component — (spear) phishing websites

Similar to spear phishing emails, phishing websites bear all the three types of evidence: ownership evidence (IP addresses and domain names), individual characteristic evidence (the writing style reflected in web pages' textual content), and class characteristic evidence (metadata associated with the websites as well as the techniques employed in the construction of the websites). The sections that follow present a speculation on the extent to which the mentioned types of evidence contribute to the attribution task.

13.1 Website source tracking

Unlike those in the lure component, the IP addresses associated with the hook component of a spear phishing attack are generally not spoofed. Therefore, the IP address of a phishing website (either obtained from the link in the lure component or via look-up by public DNS) can be used to identify the machine that hosts the phishing site. However, this knowledge is only at the tip of the iceberg of the attribution problem: the identified machine is often a zombie machine under control of the spear phisher. In traditional attacks, a zombie machine is typically compromised and consequently configured to host a phishing website. In advanced and sophisticated attacks, a zombie machine is only one among hundreds or thousands of other zombie machines that act as front-line proxies directing requests to the back-end server systems that host the phishing website (as witnessed in the case of *Rock Phish* and the more advanced technique known as *fast-flux phishing*, c.f., [212]). In fast-flux phishing, the phishing domain name is configured to be resolved to a set of five different IP addresses for a short period, and then shift to another five and so on. In a most recent form of fast-flux, known as 'double-flux', even the IP address of the DNS server under control by the phisher changes its IP address rapidly. Not only do Rock Phish and fast-flux phishing make a take-down strategy harder and consequently extend the life time of phishing sites significantly (see Figure 7), they also render the attribution task tremendously hard. To confound and hassle the defenders further, the back-end servers are usually hosted by rogue networks (see Section 2.4). It appears that there is no *automated* technique that is able to identify the servers that host phishing websites in the case of Rock Phish and fast-flux phishing, except for a very few research ideas. For example, Zhou and colleague [324] proposed a self-healing, self-protecting collaborative intrusion detection architecture to trace-back fast flux phishing domains. The method proposed by Zhou and colleagues requires a collaboration of intrusion detectors from different ISPs to monitor the fast-flux machines and correlate the suspicious communication patterns of the machines.

13.2 Website authorship attribution

Like emails, web pages also contain textual content presented in a certain structure and layout. Therefore, attribution methods presented in Sections 5 and 11.2 can, in principle, be used to provide important clues to web-page authorship. In the context of (spear)

Table 7: *An illustration of the lifetime of rogue websites for non-rock phishing, Rock Phish and fast-flux phishing attacks [212].*

	#	Mean lifetime (hrs)	Median lifetime (hrs)
Non-rock	1707	58.38	20
Rock domains	419	94.26	55
Rock IPs	122	124.9	25
Fast-flux domains	67	454.4	202
Fast-flux IPs	2995	124.6	20

phishing, determination of this type of authorship is usually impracticable, due to two major factors: (i) the majority of phishing web pages are automatically generated by phishing toolkits which are widely available on the underground market, and (ii) many phishing websites are purported to mimic the look and feel of the websites they are impersonating. Application of authorship attribution methods to phishing web pages would likely identify those either using the same phishing toolkit or impersonating similar websites. Therefore, unless an adversary deliberately leaves behind some kind of watermark in the web-page textual content (or web-page source code), it is unlikely that there exists a method to identify the actual adversary associated with a phishing website.

13.3 Adversary class identification and characterisation

There has been little work on attribution of phishing websites in terms of their origins (it is supposed that phishing websites originating from a common source may be associated with the same class of adversaries). Representative research efforts in this direction include those attempted by Wei and colleagues [307], James [144], McCombie and colleagues [203][202], and Layton and colleagues [180] [181]. The researchers either combined information of both the phishing email (Lure) and website (Hook) to gain some insight knowledge of those responsible for an attack, or entirely used information pertaining to the phishing website to carry out this task.

The most elementary among the research efforts to be presented involves grouping spam incidents with respect to spam web hosting by Wei and colleagues (c.f., [307]). In their study, the researchers found that most spam domains use wildcard DNS records to support non-existing machine names, and believed that the hosting IP addresses are much fewer than the number of domains (i.e., a large number of domains hosted on a limited number of hosts). With this vision, Wei and colleagues performed clustering of spam domains on IP group (which consists of one or several IP addresses that simultaneously host the same domain), and found that the hosting IP addresses (4,000 in total) in the experiment were distributed among 1391 network blocks residing in 61 countries. Top of the list are IP addresses located in China (with some found to host more than 8,000 real domains in a period of three months).

James [144] identified key items that can be used to identify groups of phishers. The items include bulk-mailing tool identification and features, mailing habits (e.g., their specific patterns and schedules), email origination hosts, phishing server hosts, layout of the

hostile phishing server (e.g., the use of HTML, JavaScript, PHP, and other scripts), naming convention of the URL used for the phishing site, IP address of the phishing site, and features pertaining to emails such as assignment of phishing email account names, choice of words in the subject line, and the time-zone of the originating email ([144] cited in [203]).

Motivated by the conjecture that phishing activities are carried out by a small number of highly specialised criminal groups, rather than a large number of random and unrelated individuals using similar techniques, McCombie and colleagues [203] attempted to group attackers with a common *modus operandi* (or with the same level of criminal organization) based on the similarities between phishing emails and phishing websites. The researchers extended the set of features identified by James (see above) to include additional features such as (i) the text, metadata and header information pertaining to email, (ii) the phishing web pages and web hosts together with their directory structure and files, and (iii) any characteristics that may serve as links between separate incidents. The features of phishing email used by McCombie and knowledge have been presented in Section 11.3. Regarding phishing sites, examples of the features used in the investigation (which are mostly metadata) include: details of the phishing sites' URL (i.e., commonly used words such as *Index*, *secure*, *online*, *agreement* and *login*), host IP address (categorised in terms of IP subnets), domain registrant, domain registrar, country, classless inter-domain routing (CIDR), operating system (e.g., Linux, Win32 and Unix), and web server type (e.g., Microsoft and Apache). The study was conducted using simple decision rules on a large collection of phishing email and website samples collected by a major Australian financial institution. The experiment reported an appealing result: three in the six identified groups were responsible for up to 86% of all phishing incidents.

In a similar vein, Layton and colleagues [181] focused on grouping phishing websites (based on the websites' source code) that are likely to have a common origin, or 'provenance'. The purpose of their work is two-fold. On the one hand, Layton and colleagues investigated the feasibility of grouping phishing attacks using websites' data alone. On the other hand, they proposed an automated approach for determining provenance in phishing websites that can deal with evolving phishing datasets. Recall that other research work with related goals (such as the work by McCombie and colleagues presented above) either wholly or party rely on a predetermined set of features to be used for the clustering or classification task. While making use of a predetermined set of features can be efficient for the task at hand, it is not very effective in tackling phishing attacks that evolve overtime. Layton and colleagues attempted to rectify this drawback by dynamically deriving this set of features from the phishing dataset under investigation. More specifically, the researchers first applied a 'bag-of-words' model — a popular model in text categorisation (as discussed in Section 6) — to the phishing dataset, which yielded a large and 'noisy' set of features. The researchers then experimented grouping phishing websites using two alternative approaches. In one approach, PCA was utilised to reduce the size of the original set of features before clustering was carried out on the reduced feature set using the k-means and DBSCAN algorithms. In the other approach, CLIQUE (a subspace clustering technique that can deal with high dimensional data) was directly applied to the original feature set[77]. The experimental result demonstrated that CLIQUE achieved the best clustering but was not very stable due to its extreme sensitivity to the parameters' value,

[77] Please refer to Sections 6.2.2 and 7.5.6 for a discussion of PCA, k-means, DBSCAN and CLIQUE.

while the k-means and DBSCAN algorithms proved to be more stable within a small band of variation in the parameters' value. Though DBSCAN seemed to produce more accurate results than the k-means algorithm, DBSCAN was able to perform clustering for only a portion of the dataset. Regarding the primary goal of the research, Layton and colleagues concluded that phishing websites should not be used alone in grouping phishing attacks; instead, they should form part of a larger system, combined with other attribution methods in order to generate verifiable results. The method proposed by Layton and colleagues can be enhanced by performing clustering only on the portions of the impersonating webpage that are different from the impersonated webpage (c.f., [180]). This technique takes advantage of the fact that a set of phishing websites mimicking a legitimate website tend to be much similar, and thus clustering based on features derived from the 'diff' portions (outputted by the execution of a *diff* program that take as input a phishing website and its impersonated website) would be sufficient.

13.4 Discussion

The rogue website associated with a spear phishing attack is not different from that of other types of phishing. As such, attribution of this component of a spear phishing attack can directly benefit from experience and lessons from existing studies relating to phishing websites as those presented above. In general, evidence collected from fraudulent websites is unlikely to reveal information regarding the actual adversaries (unless some kind of watermark exists). Rather, they can (at most) contribute by disclosing the common origins of spear phishing attacks, and by inferring characteristics of the class of adversaries who may be associated with such attacks. Existing methods in the literature mostly utilised the websites' source code and metadata to perform clustering of phishing websites. It may be possible that the deception techniques employed in phishing websites could provide important clues about the adversaries, and thus should be considered as part of the features in the thematic clustering scheme. To this end, it is possible for one to reuse, or adapt from, the collection of deception techniques compiled by Drake [82] as following[78]:

- making use of invalid SSL Certificates (i.e., a URL that begins with `https://`) to convince the victim that information is being transmitted over a secure connection,
- buying some time before the victim checks his/her account (e.g., by informing the victim that the provided information needs to be verified within a period of time),
- processing the submitted information (a sophisticated phisher may verify the submitted information, e.g., verifying the submitted credit card number, or sending the username and password to the legitimate company to verify whether or not they are valid), or
- exploiting the power of Javascript to
  - create a fake address bar,
  - create a fraudulent pop-up over the real company site,

[78] A more exhaustive collection of such techniques can be consulted at [143].

- disable right-click to prevent the victim from viewing the page's source code, and
- checking the browser (if the phishing website is designed to exploit the vulnerabilities of specific browsers).

With respect to adversary characterisation, the method proposed by Parker and colleagues (see Section 12.2.1) is potentially useful to infer characteristics of the attacker based on the technical methods utilised to deceive a victim and capture information he/she entered into the phishing websites. For example, if the man-in-the-middle or man-in-the-browser methods are employed in the attack, one might conclude that the attack is mounted by a sophisticated and skillful adversary.

14 Attribution of evidence in the catch component

The catch component refers to the phase of a (spear) phishing attack where the stolen credentials/information are transferred from the hook component to the spear phisher. In general, the nature of the catch component is different from that of other components such as Lure and Hook in that it is a 'prospective' crime scene, and pieces of evidence (except for the ownership evidence) provided by this component are not readily available to the defenders. This mandates attribution practitioners, rather than conducting a passive forensic investigation, to take an active approach in order to disclose, as much as possible, the *latent* evidence associated with the catch component. There are two major types of evidence potentially found in the catch component: ownership evidence (e.g., email and IP addresses pertaining to a catcher) and class characteristic evidence (e.g., mechanisms to 'catch' the stolen data).

Ownership evidence are often available to attribution practitioners: email/IP addresses are usually embedded in the malicious code associated with a hook component to enable communications between the hook component and the catcher as part of the exfiltration process. In this case, the found email address would be almost certainly that of a web-mail, and the IP address would be that of a stepping stone, zombie or node in a rogue network. For this reason, researchers have attempted to investigate methods to track catchers without entirely relying on the ownership evidence.

Active methods for tracing catchers (or attackers) are presented below. Active methods to disclose and analyse class characteristic evidence — as a part of a larger attribution scheme whose scope exceeds the catch component — will be presented in the next section.

14.1 Catch destination tracking

As mentioned in Section 3, many cyber attacks operate entirely with one-way communication manner (such as virus propagation or DDoS attacks whose mere goals are to propagate malicious programs or to flood the target with meaningless traffic, respectively), in which case advanced attackers stay reasonably safe and sound behind protection layers of IP spoofing, zombie-nets and rogue networks. In order to exfiltrate information from a victim, attacks pertaining to spear phishing attacks, as well as other types of cyber espionage, require two-way communications, necessitating in all cases an existence of reverse flows from the victim to the attacker (possibly via intermediaries). In principle, if one is to persistently follow the reverse flow, he/she will ultimately reach the attacker. In reality, the successful realization of such a vision is left to the capability of the defender to find his/her way to the offender and the capability of the offender to hide and protect himself/herself.

Different from traceback methods (see Section 11.1.1) which involve passively identifying and following the digital trail left behind by an attacker, an *active tracing* method in this context allows one to initiate a reverse data flow and actively monitor/trace the data toward the catcher. Active tracing methods are centred around the utilisation of certain decoy elements in the data to be sent back to the attacker. Several forms of such an element (in an ascending order of their respective activeness) include: *watermarks*,

honeytokens, *web bugs* and *beacons*. Though the application of these decoy elements to attribution of spear phishing may be new, the decoy elements are themselves not new and have long been used in both the whitehat and blackhat communities. A watermark is typically a piece of information that uniquely identifies the digital item to which it is attached. Watermarks are extensively studied in the realm of software protection and copyright. A honeytoken refers to any element (a piece of information, a credential, an image, an account, or a file) any access to which serves as strong evidence that the system has been compromised. A web bug is a silent piece of code that records certain activities and information of web and email users (e.g., web browsing habits and demographic data such as gender, age and zipcode), and reports the information to its owner. A web bug is usually implemented as a transparent image inserted into the HTML portion of an email, but it is also witnessed being implemented in Microsoft Office documents [269]. Web bugs are widely used by phishers to obtain information about their victims. Finally, a beacon is a piece of code secretly embedded in a bogus document that is activated and signals to its remote master when the document is opened. A watermark, web bug and beacon must meet an overarching requirement: that of concealing itself as much as possible in a containing item. General methods of hiding information (i.e., masking, repacking, dazzling) and attracting targets (i.e., mimicking, inventing and decoying) is discussed at [22], and a discussion of the theme in the context of cyberspace is available at [316]. Approaches making use of decoy elements that are potentially applicable to the attribution of the catch component are presented below.

14.1.0.3 Watermarking Watermarks in the discussed context should be easy to embed and detect (or retrieve) due to the large amount of data each node on the network needs to process. Similar to passive tracing methods discussed in Section 11.1.1, active tracing methods can be content-based (via analysing the payloads of the packets) or timing-based (via inspecting the timestamps of the packets). One early form of active content-based tracing method is 'sleepy watermark tracing' proposed by Wang and colleagues [302]. The researchers created a watermark for text-based network application[79] in the form of a *virtual null string*, a string that appears null to end users of the network applications (e.g., `telnet` and `rlogin`).

For example, the string

'`See meabc\b\b\b \b`'

will be displayed as '`See me`' to the end users.

To trace the attacker, the proposed watermark needs to be injected into backward traffic by having the network server applications (e.g., telnet and rlogin) inject the watermark into their response traffic. In order to trace along the connection chain, Wang and colleagues proposed a mechanism to find and match adjacent connections that belong to the same connection chain. One advantage of the technique is that attribution can be accomplished through a large number of stepping stones, provided the data is not encrypted.

[79] According to Wang and colleagues [302], hiding a watermark in text is more difficult than hiding data in pictures and sounds.

In order to correlate encrypted connections, active tracing methods were proposed that are based on timing characteristics of (rather than contents of) connections (c.f., [300, 301, 235, 236, 237, 238]). Due to the fact that a timing-based watermark is embedded through manipulation of the timing characteristics of the stepping stone connections, active tracing based on this type of watermark is able to handle encryption imposed by secure protocols such as SSH and IPsec. Unlike passive tracing (described in Section 11.1.1) which is unnoticeable to attackers, active tracing may be detected by the attackers. Consequently attackers may deliberately instruct stepping stones to introduce 'noise' into the packet flow in order to defeat tracing efforts. For instance, the stepping stones can insert timing perturbation or chaff traffic in the stepping-stone streams, adversely affecting timing-based correlation and limiting traceability. In addition, the packets passing through stepping stones may be repacketised to improve network performance (or perhaps to further confound timing tracing). In this respect, it is important that active watermark-based timing approaches are sufficiently robust to resist one or more of these countermeasures/conditions. To this end, the proposed watermark-based timing methods are able to deal with timing perturbation (c.f., [300, 301]), both timing perturbation and chaff (c.f., [235, 236, 237, 238]), and TCP repacketisation (c.f., [250]).

Instead of injecting a watermark into the reverse flow, another approach is to embed watermarks in documents or files. For instance, Bowel and colleagues [31] computed a watermark as a key cryptographic hash function using words extracted from the containing document. Provided that a document with an embedded watermark is among those exfiltrated by a spear phisher, it can be detected along the way, and consequently informs the defender of its travelling path. To make the watermark invisible to the end user or rendering tools, watermarks are usually hidden in the document's metadata portion or in the comments within the document format structure [31].

However, there are two major factors that impede the adoption of watermark-based tracing methods on a global scale: (i) the necessary preposition of dedicated watermark detectors in the open network, and (ii) the privacy concerns associated with searching for watermarked data/files. Therefore, methods based on watermarks are more suitable for implementation within a corporate network for detection of insider threats.

14.1.0.4 Honeytokens/web bugs A honeytoken refers to any decoy element intentionally designed to trap and detect those that access and use the element. Honeytokens can be in the form of personal credentials (e.g., login details), an image, a file, or a piece of information (e.g., a medical record of a patient). Honeytokens purposefully have no use in practice, therefore the access to, and/or the use of, honeytokens is a strong indication that a computer system has been compromised. Since access and use of honeytokens are logged, information about the attacker to a varying degree can be obtained. Research efforts in using honeytokens in cyberspace attacks include [276] which used bogus medical records, credit card numbers, and credentials to detect malicious insiders, Bowen [30] used *honeyflow* — a traffic flow that contains within it trap-based decoys (decoys that are detectable on their own such as online banking logins, credit card numbers, login accounts for online servers, and web-based email accounts) — to detect attackers who are eavesdropping network traffic.

Most relevant to the context of spear phishing is the use of honeytokens and web

bugs by McRae and Vaughn [194] to identify the source of phishing attacks. The idea underpinning the proposed method involves completing the phisher's web form with an HTML image tag of a one-pixel by one-pixel image and an HTML web page link. The image acts as a web bug since it is designed to gather information about the phisher. At the same time, both the image and the HTML file are honeytokens because they have no practical use, apart from serving as baits. Once the phisher views the results in an HTML enabled environment, the image will be retrieved from the server. In response to the image being loaded or the hyperlink being accessed by the phisher, a referral is generated in the web logs on the tracking server. According to McRae and Vaughn, information provided in the log includes: (i) the IP address of the actual machine where the results were viewed, (ii) the web page where the phisher viewed the results (iii) the browser type that was used to view these results, and (iv) a guess at the operating system of the phisher's machine. An advantage of this approach is that it tracks down a phisher to the IP address of the phisher's machine rather than intermediaries such as stepping stones or zombies. However the approach bears a major weakness: it only works in an HTML enabled environment, and thus viewing the results of phishing forms in text-only viewers is a simple but absolute countermeasure.

14.2 Discussion

Methods to reveal adversaries associated with information-theft attacks have been studied, either based on deception or by actively tracing the paths of the stolen information. In principle, many of the existing methods can be applied to the attribution of the catch component of a spear phishing attack. However, at the current stage, methods using web bugs and beacons are likely to be effective only when dealing with unsophisticated attackers (unsophisticated spear phishers). Advanced attackers would nullify at ease the effectiveness of the methods, for example, they may view the stolen information in a text-based environment, disable beacons/web bugs, cut off network communications, disable/kill local host monitoring processes, or exfiltrate the stolen item via a web-browser and avoid opening the file locally. The efficacy of using watermarks and honey tokens to apprehend the attackers is also restrained due to its dependence on the preposition of dedicated monitoring machines on the travelling path of the watermarked data and subsequent use of the stolen elements by the attackers. Such monitoring machines can be, at most, deployed fully within a corporate network, but only very sparingly on the internet. This makes *detection* of general attacks, and *attribution* of insider attacks, possible, but attribution of attacks initiated from outside the network still beyond the realm of present technical capabilities. Furthermore, in the case of spear phishing as espionage, the use of honey tokens, or watermarked items, is unlikely to be detected since the actual stolen items will not be used; instead, the information contained in the items is extracted as part of the intelligence collection process. Last but not least, as mentioned in Section 2.1, sophisticated catchers may egress stolen data through a covert channel. Hence, methods and techniques from the study of covert communications may be of significant help to investigators in uncovering this type of communication to trace the catchers. However, a discussion of these methods and techniques together with their potential assistance in catching (spear) phishers is beyond the scope of this survey.

15 Proactive/active attribution of spear phishing attacks

Attribution methods discussed in the previous sections, except for those in the catch component, passively examine and analyse pieces of evidence left behind by a spear phisher. As such, they are considered reactive attribution methods. Potentially much more information about the attack can be obtained via proactive/active attribution approaches. By *proactive*, I mean that a defender takes initiative to prearrange and set up things in such a way that certain goals of attribution can be achieved in the best way possible when a spear phishing attack occurs; and by *active*, I refer to real-time observations/interactions with the attackers or machines taking part in the spear phishing attack under investigation. There are many forms and many methods for proactive/active attribution, the majority of which leverage a notorious technique of adversaries: that of *deception*. Examples of proactive attribution methods including the preposition of traceback and watermark detecting capabilities in nodes and applications in the network, the insertion of decoy elements (e.g., watermarks and beacons) into selected data items, and the generation of bogus data, documents and files. Examples of active attribution methods include observations of real-time attacking processes, and interactions with the phisher by sending him/her data with decoy elements. Large scale realisation of proactive/active approaches relevant to attribution can be conducted by means of honeypots.

15.1 Honeypot systems

A *honeypot* is a system or resource which is deliberately designed to attract, detect and analyse adversaries. A network of honeypots is referred to as a *honeynet*. Due to its low false positive and false negative rates[80], honeypots are very efficient tools for intrusion detection and analysis. As such, honeypots are often used for profiling attacker behaviour and to gather information with respect to how attackers operate [31]. In the context of phishing, honeypots are used as (i) sensors for detecting phishing emails and (ii) 'workbenches' to study phishers' tools, behaviors, and social networks [143]. However, it is important to note that the effectiveness of a honeypot is proportional to the number of attacks that go through it.

There are various types of honeypots. A honeypot can be considered as *production* or *research* from a deployment point of view, or classified as *low interaction* or *high interaction* from a technical perspective [143] . A production honeypot set up within an organisation aims to detect and analyse attacks which either randomly or intentionally target the organisation, and provides the organisation with necessary information to strengthen its security. Conversely, research honeypots are devoted to attract attacks in the wild and study attackers in general, and inform security communities of the *status quo* tools and methodologies used by the attackers.

Low interaction honeypots deceive attackers using emulation of systems and services (e.g., a program that records and responds to TCP SYN packets), and tend to provide only

[80]By the very nature of a honeypot, the chance that a honeypot captures the activities of a legitimate user is very small.

general information about attacks and attackers. Examples of low interaction honeypots include Honeyd[81], HoneyC[82], Specter[83] and BackOfficer Friendly[84]. For instance, Honeyd, which has been deployed to detect spammers and worms, collects two types of data: (i) the data stored by the service emulation scripts (e.g., login/password combinations, web page requests, and backdoor connection attempts), and (ii) packet sniffer data [143]. While a low interaction honeypot places a low risk on the deployer, its effectiveness is limited because a successful compromise can never take place [143] (thus 'after-compromise' actions of attackers can never be explored). In the context of spear phishing, a low interaction honeypot would record the malicious payload of a spear phishing email which is then analysed by other offline tools.

Different from low interaction honeypots, high interaction honeypots use real systems (no emulators) and thus can gather and provide much more accurate and detailed information about the attacks and attackers. This benefit of high interaction honeypots comes with a concomitant risk: the attackers may use the honeypots to attack other systems, or other systems may be compromised by unexpected behaviours of the honeypots. Examples of high interaction honeypots include: Symantec Decoy Server [85] (which is previously known as Symantec ManTrap), HoneyBow[86] and the Honeynet Project's Honeywall[87]. According to [143], high interaction honeypots have been successfully used to collect unknown exploits, analyze the social interactions of intruders, and to identify mechanisms for phishing site deployment.

A relatively recent class of honeypot is a *client-side honeypot*. Aimed at collecting information about client-side attacks, a client-side honeypot crawls the Web in an automated manner, and closely monitors itself to detect if there are any changes in the host machine caused by drive-by-download. A client-side honeypot usually visits multiple pages of a website in parallel, once malicious system modifications are observed, it revisits each page individually to determine those with malicious content [156]. Examples of client-side honeypots are Capture-HPC[88], PhoneyC[89] and the Trigona Honey-Client[90].

A type of honeypot particularly relevant to phishing is known as an *email honeypot*. To hide his/her identity, a sophisticated attacker does not directly interact with the mail server. Rather, the attacker indirectly communicates with the mail server via a number of open relays[91] or open proxies[92]. To this end, an email honeypot can be implemented in either of two ways: (i) setting up honeypots as open relays (c.f., Jackpot[93]) or (ii)

[81] Honeyd. `http://www.honeyd.org/`

[82] HoneyC. `https://projects.honeynet.org/honeyc`.

[83] Specter . `http://www.specter.ch/`.

[84] Back Officer Friendly (BOF). `http://www.nfr.net/products/bof/index.html`.

[85] Symantec Decoy Server. `http://www.symantec.com/press/2003/n030623b.html`.

[86] HoneyBow. `http://sourceforge.net/projects/honeybow/`.

[87] Honeywall. `https://projects.honeynet.org/honeywall`.

[88] Capture-HPC. `https://projects.honeynet.org/capture-hpc`.

[89] PhoneyC. `http://www.honeynet.org/project/PhoneyC`.

[90] Trigona Honey-Client. `http://honeynet.org.au/?q=node/63`.

[91] An open relay is an MTA that is configured to send email on behalf of any third party. Open relays are usually used by phishers to conceal their identities.

[92] (Open) mail relays use only SMTP and store the complete message before forwarding it, whereas proxies may not be bound to any specific protocol and the message passing is done in real time.

[93] Jackpot Computer Services. `http://jackpot.uk.net/`.

configuring honeypots as open proxies (c.f., ProxiPot[94]) [143]. Honeyd can be configured to simulate both open proxies and open relays (c.f., [229, 228]). To bait phishers, email honeytokens (i.e., bogus email accounts) can be posted in public websites with the purpose of being harvested by the phishers. Information from emails either passed through the open-relay/open-proxy honeypots or sent to the bogus email accounts would be gathered to provide some insight into the attacks and attackers. Birk also implemented a honeypot system for the purpose of luring, trapping, analyzing and profiling phishers, more information about the system can be obtained at [25].

15.1.1 WOMBAT

WOMBAT[95] is a honeypot project designed with attribution as its specific goal. In particular, it aims to attribute large-scale/sophisticated attacks. Such attacks tend to leverage zombie-nets and/or rogue networks and are usually launched by criminal organisations, dissident groups, rogue corporations, and profit-oriented/underground organisations. WOMBAT is devoted to identify the root causes of problems or events (*root cause analysis*) by attempting to identify the organisations or machines behind attacks and methods used by the attacker. Preliminary efforts and results achieved by WOMBAT are presented below.

- WOMBAT uses a multi-dimensional knowledge discovery and data mining method based on CLIQUE (see Section 7.5.6) to assess the prevalence of malicious activities on the Internet and to get insights into the *modus operandi* of new attack phenomena (c.f., [288]), which can be combined with a fuzzy inference system [289] to attribute large scale attacks (e.g., determine if attack events belong to the same global attack phenomenon, or to the same army of zombie machines (c.f., [68]) or the same root cause.

- WOMBAT studied an active underground economy through an investigation of keyloggers' stealing of credentials via *drop zones*[96] (c.f., [128]). In particular, the researchers utilised various types of honeypots and decoy tools: *spamtraps* were used to collect unsolicited emails and open email attachments to trigger the infection process of the embedded malware, the links in the emails were visited by *client-side honeypots* to determine if the URLs were malicious and trying to install malware in the honeypot, *CWSandbox*[97] was used to further examine and analyse a sample from the client-side honeypots (provided that the honeypot was infected), and *SimUser* (a tool developed by the researchers) was used to simulate the behaviours of a victim using *honeytokens* as credentials.

- Based on the traces left on the deployed honeypots, Pham and Dacier [247] counted the number of various zombie-nets together with their size and their lifetime using similarity measures.

[94] Bubblegum Proxypot. `http://wn.com/Bubblegum_Proxypot`.

[95] Worldwide Observatory of Malicious Behaviors and Attack Threats (WOMBAT). `http://www.wombat-project.eu/`.

[96] Drop zones refer to anonymous collection points where stolen credentials are transferred to for later usage by phishers.

[97] CWSandbox. `http://www.cwsandbox.org/`.

- Stone-Gross and colleagues [284] developed a novel system *FIRE* that identifies Internet companies and organisations responsible for rogue networks, the results of which can be used to pinpoint, and to track the activities of, these rogue organisations. FIRE isolates networks that are consistently associated with malicious activities by actively monitoring zombie-net communication channels, drive-by-download servers, and phishing websites. Zombie-net communication channels were identified through an examination of data collected by Anubis — a tool that executes, and then analyses, Windows-based malware binaries in a virtual environment. Information about drive-by-download servers are obtained via: (i) utilisation of Wepawet (a system that checks user-submitted web pages for malicious Javascript), (ii) a daily compilation of URLs found in unsolicited emails, and (iii) an analysis of a daily-updated list of 'spamvertised' URLs provided by Spamcop[98]. Finally, phishing hosting providers were located by making use of a daily list of URLs of phishing pages provided by PhishTank[99].

15.1.2 Bowen's decoy system

A recent research effort that studied honeypots with a focus on the deception aspect was carried out by Bowen [31]. Bowen introduced a novel, large-scale automated system for generating, distributing and monitoring decoys for both network- and host-based decoy systems. The network-based decoy system is aimed at defending against eavesdropping (network taps), while a host-based decoy system is intended to defense against exfiltration of sensitive information carried out either by malware or insiders. In order to guide the design and implementation of decoy items as well maximise the induced deception, Bowen defines a set of properties for a decoy item, namely *believable*, *inciting*, *conspicuous*, *detectable*, *variability*, *non-interference*, *differentiable*, *expiration*, and *cost*. To maximise detection capabilities, decoy items in Bowen's system were implemented based on a combination of complementary/overlapping techniques, for instance, a watermark is embedded in the binary format of a document file for the purpose of detection, and a beacon is embedded in a decoy document that signals a remote web site upon opening of the document. To maximise the induced deception, Bowen proposed a notion of *perfectly believable decoys* (decoys that are completely indistinguishable from the genuine). Bowen realised this notion by embedding markings in both decoy and genuine documents. The decoy documents are marked with a keyed cryptographic hash function and the non-decoys are marked with indistinguishable randomness. The task of distinguishing these two types of marking is known as computationally infeasible [31] and thus it is almost impossible for the adversaries to discriminate between the decoy and the genuine documents.

15.2 Shadow in the Cloud

A project closely related to the WOMBAT project is *Shadow in the Cloud* [33] (together with *Tracking the Ghostnet* [73] as its predecessor). Efforts in Shadow in the Cloud are directed at malware-based attacks (including spear phishing) with the focus on gaining a

[98] Spamcop.net. `http://www.spamcop.net/`.
[99] PhishingTank. `http://www.phishtank.com/`.

comprehensive understanding of the malware networks utilised by criminal organisations or state nations behind such attacks. While WOMBAT leverages various types of honeypots and decoy systems in its study, Shadow in the Cloud carried out its investigation on real compromised systems, more specifically, the Tibetan computing systems (which were the victim of alleged Chinese cyberspace espionage against the Tibetan community [33]). Shadow in the Cloud utilised a nexus of methods and techniques, and performed information fusion to combine the results obtained. Examples of methods/techniques used in the project include *field investigation* which aims to identify portions of command and control infrastructure used by the attackers as well as to document the type of data that the attackers exfiltrated from the targets, and *technical investigation* which comprised several interrelated components such as:

- *DNS sinkholing*: registration of expired domain names previously used as command and control servers in cyber espionage attacks in order to observe incoming connections from still-compromised computers and learn about various attack methods utilised by the adversaries as well as the nature of the victims,
- *malware analysis*: analysis of malware samples from a variety of attacks in order to determine the exploits the attackers used, the themes used to lure targets into executing the malware, as well as the command and control servers used by the attackers,
- *command and control server topography*: construction of a map of the command and control infrastructure of the attackers by linking information from the sinkhole, the field investigations and the malware analysis,
- *victim identification*: identification of the victims who had been compromised through analysis of sinkhole server connections and recovery of exfiltrated documents (see below), and
- *data recovery*: retrieval of documents that had been sent to drop zones from victim systems and stolen by the attackers.

Regarding the attribution results, even though the combination of the targeted victims, the nature of the stolen documents and the known political interest of Peoples Republic of China (PRC) suggested that the espionage attack on the Tibetan network was under the direction of agents of China [33], the investigators concluded that the evidence was not sufficient to make that assertion.

15.3 Discussion

Making use of honeypots is by all means a prerequisite in any large-scale attribution effort. Implementing an effective honeypot system takes time and resources, but it is well worth the effort if attribution is taken seriously by an organisation. A honeypot system, if properly implemented, allows researchers to observe and analyse technical and social details of 'in the wild' computer attacks and enables an organisation to come up with broader and deeper answers to the 'what' and 'how', and subsequently the 'who', questions relating to the attacks targeting the organisation. Particularly, when investigating criminal

organisations and nation states behind advanced and large-scaled spear phishing attacks, honeypots facilitate attribution with results beyond what can otherwise be achieved (e.g., details regarding the malware network leveraged by the attacker and the command and control mechanism utilised) through static and individual analysis of each component of the attack. To this end, much experience and many lessons can be learnt from existing honeypot projects. Attribution practitioners can also capitalise on existing tools that support a diverse range of tasks necessary for apprehension of threat actors using honeypot systems.

16 Discussion of analysis and attribution of spear phishing evidence

Sections 10, 11, 12, 13 and 14 presented methods that may be applied to analyse and attribute pieces of evidence from a spear phishing attack. In the context of conventional crime where the target of attribution is the person who committed a crime, the presented methods constitute the first line of attribution: identification, analysis and comparison of pieces of evidence left at the crime scenes by the criminal in order to establish any possible linkage between the evidence and the criminal, as well as to provide a solid foundation for any necessary inference about him/her. In the context of cyber crime, and spear phishing in particular, the target of attribution greatly varies in practice (see Section 2.4) due to the significant difficulty associated with achieving the conventional goal of attribution. For this reason, not only do the mentioned methods play important roles in an attribution task, but they can, in many cases, also serve as final attribution solutions depending on the specific goal and the available resource/data an organisation has at its disposal. For instance, if the organisation has a collection of malicious emails as the entire dataset and is interested in determining who among all the suspects are likely responsible for a spear phishing email in question, then the application of one or more methods presented in the lure component would fulfill the goal. Likewise, if the organisation possesses a collection of malware samples and is interested in a similar attribution goal, then the use of methods detailed in the hook (malware) component would be (likely) sufficient. In the presented situations and the alikes, the mentioned sections cover methods to assist an organisation in carrying out its attribution task.

However, in the case that an organisation wishes to perform a more thorough attribution — the organisation is interested in, for example, (i) making use of a wider range of data to gain a more complete view about the attacker, ii) acquiring information about the ultimate spear phisher (rather than his accomplices) of an attack, or (iii) characterising the offender in such a way that it can assist law enforcement investigators to identify and apprehend the actual offender — then it is necessary for the organisation to carry out crime scene reconstruction and/or adversary profiling (presented in the Sections 17 and 18, respectively).

17 Reconstruction of spear phishing attacks

Thus far, methods to identify the actual adversary, or the class of adversaries, associated with each component of a spear phishing attack have been presented. It is important to keep in mind that it is not possible, in many cases, to go directly from the evidence offered by each component of a spear phishing attack to characteristics of the adversary; and the adversaries, if successfully identified for each component, do not necessarily reference to the same individual. It is possible that the identified adversaries are merely those who intentionally/unintentionally took part in the overall spear phishing scheme, and none of them are directly responsible for mounting the spear phishing attack. For instance, the stepping stone machines used in an attack may be owned by innocent parties, the email text may be copied/plagiarised from that of another person without his/her consent, the web pages may have been automatically constructed by a widely available phishing toolkit, the malicious code may be a product of a malware generation tool, the rogue/zombie networks leveraged in the attack may be hired from another party, and finally the individual that collected the stolen items may be only a *mule*[100].

If it is not possible to relate pieces of crime scene evidence to the behaviours and characteristics of the ultimate spear phisher, then additional inferences based on the pieces of evidence is necessary. These additional inferences are collectively referred to as *reconstruction of a spear phishing attack*.

Crime (scene) reconstruction[101] of conventional crime involves (i) inferring offender behaviours associated with the crime scenes (*crime scene behaviours*) and (ii) interpreting the pieces of crime scene evidence in a larger context, from which the crime is reconstructed in the form of a narrative and timeline (c.f., [55, 256]). In essence, a crime reconstruction involves forming logical conclusions based on information at the scene, or from subsequent analysis and attribution conducted with the evidence. A crime reconstruction also provides an opportunity to incorporate nonphysical evidence that may be useful for predicting adversary characteristics but not subjected to forensic scientific analysis such as intelligence information [121]. There are indeed a number of crime scene reconstruction activities already performed in the attribution of each component (e.g., malware behaviour, data exfiltration mechanism and email traversal). Through a reconstruction of events from the available evidence, the investigators now have a set of relevant information which includes physical/digital evidence (e.g., a writeprint computed from the email content, a thumbprint calculated for the embedded malware, the list of deception techniques utilised in the attack), crime scene behaviours (e.g., the phisher harvesting information about the victim before launching the attack, the malware communicating with a remote server on its successful installation), and a sequence of events (e.g., the installed malware communicating with its remote master before receiving commands from the master, and then executing the commands on the compromised machine). The results of the reconstruction of a spear phishing attack can then be used to advance predictions about adversary characteristics (see Section 18) which cannot be made on the basis of the evidence itself [121].

[100] Mules refer to the individuals who are employed/tricked by phishers to access and transfer the stolen items on behalf of the phishers.

[101] Crime scene reconstruction is also referred to as *crime reconstruction* in the literature of offender profiling.

Perhaps, another merit of crime scene reconstruction is that it eliminates non-relevant pieces of evidence from further investigation. In any type of crime, the space of evidence is potentially very large. To this end, a crime scene reconstruction brings focus to a subset of evidence and inferred crime scene behaviours, that are relevant with respect to the specific goal of profiling/attribution desired by an organisation.

Crime scene reconstruction is, in its strict sense[102], still an art. There currently does not exist a thematic method that can be used as a template. Nevertheless, attribution practitioners can learn from similar tasks carried out for each component of a spear phishing attack, capitalise on the experience accumulated through specific cases in traditional investigation, and utilise domain and expert knowledge if reconstruction is desired to be performed over the entire spear phishing attack. Fortunately in this respect, reconstruction of a spear phishing attack is generally simpler than of a conventional crime. This is due to the fact that crime scene construction in conventional crime has to be carried out in a post-incident manner (i.e., reconstructing how an attack might have been carried out based on the evidence collected *after* the attack was accomplished), while such tasks in the spear phishing context are mostly performed *mid-incident* (i.e., on receiving a spear phishing email prior to the actual execution of the attack). In addition, the chronological steps of an overall spear phishing attack is also standardised (i.e., Target, Lure, Hook and Catch). For this reason, a much larger set of evidence is often available to the attribution practitioners, and the reconstruction of a spear phishing attack is likely to be more straight-forward.

It should be noted that a spear phishing attack may be associated with other cyber crimes, either those performed to assist in the spear phishing attack (e.g., compromising stepping stone machines or web mail accounts), or those for which the spear phishing attack is intended to assist (e.g., stealing security information for subsequent crimes). This presents a broader crime scene or scenes that may be reconstructed, for the purposes of understanding the attack, the attacker as well as their intent, goals, identity or grouping. Discussion of the broad area of cyber crime reconstruction is outside the scope of this survey.

[102]The reader should not confuse crime scene reconstruction as defined in this survey with other definitions of crime scene reconstruction in the literature which also include behaviour profiling as presented in Section 18.

18 Adversary profiling

In the context of this survey, a profile refers to an artifact that describes information of interest about an adversary. The information contained in a profile is naturally subject to the goal and the nature of the specific attribution problem for which the profile is constructed. For this reason, having a well-defined goal and a clear understanding of the problem are primary requirements in building such a profile. The reader should have already observed various forms of adversary profile throughout the discussions of attribution methods in Sections 10, 11, 12, 13 and 14. For instance, with respect to email authorship, an author profile is one that described information about the author's writing style; with respect to determination of an adversary class, an adversary profile would be one that contained information about the attack patterns. Many of the discussed profiles share one or both of the two following common features: the profiles were constructed for a particular crime scene (a component) of a spear phishing attack, and/or they characterise the adversaries in terms of their 'crime scene characteristics' rather than the characteristics of the adversaries themselves.

To reduce the risk of attributing an attack to an innocent party or an accomplice of the attacker, it may be beneficial to perform attribution across crime scenes. To this end, an adversary profile is likely to contain information pertaining to more than one crime scene (McCombie and colleagues have indeed investigated this kind of profile by considering features of both the email and phishing website of an attack into the profile — see Sections 11.3 and 13.3). Provided that an organisation possesses evidence (as well as their forensic results) from multiple crime scenes (e.g., Target, Lure, Hook and Catch) and wishes to use them to obtain a more complete view about the ultimate attacker, the profiling methods presented in this survey can be reused and adapted for this purpose — *albeit* with care. Simply incorporating findings for all the evidence across multiple crime scenes into a profile is not likely to produce a 'workable solution' [231]. This is because different sets of evidence (together with their findings) usually come from different investigative perspectives, and a simple combination of the sets of evidence would lose the necessary focus and potentially be counter-productive. There are perhaps two possible options to address this problem. In the first option, crime scene reconstruction can be utilised to describe the crime from the perspective of the attribution goal through a retention of relevant pieces of evidence and an elimination of non-relevant information (e.g., if an attribution goal is to determine whether or not the adversary is likely a nation state, then pieces of evidence such as the custom email tool used in the attack would be less significant). In the second option, an adversary profile should be described in terms of a higher level of abstraction; for example, rather than having a profile containing a collection of low-level metadata (e.g., information relating to XMail, date and body length associated with an email), one can enable the profile to describe an adversary in terms of his/her modus operandi, or with respect to his/her characteristics as an adversary (e.g., skill-level, finance, resource, and capability).

As mentioned above, the majority of profiling methods in the literature aim to characterise the adversaries in terms of their crime scene characteristics (rather than the adversary characteristics) and categorise classes of adversaries in terms of their attack patterns or signatures. These methods are primarily designed to aid attribution practitioners in determining if an attack is launched by one of those responsible for previous attacks, or assist an organisation in strengthening and enhancing its security posture against adver-

saries. However, the methods are not of great help when it comes to planning offensive approaches (instead of defense mechanisms), collecting intelligence, or assisting law enforcement investigators in identifying and apprehending the attackers. To achieve these goals, defining an adversary in terms of his/her crime scene characteristics is not sufficient. It is necessary to predict characteristics *of* the adversary in order to help agencies match the described adversary with suspects in real life (who may not have any record of previous attacks). The predicted characteristics of the adversary also assists investigators in narrowing down the list of real-life suspects, the results of which can then be combined with other intelligence information or traditional investigation findings in order to determine, with some degree of certainty, the actual offender. Attempts to capture characteristics of the adversaries include: (i) the research work in email authorship attribution which uses the writing style of an author to infer information about the author such as background, gender, age and psychometric traits (see Section 11.2.2), and (ii) the research work in adversary characterisation based on malware which predicts information about the adversary associated with a malware-based attack such as capability, skill-level and connections. However, these efforts are limited both in breadth (the methods are specifically studied in the context of email and malware) and in depth (the methods are restrictively based on evidence pertaining to linguistics, attack tools and attack techniques).

The rest of this section is primarily concerned with profiling methods aimed at capturing adversary characteristics in a more generic sense that can be applied to a crime scene, or across crime scenes, and can be applied to different sources of evidence. More specifically, the profiling methods to be discussed are categorised into two groups: characterisation of conventional offenders (a major topic in offender profiling), and characterisation of cyber adversaries.

18.1 Characterisation of conventional offenders

Offender profiling enjoys a long history of attempts to relate pieces of crime scene evidence to offender characteristics — the processes which essentially involve predicting the characteristics that lead an offender to leave the pieces of evidence on the crime scenes. Despite its rich literature, the field lacks a large-scale empirical examination regarding the connection between offender characteristics and the particular pieces or patterns of evidence, which is necessary for this kind of prediction to become sufficiently reliable [121]. However, much foundational knowledge and many individual findings can be learnt from the accumulated work in the research field. A brief overview of studies relating this topic is presented, which is loosely based on work by Hicks and Sales [121].

Research efforts to characterise an offender based on crime scene evidence in the literature of offender profiling are centred around studies of the relationship among the offender's behaviours, motive and personality. It is perhaps important to clarify the meanings of the mentioned characteristics in the context of offender profiling, before any further discussion is to be made.

18.1.1 Motive

Motive is concerned with *why* an offender engaged in a criminal act. In the context of offender profiling, motive is closely related with, but should be distinguished from, *intent*[103] which indicates whether an offender purposefully committed an act. Therefore, motive requires the presence of intent [121]. Since criminal profiling is based on the presumption that crime is committed purposefully, it is motive, not intent, that has received considerable attention as a major characteristic of an offender. A motive in itself can be either *instrumental* or *expressive* [121]. Instrumental motives are those that drive offenders toward achieving a certain goal (e.g., financial gain, political advantage, or retaliation). Expressive motives are, in contrast, concerned with expressing one's point of view or one's emotions (e.g., propagandising a political/social point of view or demonstrating a disagreement).

18.1.2 Personality

The topic of personality is perhaps most extensively studied in psychology research. There are two paradigms of psychology studies which place different emphases on the influence of internal and external factors in one's personality. The first paradigm associates personality to the person: individuals possess internal personality traits that are expected to be stable across time, situations and environments. Researchers with this point of view devoted effort to identify and categorise important personality traits of human beings. For example, Cattell ([45, 46] cited in [121]) divided traits into three categories: *dynamic traits* (those that put people in motion to accomplish goals), *ability traits* (those that determine one's effectiveness in achieving goals) , and *temperament traits* (those related to the speed, energy, and emotional reactivity of people's responses). The second paradigm states that personality is shaped and influenced by the situation. In this view, an individual's personality is specific to an individual situation, and varies according to situations. This theory was supported by subsequent reports on the influence of gender, ecological settings, race and social class, and culture and history on one's personality.

Despite various important findings relating to the impact of situations on personality characteristics, studies of personality in the realm of offender profiling still largely adopt the view of personality as a human trait and neglect situations in analysing personality characteristics of an offender [121].

18.1.3 Behaviour

There are two types of behaviours about an offender that an investigator can 'guess' from crime scene evidence: *inferred behaviours* and *predicted behaviours*. The inferred behaviours refer to the kind of behaviours related to the crime scenes, also known as *crime scene behaviours*. Examples of this type of behaviour are 'the offender relayed a spear phishing email via zombie computers' or 'the offender instructed the malware to contact the command and control server upon successful installation'. Inferred behaviors

[103]Please note that *intent* adopts a different meaning in cyberspace: it often refers to the objective of an adversary.

are also what constitute the modus operandi (which consists of the behaviours necessary to the successful commission of a crime) and signature (which comprises those behaviors that fulfill some type of deeper psychological need within the offender [121]) of an attack. Predicted behaviours are concerned with general behaviours of the offender (or *offender behaviours*) such as 'the offender tends to target high-profile victims' or 'the offender invests a great deal of resources in his/her attack'. These two types of behaviours form a bidirectional relationship with other characteristics of the offender such as *motive* and *personality*. Inferred behaviours have been presented as part of crime scene evaluation and reconstruction, this section is mainly concerned with predicted behaviours of an offender.

According to Hicks and Sales [121], offender behaviours are most useful in identifying and apprehending an unknown perpetrator. Their merits are due to the fact that behaviours are observable, tangible, and more easily described and comprehended than *latent* characteristics such as motive and personality. However, motive and personality characteristics are incontrovertibly important to an understanding of offenders and their offenses. Therefore, it is important to have a way to access these two latent offender characteristics. In convention criminal investigation, personalities and motives may be revealed via projective testing[104], self-reports[105], or examinations of their manifestation in behaviours.

In the context of spear phishing, not only are the first two methods susceptible to the accuracy of an offender's self-description or interpretation, but they also require some sort of interaction with the human suspects, which is not the case in the context of spear phishing. Inferring offender characteristics in this context is most likely to draw from the third approach in which motive and personality characteristics are presumably reflected in behaviors which can be inferred from crime scene behaviours and/or crime scene evidence. This approach therefore requires the presence of a model which captures the relationship between the mentioned aspects relating to crime scenes and the offender, thereby enabling inferences from one aspect to another.

18.1.4 Model of crime scene evidence, motives, personality and behaviour

The literature of offender profiling has yet to offer a complete scientific method to relate crime scene evidence, behaviour and motive, and personality. Instead, the literature provides a large collection of individual studies on different aspects of this relationship. For instance, relationships between motives and behaviours have been investigated in a series of research work on arsonists (c.f., [43, 260, 157]) and those between personality and behaviours have been studied in sex offenders (c.f., [44, 127, 153, 255]). Though these studies provide important findings, in a broader picture, they only address one part of profiling and the bivariate analyses used in their studies are not sufficient to capture the intricate relationships between crime scene evidence, motives, personality and behaviours. The research work by Canter and colleagues [42, 43, 44] is the only research work among

[104] Projective testing involves triggering a person to respond in a certain way that may reveal his internal motivations and personality characteristics by presenting him/her with ambiguous stimuli and asking him/her to interpret the stimuli in some meaningful way.

[105] A self-report method involves asking a person to answer a questionnaire designed in such a way that it can reveal various personality characteristics.

them that takes into account all these characteristics (motives, personality and behaviour) in analysing crime scene evidence to infer information about the offenders. More specifically, there are three basic types of variables in Canter's profiling model: crime scene actions (corresponding to crime scene behaviours), offender themes (addressing aspects of an offender such as the offender's motives and personality), and offender background characteristics (being predicted from the offender themes). Unfortunately, according to Hicks and Sales [121], Canter's model inevitably inherits the limitations of his overall offender profiling approach (as discussed in Section 8.2).

To address this gap in the literature with respect to a comprehensive representation of the multivariate relationship between crime scenes and offender characteristics, Hicks and Sales [121] proposed a more generic model of profiling which captures not only the types of bivariate relationship that have been studied, but also the possible intricate relationships that have been unaccounted for in the literature. The simplest form of the model (which is graphically depicted in Figure 9 [121]) consists of the following three tiers:

- *crime scene evidence and first-level offender behaviours*[106] which involves drawing inferences about behaviours that are directly related to crime scene evidence,
- *motive, personality, and first-level offender behaviours* which involves making predictions about motives and personality characteristics using first-level offender behaviours, and
- *motive, personality, and second-level offender behaviors* which involves moving away from crime scene evidence and predicting the second-level offender behaviours (behaviours that are not necessarily related to the commission of the crime, but are instead investigation-relevant behaviors predicted from the model variables)[107].

The goal of the presented model is to identify offender behaviours relevant to investigations, subsequently assisting law enforcement in identifying the correct perpetrator. Therefore, each branch of the model terminates in a second-level behavior, a collection of which constitute an offender's behaviour profile that law enforcement can use in an investigation. The model can be instantiated into different forms, depending on the type of crime, the nature and quality of crime scene evidence, and the reliability and validity of particular relationships between the model's variables [121]. For the purpose of illustration, Figure 10 is included to represent a fraction of such a model when applied to a spear phishing case[108]. It is important to note that this is only a simplified example and that what would be dealt with in practice is likely much more complex (see Figure 11 for an example of such a model).

Toward its applications in practice, the described model of profiling requires scientific findings to form actual links among the variables within the model. To this end, one can capitalise on individual attribution/forensics methods investigated in the literature applied to specific relationships in order to derive such linkings. For example, techniques

[106]First-level offender behaviours correspond to the aforementioned crime scene behaviours.

[107]The model can continue branching outward beyond tier 3 to predict additional second-level offender behaviours if the situation demands.

[108]Please note that the predictors and relationships used in the examples are not empirically verified with respect to their validity and reliability.

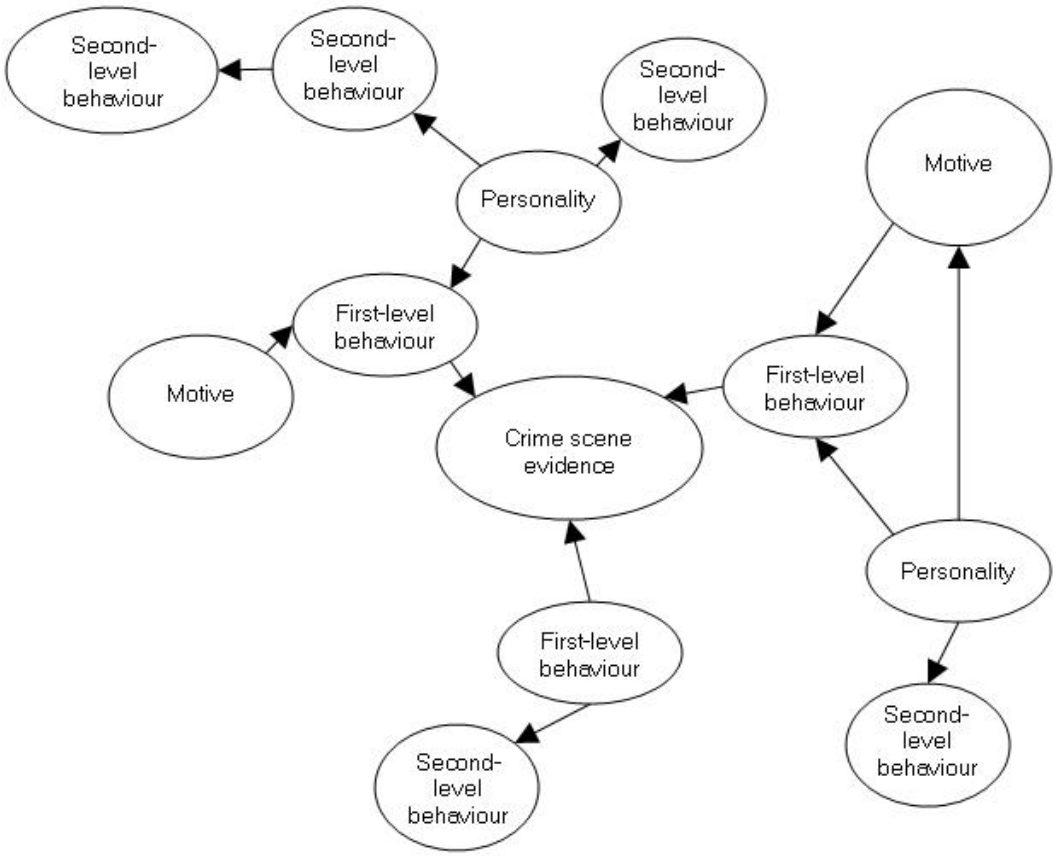


Figure 9: *Basic structure of a scientific model of profiling (adapted from [121]).*

in the paradigm of malware analysis can be leveraged to infer from the 'malware evidence' that 'the malware exploits unpatched vulnerabilities'.

18.2 Characterisation of cyber adversaries

Offender characterisation/profiling methods for conventional crime are mostly concerned with individual offenders. Thus far, an important type of offender in cyberspace — *groups* or *organisations*— has been neglected in this regard. Although the conceptual scientific profiling model presented above possesses sufficient generality for it to be used in characterising this collective type of offender, the model does not distinguish between the two types of cyber offender, which is necessary to properly emphasise and recognise the prevalent role of groups (e.g., nation states and criminal organisations) in advanced cyber attacks.

It appears that the only published work that comprehensively addresses the problem of cyber offender characterisation that takes into account the presence of groups is the study by Parker and colleagues [231]. More specifically, the researchers proposed a cyber adversary characterisation framework that allows for both a theoretical categorisation of cyber adversaries and a post-incident forensic characterisation. Theoretical characterisation theory is concerned with characterisation of adversaries in a 'prospective manner',

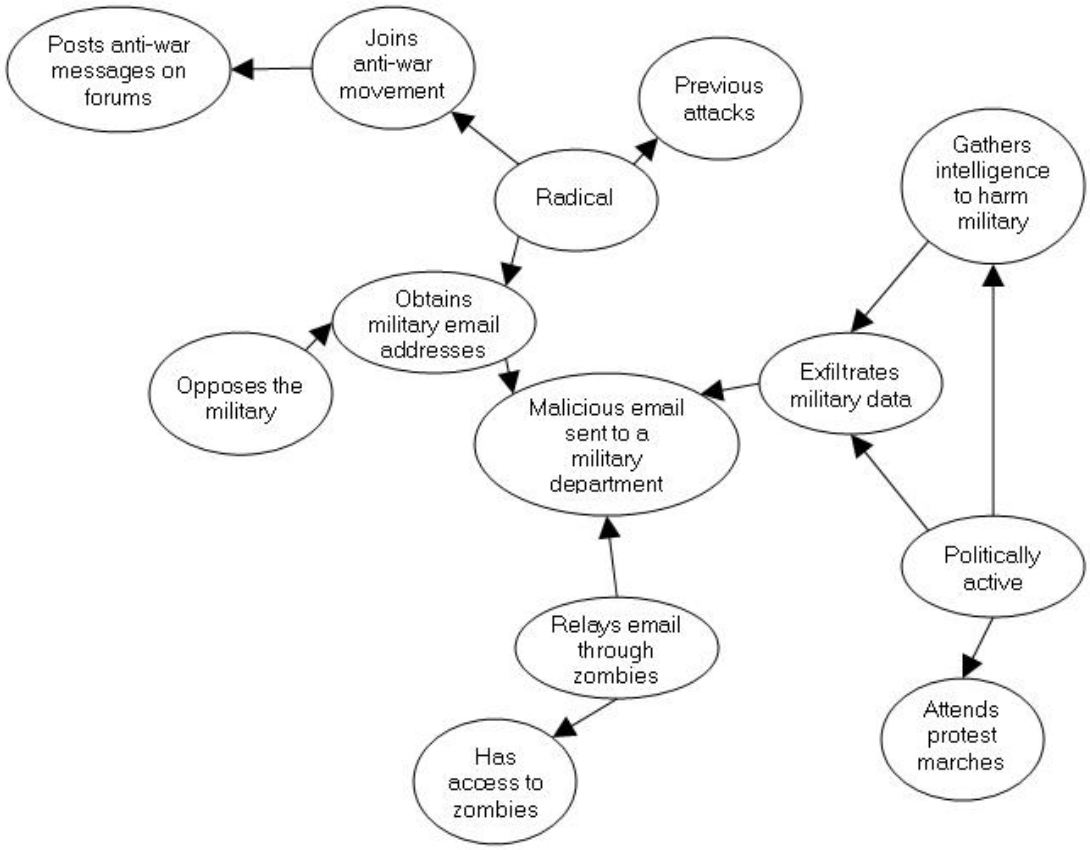


Figure 10: *An illustration of the application of Figure 9 in the context of spear phishing.*

and possibly of most use for characterising possible threats to an organisation, for improving the security posture of the organisation, and for devising methodologies to test the resilience of the organisation's systems against the known and unknown [231]. In contrast, post-incident or forensic characterisations — the type of attribution covered in this survey — involves gaining insight about the 'real' adversary using information available only after an incident has occurred.

The cornerstone of the framework by Parker and colleagues is the *adversary object matrix* (see below) which captures characteristics of cyber adversaries in relation with their *environment* and *attack target* in a *dynamic* and *interactive* manner. An understanding of the adversary object matrix is a prerequisite for grasping the essence of Parker and colleagues' framework.

18.2.1 The adversary object matrix

Instead of depicting an offender as a human individual with certain personalities and behaviours, the adversary object matrix describes cyber adversaries via a set of core *adversarial properties*, namely *Environment Property*, *Attacker Property* and *Target Property* (see Figure 12).

On the one hand, the matrix allows for characterisation of adversaries in terms of

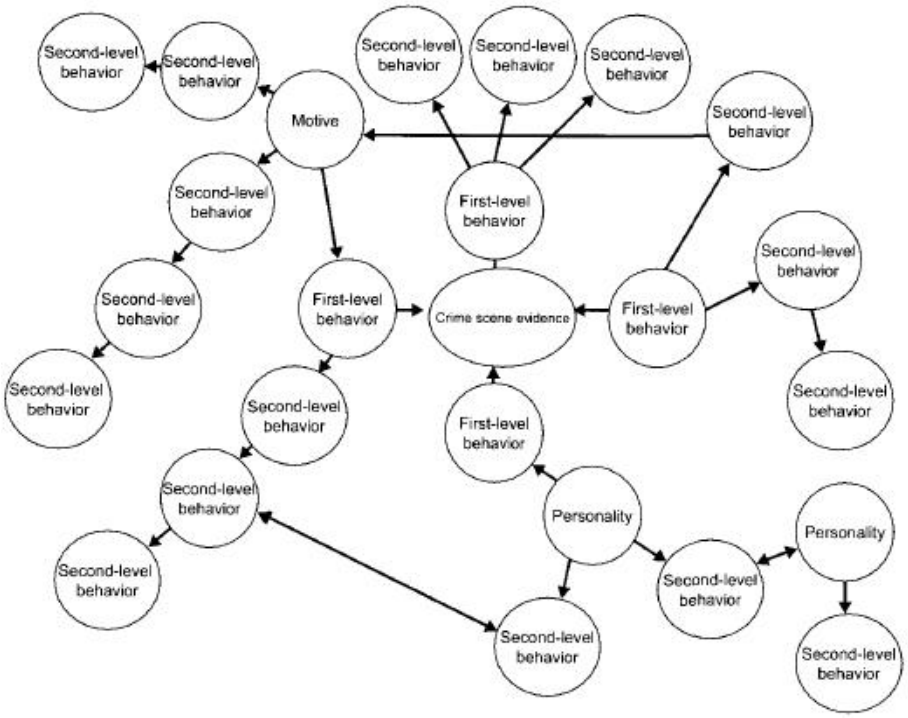


Figure 11: *A more complex structure of a scientific model of profiling [121].*

various aspects contained in the three mentioned properties. In this regard, each property in the matrix contains a list of *objects* associated with the different aspects of the property. For instance, Environment Property contains 'cultural/political impact' and 'intelligence/association' objects, Attacker Property consists of 'motivation' and 'resource' objects, and Target Property includes 'environment' and 'value' objects. Each object in turn comprises a list of *elements* which are members of the object, e.g. , 'intellectual property' and 'internet bandwidth' are elements of the 'value' object, and 'time' and 'finance' are elements of the 'resource' object. On the other hand, the adversary object matrix facilitates a categorisation of the adversaries in a manner which represents the relationship and interactions among the adversarial properties (and their objects). For example, objects in Environment Property directly impact on the capability object of Attacker Property, objects in both Environment Property and Target Property directly impact on the motivation object, and objects within Attacker Property impact an adversary's motivation toward, and his/her capability against, a target.

18.2.1.1 Environment Property Environment Property serves as a good starting point to characterise cyber adversaries. It assists in clarifying the root of an adversarial act, for example, motivation of an adversary, level of sophistication of an attack and the origin of the resources used in an attack. The objects contained in the adversary environment property include the following:

- the *world events/political environments* object captures information relating to the political environment of an adversary (e.g., the political environment in the state

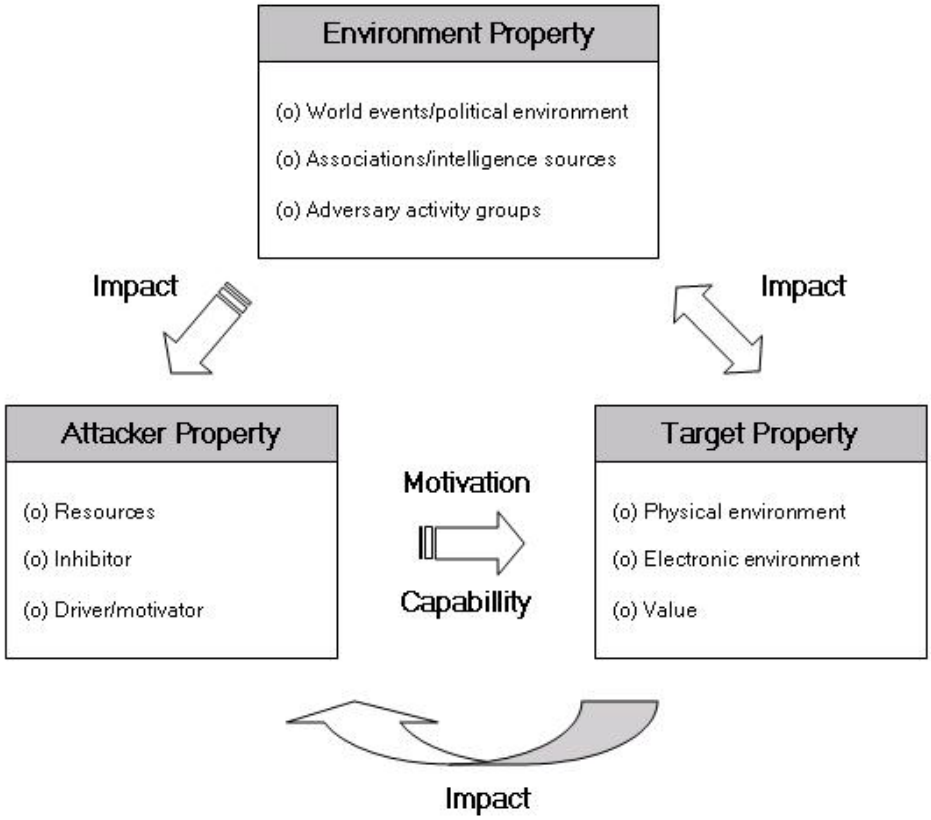


Figure 12: *A graphical illustration of the adversary object matrix (adapted from [231]).*

from which an adversary originates, the wealth of the state where a potential adversary lives, and the cyber laws of the state from where the adversary launches an attack),

- the *association and intelligence source* object holds information regarding the entities with and from whom an adversary is associated and receives support — this assists one in determining the types of resource the adversary has access to, and thus the kinds of attacks one can expect from the adversary,

- the *adversary activity groups* object indicates whether a group takes part in an adversary act[109].

  The group object significantly influences the nature of an adversarial act and on the manner in which the act is conducted. For instance, this object is likely to influence Attacker Property in the following aspects: *attack objective* (e.g., the attack objective will almost always be an objective that benefits the group), *knowledge/skills* (e.g., the (possibly multi-staged) attack often utilised the combined knowledge/skills of multiple members rather than any one individual), *finance* (e.g., the finance is likely

[109]It is important to distinguish groups from associations: members of an association tend to act individually, sometimes without the knowledge of other members in the group, whereas all members in a group are aware of the adversarial act, and usually take part in the attack either directly/indirectly.

to be more readily available than those adversaries without the group object), *time* (e.g., the time invested in preparation of an attack is significantly greater for those without the group object), *initial access* (e.g., an adversarial group may be able to gain initial access through other group members such as insiders), and *attitude to attributes of attack* (e.g., the attitude to the attributes of the attack, such as fear of detection and attribution, will probably be that of the group rather than actual individuals carrying out the attack).

18.2.1.2 Attacker Property Attacker Property contains objects that attempt to describe an adversary's behaviours in a given attack situation. The values contained in the attacker objects are subject to the data passed to Attacker Property from Environment Property. The objects belonging to Attacker Property include the those presented below.

- The *resource object* describes the resources possessed by an attacker. The resource values are relative to given attack attempts and are derived from the associations and intelligence and group objects in Environment Property. Its primary function is to offset the variable values contained in the inhibitor objects and thus increase the likelihood of success. The resource object contains the following elements:
  - the *time element* measures the time that an adversary takes to execute his/her attack against a target. Values of this element are impacted on by the group and political objects in Environment Property.
  - the *skills/knowledge element* is the most significant element of the resource object which both influences the capability against a target in a given attack attempt, and represses various elements within the inhibitor object. Values of this element are impacted on by the group and association objects in Environment Property.
  - the *finance element* measures the liquid finance to which the adversary has access for the purpose of their attack. This is a prerequisite for a cyber adversary to execute a successful attack (e.g., either by opening up a new attack opportunity or providing an elevated level of initial access). Values of this element are impacted on by the group and political/personal environment objects in Environment Property.
  - the *initial access element* indicates the level of initial access an adversary has at the commencement of his/her attack. Having some sort of initial access would likely increase the chance of success of a given attempt.
- The *inhibitor object* and its elements represent the adversaries' attitude to the adverse attributes of an attack against a known target, given an attempt. The primary objects that impact the elements of the inhibitor object are: the objects within Target Property and the resource object as well as the driver/motivator object within Attacker Property. Elements contained in the inhibitor object includes:
  - *impact given success*, e.g., whether the payoff given success is less than the resources required to ensure success;
  - *perceived probability of success given an attempt*, i.e., the likelihood of completion of an objective in the eye of an adversary;

  - *perceived probability of detection given an attempt* — the adversary may offset the probability of detection given an attempt with the use of additional resources;
  - *perceived probability of attribution given detection* — the adversary may use additional resources to ensure the integrity of the attackers identity given detection;
  - *perceived consequences to adversary given detection and attribution* — these values are primarily in proportion to the environmental properties, in relation to the target properties due to the law and politics of the country in which the adversary lives; and
  - *adversary uncertainty given the attack parameters*, e.g., uncertainty in the mind of an adversary resulting from a lack of information regarding any of the elements within the inhibitor object, and therefore a lack of ability to make informed decisions regarding a course of action.

- The *driver/motivator object*, directly impacted by the events/political object in Environment Property, objects in Target Property, and the resource object in Attacker Property, is responsible for describing the adversaries' objective and ultimately, their motivations. It often impacts the attitude toward the attributes of the attack represented by the inhibitor object. Elements within this object include:
  - *impact given success* — the result of this element is considered to be an attack motivator if the payoff is considered by an adversary to be of greater value than the resources consumed in an attack's execution; and
  - *perceived probability of success given an attempt* — the result of this element is considered to be an attack motivator if the perceived probability of success given an attempt meets an adversary's expectation threshold.

18.2.1.3 Target Property Target Property contains objects that attempt to describe the target from the perspective of cyber adversaries, and have direct impact on the inhibitor/driver object of Attacker Property. Target Property contains a number of objects, including physical environment, electronic environment and value.

- The *physical environment* objects cover aspects of the target such as its physical location, its owner, and any defenders that exist within the target's physical environment. These types of objects potentially influence both the inhibitor and driver objects of Attacker Property.
  - For instance, the *target location* object influences perceived consequences of attribution and motivator objects (e.g., the greater the physical separation between adversaries and their target, and/or the weaker the cyber laws in both their countries of residence and foreign states, the lesser the consequences of attribution at times when their attacks are detected, and, of course, the greater motivation. Likewise, the *target owner* object influences (i) perceived consequences of attribution (e.g., specific owners are likely to provoke, to varying degrees, fear of attribution, as well as fear of consequence if attribution occurs),

(ii) perceived probability of detection (e.g., owners have different capabilities to detect a cyber attack as it is occurring), and (iii) motivator objects (e.g., the attacker may be the one that holds a grudge against the owner of the target, or one that has an interest in the resources possessed by a particular owner).

- The *electronic environment* objects consist of everything from computer hardware to operating systems, network operating systems and other technologies. These types of object, again, can influence the adversary's attitude toward an attack. For instance, the technical object influences perceived probability of success, and thus an adversaries' motivation — adversaries will always view the technical properties of a target in terms of whether the technical resources they possess are sufficient to ensure the success of an attack.
- The *value* object consists of the elements of the target that may be considered of value to cyber adversaries, such as intellectual properties, internet bandwidth, and other resources.

To predict possible adversaries who may launch an attack against an organisation, the adversary object matrix, once populated with data and having its elements evaluated and scored, allows the organisation to infer the type of adversaries that are likely to have an interest in, as well as the type of threat posed to, a given organisation's asset. Regarding attribution of a 'real' attack (which has already occurred), results from attribution and forensic investigations (such as those described in Section 12.2.1) as well as other sources of information can be applied to the framework in order to characterise the adversary involved in the commission of the attack.

18.3 Discussion

Naturally, adversary profiling can be performed from multiple perspectives and at different levels of attribution. Three approaches to profiling have been mentioned and/or discussed in this section: (i) profiling as categorisation of adversaries based on crime scene characteristics, which primarily serves to identify classes of adversaries with a common attack signature or modus operandi (corresponding to attribution level IIIa); (ii) conventional offender profiling that aims to infer characteristics of the actual adversary (corresponding to attribution level IIIb) in order to match the adversary with real-life suspects, as well as to assist law enforcement investigators to identify and apprehend the 'real' adversary; and (iii) cyber adversary characterisation that encompasses both singular and collective types of adversaries (corresponding to attribution levels III and IV) and allows for both theoretical and post-incident characterisation of adversaries of cyber attacks. For the first type of profiling, individual profiling methods presented for different components of spear phishing (based on sets of features and machine learning algorithms) can be reused or adapted for use. In the second type of profiling, one can capitalise on the conceptual scientific profiling model proposed by Hicks and Sales (See Section 8.1), and apply attribution methods (many of which presented in Sections 10, 11, 12, 13 and 14) in the literature to establish the actual links among the variables of the model. Regarding the third type of profiling, this profiling framework is primarily devised for theoretical characterisation of adversaries given a type of asset, and thus does not explicitly incorporate crime scene evidence as

part of the model. To this end, the individual attribution methods can be, again, applied to predict values for various elements in Attacker Property and Target Property, based on which other characteristics about the adversary can be inferred, and a more complete view about the adversary can be depicted.

19 Conclusions

Spear phishing has emerged as a considerable threat to the security of nations, organisations and individuals. The ability to successfully attribute those responsible for this kind of attack is highly desired in order to: develop intelligence about adversaries, apprehend, suppress or exploit the adversary, put in place proactive defences appropriate to the adversary, or to act against the adversary for the purpose of retaliation and deterrence.

However significant obstacles have impeded our attribution abilities. Attribution of spear phishing is remarkably difficult as it entails a variety of technical, social and psychological factors. Despite these difficulties, the research conducted in this survey has indicates that attribution of spear phishing attacks can still be worthwhile to pursue because:

- as complex multi-stage multi-step cyber attacks, spear phishing incidents provide many opportunities to uncover digital trails left behind by the attackers;
- as attribution of spear phishing can be carried out at different levels, attribution results achieved at any level may be valuable to investigators;
- ultimately two-way communication is required in a spear phishing attack, despite the many ways to hide each communication path, they do not always prevent attribution; and
- though many individual attribution methods are likely to be subverted through deception, deploying a variety of methods with appropriate and careful combination is likely to yield more reliable results (e.g., attribution results can be verified by means of redundant methods and strengthened through the use of complementary approaches).

Currently, there exist very few, if any, attribution methods directly addressing spear phishing in the literature, hence investigators must reuse or adapt thematic methods in related domains. To this end, the methods presented in the survey have their roots in divergent research disciplines including authorship attribution, offender profiling, software authorship, malware classification and IP traceback. Various aspects of the attribution problem of spear phishing together with the relevant attribution methods discussed in the survey are summarised below.

- In order to approach the attribution task, the conceptual scientific model for offender profiling was leveraged for use as the attribution methodology. More specifically, each component of a spear phishing attack is treated as a crime scene, and data pertaining to each of the components is categorised into different types of evidence (i.e., ownership, individual characteristic and class characteristic). The attribution problem was addressed for each crime scene and each type of evidence.
  - Regarding attribution of the target component, general recommendations regarding how to infer information about the adversary through the remnant of the component were provided.

  - With respect to attribution of the lure component, the reviewed attribution methods include (i) traceback techniques to identify the source of a spear phishing email, (ii) email authorship attribution methods to determine and characterise the email's author, and (iii) various learning/classification approaches based on email metadata and their alikes to identify classes of adversaries.
  - In terms of attribution of the hook component, the discussed methods include (i) software authorship methods to compare a piece of malware to those of the suspects, (ii) approaches to characterise the adversaries based on malware characteristics, (iii) malware analysis and classification techniques to categorise classes of adversaries, and (iv) various methods to determine the origins of phishing websites as well as group of adversaries associated with the websites.
  - Regarding the catch component, active tracing methods which make use of web bugs, watermarks and honeytokens were presented.

- In addition to reactive attribution methods, potentially much more information about the attack can be obtained via proactive/active attribution approaches. In this regard, large scale realisation of proactive/active approaches by means of honeypots, decoy systems and field investigations were described.

- Toward characterisation of the ultimate adversary, existing conventional offender profiling and cyber adversary characterisation methods were presented that can be leveraged and adapted for use in the spear phishing context.

It is important to note that attribution results delivered by any of the methods discussed in the survey should be treated as *suggestive* rather than *conclusive*. Currently, due to various factors, it is not feasible to reliably pinpoint the exact attacker behind any single spear phishing attack using technical methods alone. For this reason, extant attribution methods should be applied either to characterise the attacker or to gain information about the attacker in order to narrow down the group of suspects.

The application of attribution of spear phishing attacks in practice remains dependent on the goal and resources of the nation, organisation or individual concerned. If intelligence about an adversary is sought, attribution carried out at any of the levels would be useful; if proactive defences are to be put in place, utilisation of methods that identify adversary classes may be adequate; if the adversary is to be apprehended, suppressed, exploited, retaliated against or deterred, then a thorough analysis of a complete attack would be mandatory.

In this context, this report offers a comprehensive survey of current methods and techniques of spear phishing attribution. While the final outcome of painting a detailed picture of adversaries is still a future prospect, it is expected that an appropriate and careful utilisation of the presented methods would allow one to discern the silhouette of the shadowy adversary in light of attribution evidence.

References

1. Abbasi, A., and Chen, H. 2005, Applying authorship analysis to extremist-group Web forum messages. *IEEE Intelligent Systems*, Vol. 20, No. 5, pp. 67–75.

2. Abbasi, A. and Chen, H. 2008. Writeprints: A stylometric approach to identity-level identification and similarity detection. *ACM Transactions on Information Systems*, Vol. 26, No. 2.

3. Amin, R. M. 2002. Detecting targeted malicious email through supervised classification of persistent threat and recipient oriented features. PhD Thesis, The George Washington University, USA.

4. Anderson, A., Corney, M., de Vel, O. and Mohay, G. 2001. Multi-topic E-mail authorship attribution forensics. In Proceedings Workshop on Data Mining for Security Applications, 8th ACM Conference on Computer Security (CCS).

5. Argamon, S. 2008, Interpreting Burrowss Delta: Geometric and probabilistic foundations. *Literary and Linguistic Computing*, in press.

6. Argamon, S., Koppel, M. and Avneri, G. 1998, Style-based text categorization: What newspaper am I reading? In Proceedings of AAAI Workshop on Learning for Text Categorization, pp. 1–4.

7. Argamon, S., Koppel , M., Fine, J. and Shimoni, A. 2003, Gender, Genre, and Writing Style in Formal Written Texts. *Text*, Vol. 23, No. 3.

8. Argamon, S., Sari, M. and Stein, S.S. 2003. Style mining of electronic messages for multiple authorship discrimination: First results. Proceedings of the 9th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, ACM Press, pp. 475–480.

9. Argamon, S., Whitelaw, C., Chase, P., Hota, S.R., Garg, N. and Levitan, S. 2007. Stylistic text classification using functional lexical features. *Journal of the American Society for Information Science and Technology*, Vol. 58, No. 6, pp. 802–822.

10. Armin, J., McQuaid, J. and Jonkman, M. 2008. Atrivo - Cyber Crime USA. `http://hostexploit.com/downloads/Atrivowhitepaper082808ac.pdf`.

11. Asaka, M., Okazawa, S., Taguchi, A. and Goto, S. 1999. The Implementation of IDA: An Intrusion Detection Agent Systems. In Proceedings of the Eleventh FIRST Conference 1999, Brisbane, Australia.

12. Asaka, M., Okazawa, S., Taguchi, A. and Goto, S. 1999. A Method of Tracing Intruders by Use of Mobile Agents. In Proceedings of the Ninth Annual Conference of the Internet Society.

13. Asaka, M., Tsuchiya, M., Onabuta, T., Okazawa, S. and Goto, S. 1999. Local Attack Detection and Intrusion Route Tracing. *IEICE Transaction on Communications*, Vol.E82-B, No.11, pp.1826-1833.

14. Baayen, R. H., Van Halteren, H. and Tweedie, F. 1996. Outside the cave of shadows: Using syntactic annotation to enhance authorship attribution. *Literary and Linguistic Computing*, Vol. 11, pp. 121–131.

15. Baecher, P., Koetter, M., Holz, T., Dornseif, M. and Freiling, F. C. 2006. The nepenthes platform: An efficient approach to collect malware. In Proceedings of the Ninth Symposium on Recent Advances in Intrusion Detection (RAID06), pp. 165–184.

16. Bailey, M., Oberheide, J., Andersen, J., Mao, Z. M., Jahanian, F. and Nazario, J. 2007. Automated classification and analysis of internet malware. In Proceedings of the 10th International Symposium on Recent Advances in Intrusion Detection (RAID07).

17. Barthes, R. 1984. The Death of the Author. *Image, Music, Text: Essays Selected and Translated by Stephen Heath*, pp. 148.

18. Barnett, R. C. 2004. Open proxy honeypots. Honeynet Project.

19. Bayer, U. , Kruegel, C. and Kirda, E. 2006. TTAnalyze: A Tool for Analyzing Malware. In the Fifteenth European Institute for Computer Antivirus Research (EICAR 2006) Annual Conference.

20. Bayer, U., Moser, A., Kruegel, C. and Kirda, E. 2006. Dynamic analysis of malicious code. *Journal in Computer Virology*, Vol. 2, pp. 67–77.

21. Bayer, U., Milanicomparetti, P., Hlauschek, C., Kruegel, C. and Andkirda, E. 2009. Scalable, Behavior-based Malware Clustering. In Proceedings of Network and Distributed System Security Symposium.

22. J. Bell and B. Whaley. *Cheating and Deception*. Transaction Publishers, New Brunswick, NJ, 1982.

23. Bellovin, S. 2003. ICMP Traceback Messages. IETF - Internet Engineering Task Force. `http://tools.ietf.org/id/draft-ietf-itrace-01.txt`.

24. Betjtlich, R. 2010. Attribution Using 20 Characteristics. TaoSecurity. `http://taosecurity.blogspot.com/2010/01/attribution-using-20-characteristics.html`.

25. Birk, D., Gajek, S., Grbert, F. and Sadeghi, A.-R. 2007. A Forensic Framework for Tracing Phishers. IFIP Summer School on The Future of Identity in the Information Society, Karlstad, Sweden.

26. Bizeul, D. 2007. Russian Business Network Study. `http://www.bizeul.org/files/RBNstudy.pdf`.

27. Bloom, B., Space/time trade-offs in hash coding with allowable errors, Communications of the ACM, 13, No. 7:422426, July 1970.

28. Blum, A., Song, D. and Venkataraman, S. 2004. Detection of Interactive Stepping Stones with Maximum Delay Bound: Algorithms and Confidence Bounds. In Proceedings of the Seventh International Symposium on Recent Advances in Intrusion Detection (RAID).

29. Bongardt, S. A. 2010. An introduction to the behavioral profiling of computer network intrusions. *The Forensic Examiner*.

30. Bowen, B. M., Kemerlis, V. P., Prabhu, P., Keromytis, A. D. and Stolfo, S. J. 2010. Automating the injection of believable decoys to detect snooping. In Proceedings of the third ACM conference on Wireless network security, ACM New York, NY, USA.

31. Bowen, B. M. 2011. Design and Analysis of Decoy Systems for Computer Security. PhD Thesis. School of Arts and Sciences, Columbia University.

32. Bowyer, K. W. and Hall, L. O. 1999. Experience using MOSS to detect cheating on programming assignments. In Proceedings of the Twenty-Ninth ASEE/IEEE Frontiers in Education Conference, San Juan, Puerto Rico, pp. 18–22.

33. Bradbury, D. 2010. Shadows in the cloud: China involvement in advanced persistent threats. *network Security*.

34. Brennan, M. and Greenstadt, R. 2009. Practical attacks against authorship recognition techniques. In Proceedings of the Twenty-first IAAI Conference on Artificial Intelligence.

35. Bull, J., Collins, C., Coughlin, E. and Sharp, D. 2002. Technical review of plagiarism detection software report. Technical Report LU1 3JU, Computer Assisted Assessment Centre, University of Luton, Bedfordshire, United Kingdom.

36. Burger, J. and Henderson, J. 2006. An exploration of features for predicting blogger age. In AAAI Spring Symposium on Computational Approaches to Analyzing Weblogs.

37. Burger, J., Henderson, J., Kim, G. and Zarrella, G. 2011. Discriminating gender on Twitter. Technical paper. The MITRE Corporation.

38. Burgess, A. W., Hartman, C. R., Ressler, R. K., Douglas, J. E. and McCormack, A. 1986. Sexual homicide: A motivational model. *Journal of Interpersonal Violence*, Vol. I, pp. 251–272.

39. Burrows, J.F. 1987. Word patterns and story shapes: The statistical analysis of narrative style. *Literary and Linguistic Computing*, Vol. 2, 61–67.

40. Burrows, S. D. 2010. Source Code Authorship Attribution. PhD Thesis. RMIT University, Melbourne, Australia.

41. Canter, D. 1989. Offender profiles. *The Psychologist*, Vol. 2, No. , No. 1, pp. 12–16.

42. Canter, D. 2004. Offender Profiling and Investigative Psychology. *Journal of Investigatice Psychology and Offender Profiling*, Vol. 1, pp 1–15.

43. Canter, D. and Fritzon, K. 1998. Differentiating arsonists: A model of firesetting actions and characteristics. *Journal of Criminal and Legal Psychology*, Vol. 3, pp. 73–96.

44. Canter, D. and Gregory, A. 1994. Identifying the residential location of rapists. *Journal of the Forensic Science Society*, Vol. 34, pp. 169–175.

45. Cattell, R. B., Cattell, A. K., and Cattell, H. E. P. 1993. The 16 Personality Factors Questionnaire (5th ed.). Champaign, IL: Institute for Personality and Ability Testing.

46. Cattell, R. B., and Cattell, M. D. L. 1969. Handbook for the high school personality questionnaire. Champaign, IL: Institute for Personality and Ability Testing.

47. Carvalho, V. R. and Cohen, W.W. 2004. Learning to Extract Signature and Reply Lines from Email. IN Proceedings of the 2004 Conference on Email and Anti-Spam. Mountain View, California, US.

48. Carvalho V., and Cohen, W., 2006. Improving Email Speech Act Analysis via N-gram Selection. In Proceedings of the Joint Conference on Human Language Technologies and the Annual Meeting of the North American Chapter of the Association of Computational Linguistics (HLT-NAACL).

49. Carvalho, V. R. 2008. Modeling Intention in Email. PhD Thesis. Carnegie Mellon University, US.

50. Carrier, B. and Shields, C. 2002. A Recursive Session Token Protocol For Use in Computer Forensics and TCP Traceback. `http://citeseer.nj.nec.com/510558.html`.

51. Chang, H. Y., Chen, P., Hayatnagarkar, A., Narayan, R., Sheth, P., Vo, N., Wu, C. L., Wu, S. F., Zhang, L., Zhang, X., Gong, F., Jou, F., Sargor, C. and Wu., X. 2000. Design and Implementation of A Real-Time Decentralized Source Identification System for Untrusted IP Packets. In Proceedings of the DARPA Information Survivability Conference and Exposition (DISCEX 2000). IEEE Computer Society Press.

52. Chang, H. Y., Narayan, R., Wu, S. F., Vetter, B. M., Brown, M., Yuill, J.J., Wang, X., Sargor, C., Jou, F. and Gong, F. 1999. Deciduous: Decentralized Source Identification for Network-based Intrusions. In Proceedings of the Sixth IFIP/IEEE International Symposium on Integrated Network Management, IEEE Communications Society Press.

53. Chaski, C. E. 2001. Empirical evaluation of language-based author identification techniques. *Forensic Linguistics*, Vol. 8, No. 1. pp. 1–65.

54. Chen, W.-B. and Zhang, C. 2009. Image Spam Clustering: An Unsupervised Approach. In Proceedings of the First ACM workshop on Multimedia in forensics, New York, NY, USA.

55. Chisum, W. J. and Rynearson, J. M. 1997. Evidence and crime scene reconstruction (5th ed.). National Crime Scene Investigation and Training.

56. Clark, D. and Landau, S. 2010. Untangling Attribution. In proceedings of a Workshop on Deterring CyberAttacks:Informing Strategies and Developing Options for U.S. Policy, National Research Council.

57. Clement, R. and Sharp, D. 2003. Ngram and Bayesian Classification of Documents. *Literary and Linguistic Computing*, Vol. 18, pp. 423–447.

58. Cloppert, M. 2009. Security Intelligence: Attacking the Kill Chain. SANS Computer Forensics. `http://computer-forensics.sans.org/blog/2009/10/14/security-intelligence-attacking-the-kill-chain/`.

59. Cohen, F. B. 2010. Attribution of messages to sources in digital forensics cases. In Proceedings of the 43rd Hawaii International Conference on System Sciences, Hawaii, US.

60. Cohen, W., Carvalho, V. and Mitchell, T. Learning to classify email into 'Speech Acts'. In Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP-2004), pp. 309–316.

61. Carvalho, V., and Cohen, W. W. 2005. On the Collective Classification of Email Speech Acts. In Proceedings of the 28th Annual International ACM Special Interest Group on Information Retrieval Conference on Research and Development in Information Retrieval, New York: Association for Computing Machinery, pp. 345–352.

62. Cohen, D. and Narayanaswamy, K. 2004. Survey/Analysis of Levels I, II, and III Attack Attribution Techniques. Technical Report. Advanced Research Development Activity (ARDA).

63. Collberg, C. and Kobourov, S. 2005. Self-plagiarism in computer science. *Communications of the ACM*, Vol. 48, No. 4, pp. 88–94.

64. Corney, M., Anderson, A., Mohay, G. and de Vel, O. 2002. Identifying the Authors of Suspect Email. *Computers Security Journal*.

65. Corney, M. W. 2003. Analysing E-mail Text Authorship for Forensic Purposes. Master Thesis. Queensland University of Technology.

66. Creamer, G., Rowe, R., Hershkop, S. and Stolfo, S. J. 2009. Segmentation and Automated Social Hierarchy Detection through Email Network Analysis. *Advances in Web Mining and Web Usage Analysis*,*Lecture Notes in Computer Science*, Vol. 5439, pp. 40–5.

67. Culotta, A., Bekkerman, R. and McCallum, A. 2004. Extracting social networks and contact information from email and the web. In Proceedings of the first Conference on Email and Anti-Spam, Mountain View, California, US.

68. Dacier, M., Leita, C., Thonnard, O., Pham, V. H. and Kirda, E. 2009. Assessing Cybercrime Through the Eyes of the WOMBAT, chapter 3 (pp. 103-136) in *Cyber Situational Awareness: Issues and Research*, Sushil Jajodia, Peng Liu, Vipin Swarup, Cliff Wang, eds., ISBN: 98-1-4419-0139-2, Springer International Series on Advances in Information Security.

69. De Morgan, S. 1882. *Memoir of Augustus de Morgan by his Wife Sophia Elizabeth de Morgan With Selections From His Letters*. London: Longmans, Green, and Co.

70. De Vel, O. 2000. Mining e-mail authorship. In Proceedings of the Workshop on Text Mining, ACM International Conference on Knowledge Discovery and Data Mining (KDD2000), Boston, MA.

71. De Vel, O. , Anderson, A., Corney, M. and Mohay, G. 2001. Mining E-mail Content for Author Identification Forensics. *SIGMOD Record*, Vol. 30, No. 4, pp. 55–64.

72. Dean, D., Franklin, M. and Stubblefield, A. 2001. An algebraic approach to IP Traceback. In Proceedings of the 2001 Network and Distributed System Security (NDSS) Symposium.

73. Deibert, R., Manchanda, A., Rohozinski, R., Villeneuve, N. and Walton, G. 2009. Tracking GhostNet: Investigating a cyber espionage network. `http://www.scribd.com/doc/13731776/Tracking-GhostNet-Investigating-a-Cyber-Espionage-Network`.

74. Depue, R., Douglas, J., Hazelwood, R., and Ressler, R. 1995. Criminal Investigative Analysis: An Overview. In Burgess, A. and Hazelwood, R. (Eds), *Practical Aspects of Rape Investigation*. CRC Press.

75. Diederich, J., Kindermann, J., Leopold, E. and Paass, G. 2003, Authorship Attribution with Support Vector Machines, Applied Intelligence, Vol. 19, No. 1, pp. 109–123.

76. Ding, H. and Samadzadeh, M. H. 2004. Extraction of Java program fingerprints for software authorship identification. *Journal of Systems and Software*, Vol. 72, No. 1, pp. 49–57.

77. Dobitz, K., Haas, B., Holtje, M., Jokerst, A., Ochsner, G., Silva, S., Johnson, K., Hudson, I. and John, G. 2008. The Characterization and Measurement of Cyber Warfare. Project 08-01. Global Innovation and Strategy Center, Omaha, NE, US.

78. Doeppner, T., Philip, W., Klein, N. and Koyfman, A. 2000. Using Router Stamping to Identify the Source of IP Packets. In Proceedings of the Seventh ACM Conference on Computer and Communications Security (CCS), Athens, Greece, pp. 184–189.

79. Dom, B., Eiron, I., Cozzi, A. and Zhang, Y. 2003. Graph-based ranking algorithms for e-mail expertise analysis. In Proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, San Diego, California, US, pp. 42–48.

80. Donoho, D., Flesia, A., Shankar, U., Paxson, V., Coit, J. and Staniford, S. 2002. Multiscale Stepping-stone Detection: Detecting Pairs of Jittered Interactive Streams by Exploiting Maximum Tolerable Delay. In Proceedings of the Fifth International Symposium on Recent Advances in Intrusion Detection (RAID).

81. Dowden, C., Bennell, C. and Bloomfield, S. 2007. Advances in offender profiling: A systematic review of the profiling literatures published over the past 30 years. *Journal of Police and Criminal Psychology*, Vol. 22, pp. 44–56.

82. Drake, C. E., Oliver, J. J. and Koontz, E. J. 2004. Anatomy of a Phishing Email. In Proceedings of the First Conference on Email and Anti-Spam (CEAS), Mountain View, CA, USA..

83. Dreier, D. J. 2009. Blog Fingerprinting: Identifying Anonymous Posts Written by an Author of Internet Using Word and Character Frequency Analysis. Master Thesis. Naval Postgraduate School, Monterey, California, US.

84. Duffek, D. N., Pouzin, J., Dominic, J. E., Hilerio, I. and Curry, L. G. 2004. Method and apparatus to associate a modifiable CRM related token to an email. United States Patent Application 20060155715.

85. Eddy, H.T. 1887. The Characteristic Curves of Composition. *Science*, pp. 237–246

86. Eissen, S. M. and Stein, B. 2006. Intrinsic plagiarism detection. In Proceedings of the Twenty-Eighth European Conference on IR Research, London, United Kingdom, pp. 565–569.

87. Elenbogen, B. S. and Seliya, N. 2008. Detecting outsourced student programming assignments. *Journal of Computing Sciences in Colleges*, Vol. 23, No. 3, pp. 50–57.

88. Ellegrd, A. 1962a. A Statistical Method for Determining Authorship: the junius letters, pp. 1769–1772. Gothenburg: Acta Universitatis Gothoburgensis.

89. Els, T. 2009. Identity Resolution in Email Collections. PhD Thesis. University of MaryLand.

90. Elsayed, T. and Oard, D. W. 2006. Modeling identity in archival collections of email: a preliminary study. In Third Conference on Email and Anti-spam (CEAS), Mountain View, CA, US.

91. Ester, M., Kriegel, H.-P., Sander, J. and Xu, X. 1996. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In Proceedings of knowledge discovery in databases.

92. Estival, D. 2008. Author attribution with email messages. *Journal of Science*. Vietnam National University, pp. 1–9.

93. Estival, D., Gaustad, T., Pham, S. B., Radford, W. and Hutchinson, B. 2007. TAT: an author profiling tool with application to Arabic emails. In Proceedings of the 5th Australasian Language Technology Workshop, Melbourne, Australia. pp. 21–30.

94. Frank, E. and Witten, I. H. 1998. Generating accurate rule sets without global optimization. In Proceedings of the Fifteenth International Conference on Machine Learning, Madison, San Francisco, pp. 144–151.

95. Farnham, W. 1916. Colloquial Contractions in Beaumont, Fletcher, Massinger, and Shakespeare as a Test of Authorship. *Modern Language Association of America*, Vol. 31, pp. 326–358.

96. Ferguson, P. and Senie, D. 1998. Network ingress filtering: Defeating denial of service attacks which employ IP source address spoofing. RFC 2267 - Internet Engineering Task Force.

97. Forsyth, R.S. and Holmes, D.I. 1996. Feature finding for text classification. *Literary and Linguistic Computing*, Vol. 11, No. 4, pp. 163–174.

98. Frantzeskou, G., Gritzalis, S. and MacDonell, S. G. 2004. Source code authorship analysis for supporting the cybercrime investigation process. In Proceedings of the First International Conference on E-business and Telecommunication Networks, Setubal, Portugal, pp. 85–92.

99. Frantzeskou, G., MacDonell, S. G., Stamatatos, E. and Gritzalis, S. 2008. Examining the significance of high-level programming features in source code author classification. *Journal of Systems and Software*, Vol. 81, No. 3, pp. 447–460.

100. Frantzeskou, G., Stamatatos, E. and Gritzalis, S. 2005. Supporting the cybercrime investigation process: Effective discrimination of source code authors based on byte-level information. In Proceedings of the Second International Conference on E-business and Telecommunication Networks, eading, United Kingdom, pp. 283–290.

101. Frantzeskou, G., Stamatatos, E., Gritzalis, S. and Katsikas, S. 2006. Source code author identification based on n-gram author profiles. *Artificial Intelligence Applications and Innovations*, pp. 508–515. Springer.

102. Frantzeskou, G., Stamatatos, E., Gritzalis, S. and Katsikas, S. 2006. Effective identification of source code authors using byte-level information. In Proceedings of the Twenty-Eighth International Conference on Software Engineering, Shanghai, China, pp. 893–896.

103. Frantzeskou, G., Stamatatos, E., Gritzalis, S., Chaski, C. E. and Howald, B. S. 2007. Identifying authorship by byte-level n-grams: The source code author profile (SCAP) method. *International Journal of Digital Evidence*, Vol. 6, No. 1.

104. Gamon, M. 2004. Linguistic correlates of style: Authorship classification with deep linguistic analysis features. In Proceedings of the 20th International Conference on Computational Linguistics, pp. 611–617.

105. Gamon, M. 2004. Linguistic correlates of style: Authorship classification with deep linguistic analysis features. In Proceedings of the 2004 International Conference on Computational Linguistics (COLING), pp. 611–617.

106. Garera, S., Provos, N., Chew, M., and Rubin, A. D. 2007. A Framework for Detection and Measurement of Phishing Attacks. Proc. of the ACM Workshop on Rapid Malcode (WORM), Alexandria, VA.

107. Genkin, A., Lewis, D. and Madigan, D. 2006, Large-scale Bayesian logistic regression for text categorization. *Technometrics*.

108. Gero, A. 2006. Tracking electronic mail messages. United States Patent Application 20060259556.

109. Gershwin, L. K. 2001. Statement for the Record for the Joint Economic Committee: Cyber Threat Trends and US Network Security. `http://www.house.gov/jec/hearings/gershwin.pdf`.

110. Gheorghescu, M. 2005. An Automated Virus Classification System. In Proceedings of the 2005 Virus Bulletin Conference, pp.294300.

111. Granger, S., Dagneaux, E. and Meunier, F. 2002, The International Corpus of Learner English. Handbook and CD-ROM. Louvain-la-Neuve: Presses Universitaires de Louvain.

112. Gray, A. R., Sallis, P. J. and MacDonell, S. G. 1997. Software forensics: Extending authorship analysis techniques to computer programs. In Proceedings of the Third Biannual Conference of the International Association of Forensic Linguists, Birmingham, UK, pp. 1–8.

113. Gray, A. R., Sallis, P. J. and MacDonell, S. G.. 1998. IDENTIFIED (Integrated Dictionary-based Extraction of Non-language-dependent Token Information for Forensic Identification, Examination, and Discrimination): a dictionary-based system for extracting source code metrics for software forensics. In Proceedings of Software Engineering: Education and Practice, New Zealand, pp. 252–259.

114. Grieve, J. 2007. Quantitative authorship attribution: An evaluation of techniques. *Literary and Linguistic Computing*, Vol. 22, No. 3, pp. 251–270.

115. Griffith, V. and Jakobsson, M. 2006.. Messin with Texas: Deriving mothers maiden names using public records. *RSA CryptoBytes*, Vol. 8, No. 1.

116. Grow. B., Epstein, K. and Tschang, C. 2008. The New E-spionage Threat. *A BusinessWeek*.

117. Grozea, C., Gohl, C. and Popescu, M. 2009. ENCOPLOT: Pairwise sequence matching in linear time applied to plagiarism detection. In Proceedings of the Third PAN Workshop on Uncovering Plagiarism, Authorship and Social Software Misuse, San Sebastian, Spain, pp. 10–18.

118. Guan, Y. and Zhang, L. 2008. Attack traceback and attribution. *Wiley Handbook of Science and Technology for Homeland Security*. Wiley-Interscience.

119. Hamilton, M., Tahaghoghi, S. M. M. and Walker, C. 2004. Educating students about plagiarism avoidance a computer science perspective. In Proceedings of the Twelfth International Conference on Computers in Education, Melbourne, Australia, pp. 1275–1284.

120. Hevner, A. R. and March, S. T. 2003. The Information System Research Cycle. *MIS Quarterly*, Vol. 28, No. 1, pp. 111–113.

121. Hicks, S. J. and Sales, B. D. 2006. Criminal Profiling: Developing an Effective Science and Practice. *American Psychological Association*.

122. Hirst, G. and Feiguina, O. 2007. Bigrams of syntactic labels for authorship discrimination of short texts. *Literary and Linguistic Computing*, Vol. 22, No. 4, pp. 405–417.

123. Holmes, D. 1985. The Analysis of Literary Style: A Review. *The Journal of the Royal Statistical Society A*, Vol. 148, pp. 328–341.

124. Holmes, D. 1998. The evolution of stylometry in humanities scholarship. *Literary and Linguistic Computing*, Vol. 13, No. 3, pp. 111–117.

125. Holmes, D.I. 1992. A stylometric analysis of Mormon scripture and related texts. *Journal of Royal Statistical Society*, Vol.1 55, pp. 91–120.

126. Holmes, D. and Forsyth, R. 1995. The Federalist revisited: New directions in authorship attribution. *Literary and Linguistic Computing*, pp. 111–127.

127. Holmes, R. and Holmes, S. 1996. *Profiling violent crimes: An investigative tool.* Thousand Oaks, CA: Sage.

128. Holz, T. and Engelberth, M. 2009, Felix Freiling, Learning More About the Underground Economy: A Case-Study of Keyloggers and Dropzones. In the Fourteen European Symposium on Research in Computer Security (ESORICS 2009), Saint Malo, Brittany, France.

129. Holzer, R., Malin, B. and Sweeney, L. 2005. Email alias detection using social network analysis. In Proceedings of the Third International Workshop on Link discovery, Illinois, United States, pp. 52–57.

130. Holz, T., Willems, C., Rieck, K., Duessel, P. and Laskov, P. 2008. Learning and Classification of Malware Behavior. In Proceedings of the Fifth Conference on Detection of Intrusions and Malware and Vulnerability Assessment (DIMVA 08).

131. Homem, N. and Carvalho, J.P. 2011. Authorship identification and author fuzzy "fingerprints". Fuzzy Information Processing Society (NAFIPS), 2011 Annual Meeting of the North American , El Paso, TX, pp. 1–6.

132. Homem, N. and Carvalho, J.P 2011. Authorship identification and author fuzzy "fingerprints". In Proceedings of Fuzzy Information Processing Society (NAFIPS), 2011 Annual Meeting of the North American, pp. 1—6.

133. Hoorn, J., Frank, S., Kowalczyk, W. and van der Ham, F. 1999, Neural network identification of poets using letter sequences. *Literary and Linguistic Computing*, Vol. 14, No. 3, pp. 311–338.

134. Hunker, J., Hutchison, B. and Margulies, J. 2008. Role and Challenges for Sufficient CyberAttack Attribution. Dartmouth College: The Institute for Information Infrastructure Protection (The I3P).

135. Iqbal, F., Hadjidj, R., Fung, B. C. and Debbabi, M. 2008. A Novel Approach of Mining Write-Prints for Authorship Attribution in E-mail Forensics. *Digital Investigation*, Vol. 5, No. 1, pp. 42–51.

136. Iqbal, F. 2011. Messaging Forensic Framework for Cybercrime Investigation. PhD Thesis. Concordia University, Canada.

137. Iqbal, F., Binsalleeh, H., Fung, B.C.M. and Debbabi, M. 2010. Mining writeprints from anonymous e-mails for forensic investigation. *Digital Investigation.*

138. Iqbal, F., Binsalleeh, H., Fung, B. C. M. and Debbabi, M. 2011. A unified data mining solution for authorship analysis in anonymous textual communications . *Information Sciences*, doi:10.1016/j.ins.2011.03.006.

139. Iqbal, F., Khan, L.A., Fung, B.C.M. and Debbabi, M. 2010. E-mail authorship verification for forensic investigation. In Proceedings of the 25th ACM SIGAPP Symposium on Applied Computing (SAC), Sierre, Switzerland, pp. 1591–1598.

140. Jahankhani, H. and Al-Nemrat, A. 2010. Examination of cyber-criminal behaviour. *International Journal of Information Science and Management*, Special Issue, pp 41–48.

141. Jagatic, T., Johnson, N., Jakobsson, M. and Menczer, F. 2007. Social phishing. *Communications of the ACM*, Vol. 50, No. 10, pp. 94–100.

142. Jakobsson, M., Jagatic, T. and Stamm, S. 2005. Phishing for clues: Inferring context using cascading style sheets and browser history. `http://www.browser-recon.info`.

143. Jakobsson, M. and Myers, S. 2007. *Phishing and Countermeasures*. Wiley-Intersience. A John Wiley and Sons, Inc.

144. James, L. 2005. *Phishing Exposed*. Rockland MA Syngress Publishing.

145. Jiang, X. and Xu, D. 2004. Collapsar: A VM-based architecture for network attack detention center. In Proceedings of the 13th USENIX Security Symposium.

146. Jonas, J. 2006. Identity Resolution: 23 Years of Practical Experience and Observations at Scale. In Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data Chicago. ACM Press.

147. Joula, P. 2006. Authorship attribution. *Foundations and Trends in Information Retrieval*, Vol. 1, No. 3, pp. 233–334.

148. Juola, P. 2009. JGAAP: A System for Comparative Evaluation of Authorship Attribution. In Proceedings of the Chicago Colloquium on Digital Humanities and Computer Science, Vol.1 , No. 1, pp. 1–5.

149. Joy, M. and Luck, M. 1999. Plagiarism in programming assignments. IEEE Transactions on Education, Vol. 42 , No. 2, pp. 129–133.

150. Karim, Md. E., Walenstein, A., Lakhotia, A. and Parida, L. 2005. Malware phylogeny generation using permutations of code. *Journal In Computer Virology*, No. 12, pp. 13–23.

151. Karlgren, J. and Eriksson, G. 2007. Authors, genre, and linguistic convention. In Proceedings of the SIGIR Workshop on Plagiarism Analysis, Authorship Attribution, and Near-Duplicate Detection, pp. 23–28.

152. Kacmarcik, G. and Gamon, M. 2006. Obfuscating document stylometry to preserve author anonymity. In Calzolari, N., Cardie, C. and Isabelle, P. (eds), Proceedings of the Twenty-First International Conference on Computational Linguistics and Forty-Fourth Annual Meeting of the Association for Computational Linguistics Main Conference Poster Sessions, pp. 444–451, Sydney, Australia.

153. Kaufman, K. L., Hilliker, D. R., Lathrop, P. and Daleiden, E. L. 1993. Assessing child sexual offenders' modus operandi: Accuracy in self-reported use of threats and coercion. *Annals of Sex Research*, Vol. 6, pp. 213–229.

154. Keselj, V., Peng, F., Cercone, N. and Thomas, C. 2003. N-gram-based Author Profiles for Authorship Attribution. In Proceddings of the Conference Pacific Association for Computational Linguistics.

155. Khmelev, D.V. and Tweedie, F.J. 2001. Using Markov chains for identification of writers. *Literary and Linguistic Computing*, Vol. 16, No. 4, pp. 299–307.

156. Kirda, R. 2007. D12 (D5.1) Root Causes Analysis. Worldwide Observatory of Malicious Behaviors and Attack Threats (WOMBAT).

157. Kocsis, R. N. and Cooksey, R. W. 2002. Criminal psychological profiling of serial arson crimes. *International journal of Offender Therapy and Comparative Criminology*, Vol. 46, pp. 631–656.

158. Kolter, J. Z. and Maloof, M. A. 2006. Learning to detect and classify malicious executables in the wild. *Journal of Machine Learning Research*, Vol. 7, pp. 2721–2744.

159. Koppel, M. and Schler, J. 2003. Exploiting stylistic idiosyncrasies for authorship attribution. In Proceedings of IJCAI03 Workshop on Computational Approaches to Style Analysis and Synthesis, pp. 69–72.

160. Kjell, B. 1994a. Authorship attribution of text samples using neural networks and Bayesian classifiers. In Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, San Antonio, Texas.

161. Koppel, M., Argamon, S. and Shimoni, A. R. 2002. Automatically categorizing written texts by author gender. *Literary and Linguistic Computing*, Vol. 17, No. 4, pp. 401–412.

162. Koppel, M., Argamon, S. Shimoni, A. 2002, Automatically categorizing written texts by author gender. *Literary and Linguistic Computing*, Vol. 17, No. 4, pp. 401–412.

163. Koppel, M., Akiva, N. and Dagan, I. 2006. Feature instability as a criterion for selecting potential style markers. *Journal of the American Society for Information Science and Technology*, Vol. 57, No. 11, pp. 1519–1525.

164. Koppel, M., Schler, J. and Bonchek-Dokow, E. 2007. Measuring differentiability: Unmasking pseudonymous authors. *Journal of Machine Learning Research*, Vol. 8, pp. 1261–1276.

165. Koppel, M. and Schler, J. 2003, Exploiting Stylistic Idiosyncrasies for Authorship Attribution. In Proceedings of IJCAI'03 Workshop on Computational Approaches to Style Analysis and Synthesis, pp. 69–72.

166. Koppel, M. and Schler, J. 2004. Authorship verification as a one-class classification problem. In Proceedings of the 21st International Conference on Machine Learning, NewYork, ACM Press, pp. 62.

167. Koppel, M., Schler, J. and Zigdon, K. 2005. Determining an Authors Native Language by Mining a Text for Errors. In Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 05), Chicago IL.

168. Koppel, M., Schler, J. and Argamon, S. 2009. Computational Methods in Authorship Attribution. *Journal of the American Society for information Science and Technology*, Vol. 60, No. 1, pp. 9–26.

169. Koppel, M., Schler, J. and Argamon, S. 2010. Authorship attribution in the wild. *Language Resources and Evaluation*. Springer.

170. Kothari, J., Shevertalov, M., Stehle, E. and Mancoridis, S. 2007. A probabilistic approach to source code authorship identification. In Proceedings of the Fourth International Conference on Information Technology, Las Vegas, Nevada, pp. 243–248.

171. Krebs, B. 2007. Taking on the Russian Business Network. `http://voices.washingtonpost.com/securityfix/2007/10/taking_on_the_russian_business.html`.

172. Krebs, B. 2008. Report Slams U.S. Host as Major Source of Badware. `http://voices.washingtonpost.com/securityfix/2008/08/report_slams_us_host_as_major.html`.

173. Krsul, I. 1994. Authorship analysis: Identifying the author of a program. Technical Report CSD-TR-94-030, Department of Computer Sciences, Purdue University.

174. Krsul, I. and Spafford, E. H. 1996. Authorship analysis: Identifying the author of a program. Technical Report TR-96-052, Department of Computer Sciences, Purdue University.

175. Krsul, I. and Spafford, E. H. 1997. Authorship analysis: Identifying the author of a program. *Computers and Security*, Vol. 16, No. 3, pp. 233–257.

176. Kukushkina, O. V., Polikarpov, A. A. and Khmelev, D. V. 2001. Using literal and grammatical statistics for authorship attribution. *Problems of Information Transmission*, Vol. 37, No. 2, pp. 172–184.

177. Kwan, L., Ray, P. and Stephens, G. 2008. Towards a Methodology for Profiling Cyber Criminals. In proceedings of the 41st Hawaii International Conference on System Sciences.

178. Lampert, A., Dale, R. and Paris, C. 2009. Segmenting email message text into zones. In Proceedings of Empirical Methods in Natural Language Processing, Singapore, pp. 919–928.

179. Lange, R. C. and Mancoridis, S. 2007. Using code metric histograms and genetic algorithms to perform author identification for software forensics. In Proceedings of the Ninth Annual Conference on Genetic and Evolutionary Computation, London, United Kingdom, pp. 2082–2089.

180. Layton, R., Brown, S. and Watters, P. 2009. Using Differencing to Increase Distinctiveness for Phishing Website Clustering . In Proceedings of Ubiquitous, Autonomic and Trusted Computing, Brisbane, Queensland, Australia, pp. 488–492.

181. Layton, R. and Watters, P. 2009. Determining provenance of phishing websites using automated conceptual analysis. In Proceedings of eCrime Researchers Summit, Tacoma, Western Australia, Australia, pp. 1 – 7.

182. Ledger, G.R. and Merriam, T.V.N. 1994. Shakespeare, Fletcher, and the two Noble Kinsmen. *Literary and Linguistic Computing*, Vol.9, pp. 235–248.

183. Lee, T. and Mody, J. J. 2006. Behavioral Classification. In European Expert Group for IT-Security (EICAR) Conference.

184. Lee, W. and Park, K. 2001. On the Effectiveness of Probabilistic Packet Marking for IP Traceback under Denial of Service Attack. In Proceedings of the IEEE International Conference on Computer Communications.

185. Leita, C., Dacier, M. and Massicotte, F. 2006. Automatic handling of protocol dependencies and reaction to 0-day attacks with ScriptGen based honeypots. In Proceedings of the 9th Symposium on Recent Advances in Intrusion Detection (RAID06).

186. Li, J., Zheng, R. and Chen, H. 2006. From fingerprint to writeprint. *Communications of the ACM*, Vol. 49, pp. 76–82.

187. Linn, C. and Debray, S. 2003. Obfuscation of executable code to improve resistance to static disassembly. In Proceedings of the 10th ACM conference on Computer and communications security, New York, NY, USA, pp. 290–299.

188. Longstaff, T. A. and Schultz, E. E. 1993. Beyond preliminary analysis of the WANK and OILZ worms: A case study of malicious code. *Computers and Security*, Vol. 12, No. 1, pp. 61–77.

189. Lowe, D. and Matthews, R. 1995. Shakespeare vs. Fletcher: A stylometric analysis by radial basis functions. *Computers and the Humanities*, Vol. 29, pp. 449–461.

190. Luyckx, K., and Daelemans, W. 2008. Authorship attribution and verification with many authors and limited data. In Proceedings of the 22nd International Conference on Computational Linguistics (COLING 2008), Manchester, UK, pp. 513–520.

191. Luyckx, K. and Daelemans, W. 2011. The effect of author set size and data size in authorship attribution. *Literary and Linguistics Computing*, Vol. 26, No. 1. Oxford University Press.

192. MacDonell, S. G., Gray, A. R., MacLennan,G. and Sallis, P. J. 1999. Software forensics for discriminating between program authors using case-based reasoning, feed-forward neural networks and multiple discriminant analysis. In Proceedings of the Sixth International Conference on Neural Information Processing, Perth, Australia, pp. 66–71.

193. MacDonell, S. G., Buckingham, D., Gray, A. R. and Sallis, P. J. 2004. Software forensics: Extending authorship analysis techniques to computer programs. *Journal of Law and Information Science*, Vol. 13, No. 1.

194. McRae, C. M. and Vaughn, R. B. 2007. Phighting the phisher: Using web bugs and honeytokens to investigate the source of phishing attacks. In Proceedings of the 40th Hawaii International Conference on System Sciences (HICSS), Washington, DC, USA, 2007, pp. 270c–270c.

195. Madigan, D., Genkin, A., Lewis, D., Argamon, S., Fradkin, D. and Ye, L. 2005. Author identification on the large scale. In Proceedings of Classification Society of North America 05.

196. Mankin, A., Massey, D., Wu, C.-H., Wu, S. F. and Zhang, L. 2001. On Design and Evolution of Intention-Driven ICMP Traceback. In Proceedings of IEEE International Conference on Computer Communications and Networks.

197. Mascol, C. (a.k.a. Smith, W. B.) 1888a. Curves of Pauline and Pseudo-Pauline Style I. *Unitarian Review*, Vol. 30, pp. 452–460.

198. Matthews, R. A. J. and Merriam, T. V. N. 1993. Neural computation in stylometry I: An application to the works of Shakespeare and Fletcher. *Literary and Linguistic Computing*, Vol. 8, pp. 103–209.

199. McArthur, R. and Bruza, P. 2003. Discovery of implicit and explicit connections between people using email utterance. In Proceedings of the Eighth European Conference of Computer-supported Cooperative Work, Finland, pp. 21–40.

200. McCallum, A., Corrada-Emmanuel, A. and Wang, X. 2005. Topic and role discovery in social networks. In Proceedings of the Ninteenth International Joint Conference on Artificial Intelligence, Edinburgh, Scotland, UK, pp. 786–791.

201. McCarthy, P. M., Lewis, G. A., Dufty, D. F. and McNamara, D. S. 2006. Analyzing writing styles with coh-metrix. In Proceedings of the Florida Artificial Intelligence.

202. McCombie, S. J. 2011. Phishing the long line: Transnational Cybercrime from Eastern Europe to Australia. PhD Thesis. Macquarie University, Australia.

203. McCombie, S., Watters, P., Ng, A. and Watson, B. 2008. Forensic characteristics of phishing — petty theft or organized crime? In [202].

204. Mealand, D. L. 1995. Correspondence analysis of Luke. *Literary and Linguistic Computing*, Vol. 10, pp. 171–182.

205. Mendenhall, T. C. 1887. The Characteristic Curves of Composition. *Science*, Vol. 11, pp. 237–249.

206. Mendenhall, T. C. 1901. A Mechanical Solution to a Literary Problem. *Popular Science Monthly*, Vol. 9, pp. 97–110.

207. Merriam, T. 1988. Was Hand B in Sir Thomas More Heywoods Autograph? *Notes and Queries*, Vol. 233, pp. 455–458.

208. Merriam, T. 1994. Letter Frequency as a Discriminator of Authorship. *Notes and Queries*, Vol. 239, pp. 467–469.

209. Michalski, R. S., Carbonell, J. G. and Mitchell, T. M. 1983. *Machine learning: an artificial intelligence approach*. Morgan Kauf.

210. Minkov, E., Wang, R. C. and Cohen, W. W. 2005. Extracting personal names from email: applying named entity recognition to informal text. In Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing, Morristown, NJ, US, pp. 443–450.

211. Moore, T. 2007. Phishing and the economics of e-crime. *Infosecurity*. Elsevier.

212. Moore, T. and Clayton, R. 2007. An empirical analysis of the current state of phishing attack and defence. In Proceedings of the Sixth Workshop on the Economics of Information Security.

213. Morley, D. and Parker, C. 2009. *Understanding Computers: Today and Tomorrow.* CengageLearning.

214. Moser, A., Kruegel, C. and Kirda, E. 2007a. Exploring multiple execution paths for malware analysis. In Proceedings of 2007 IEEE Symposium on Security and Privacy.

215. Moser, A., Kruegel, C. and Kirda, E. 2007b. Limits of Static Analysis for Malware Detection. In Proceedings of Annual Computer Security Applications Conference (ACSAC), pp. 421–430.

216. Navarro, G. 2001. A guided tour to approximate string matching. *ACM Computing Surveys*, Vol. 33, No. 1, pp. 31–88.

217. Mosteller, F. and Wallace, D. 1963. Inference in an Authorship Problem. *Journal of The American Statistical Association*, Vol. 58, pp. 275–309.

218. Mosteller, F. and Wallace, D. 1964. Inference and Disputed Authorship: The Federalist (1st Edition). *Reading.* Addison-Wesley.

219. Mosteller, F. and Wallace, D. 1984. *Applied Bayesian and Classical Inference: The Case of the Federalist Papers* (2nd Edition). New York: Springer-Verlag.

220. Mozgovoy, M. 2007. Enhancing Computer-Aided Plagiarism Detection. PhD thesis, Department of Computer Science and Statistics, University of Joensuu, Joensuu, Finland.

221. Mozgovoy, M., Fredriksson, K., White, D., Joy, M. and Sutinen, E. 2005. Fast plagiarism detection system. In Proceedings of the Twelfth International Conference on String Processing and Information Retrieval, Buenos Aires, Argentina, pp. 267–270.

222. Nawrot, M. 2011. Automatic Author Attribution for Short Text Documents. *Human Language Technology. Challenges for Computer Science and Linguistics*, *Lecture Notes in Computer Science*, Vol. 6562, pp. 468–477.

223. Novak, J., Raghavan, P., and Tomkins, A. 2004. Anti-aliasing on the web. In Proceedings of the 13th International World Wide Web Conference, pp. 30-39.

224. Nykodym, N., Taylor, R. and Vilela, J. 2005. Criminal profiling and insider cyber crime. *Digital Investigation*, Vol. 2, No. 26, pp. 261–267

225. ODonnell, B. 1966. Stephen Cranes The ORuddy: a problem in authorship discrimination. In Leed (ed.) *The Computer and Literary Style.* Kent State University Press, pp. 107–115.

226. Oman, P. and Cook, C. 1989. Programming style authorshio analysis. In Proceedings of the Seventeenth Annual ACM Computer Science Conference, pp. 320–326.

227. Osborn, A. S. 1929. *Questioned Documents* (Second edition). Albany, NY: Biyd Printing Co.

228. Oudot, L. 2003. Fighting Spammers With Honeypots: Part 1. `http://www.securityfocus.com/infocus/1747`.

229. Oudot, L. 2003. Fighting Spammers With Honeypots: Part 2.http://www.symantec.com/connect/articles/fighting-spammers-honeypots-part-2.

230. Oudot, L. 2004. Wireless honeypot countermeasures. Technical report, SecurityFocus.

231. Parker, T., Shaw, E., Stroz, E., Devost, M. G. and Sachs, M. H. 2004. *Cyber Adversary Characterisation*. Syngress.

232. Parsons, L., Haque, E. and Liu, H. 2004. Subspace Clustering for High Dimensional Data: A Review. *SIGKDD Explorations*, Vol. 6, No. 1, pp. 90–105.

233. Partridge, C., Jones, C., Waitzman, D. and Snoeren, A. 2001. New Protocols to Support Internet Traceback. http://www.ir.bbn.com/documents/internet_drafts/draft-partridge-ippt-discuss-00.txt.

234. Peltier, Thomas R. 2006. Social Engineering: Concepts and Solutions. *The EDP Audit, Control, and Security Newsletter*(DPACS), Vol. 33, No. 8. Taylor and Francis.

235. Peng, P., Ning, P., Reeves, D. S. 2006. On the Secrecy of Timing-Based Active Watermarking Trace-Back Techniques. In Proceedings of the 2006 IEEE Symposium on Security and Privacy, pp. 334–348.

236. Peng, P., Ning, P., Reeves, D. S. and Wang, X. 2005. Active Timing-Based Correlation of Perturbed Traffic Flows with Chaff Packets. Technical Report TR-2005-11, Department of Computer Science, North Carolina State University

237. Peng, P., Ning, P., Reeves, D. S. and Wang, X. 2005. Active Timing-Based Correlation of Perturbed Traffic Flows with Chaff Packets. In Proceedings of the Second International Workshop on Security in Distributed Computing Systems (SDCS).

238. Peng, P., Ning, P. , Reeves, D. S. and Wang, X. 2006. Active Timing Correlation of Traffic Flows with Chaff Packets.

239. Peng, F., Schuurmans, D. and Keselj, V. and Wang, S. 2003a. Language independent authorship attribution using character level language models. In Proceedings of the Tenth Conference of the European Chapter of the Association for Computational Linguistics.

240. Pennebaker, J. W. and Stone, L. D. 2003. Words of wisdom: Language use over the life span. *Journal of Personality and Social Psychology*, Vol. 85, No. 2, pp. 291–301.

241. Peng, F., Schuurmans, D., Keselj, V., and Wang, S. 2003b. Automated authorship attribution with character level language models. In Proceedings of the Tenth Conference of the European Chapter of the Association for Computational Linguistics, pp. 267–274.

242. Peng, F., Schuurmans, D. and Wang, S. 2004, Augumenting Naive Bayes Text Classifier with Statistical Language Models. *Information Retrieval*, Vol. 7 , No. 3–4, pp. 317–345.

243. Peng, F., Shuurmans, D., Keselj, V. and Wang, S. 2003. Language independent authorship attribution using character level language models. In Proceedings of the Tenth Conference of the European Chapter of the Association for Computational Linguistics, Morristown, NJ, pp. 267–274.

244. Pennebaker, J. W. and King, L. A. 1999, Linguistic Styles: Language Use as an Individual Difference. *Journal of Personality and Social Psychology*, Vol. 77 , No. 6, pp. 1296–1312.

245. Pennebaker, J. W., Mehl, M. R. and Niederhoffer, K. 2003. Psychological aspects of natural language use: Our words, our selves. *Annual Review of Psychology*, Vol. 54, pp. 547–577.

246. Pham, D. D., Tran, G. B. and Pham, S. B. 2009. Author Profiling for Vietnamese Blogs. In Proceedings of the International Conference on Asian Language Processing, pp. 190–194.

247. Pham, V. H. and Dacier, M. 2009. Honeypot traces forensics : the observation view point matters. In Proceedings of the Third International Conference on Network and System Security,Gold Cost, Australia.

248. Popov, I. V. , Debray, S. K. and Andrews, G. R. 2007. Binary Obfuscation Using Signals. In Proceedings of USENIX Security Symposium, Boston, MA, USA.

249. Prechelt, L., Malpohl, G. and Philippsen, M. 2002. Finding plagiarisms among a set of programs with JPlag. *Journal of Universal Computer Science*, Vol. 8, No. 11, pp. 1016–1038.

250. Pyun, Y., Park, Y., Wang X. and Reeves, D. S. 2006. Tracing traffic through intermediate hosts that repacketize flows. In Proceedings of IEEE Conference on Computer Communications, pp. 634–642.

251. Quintanila, C. A., Lee, M., Lee, S. and Skinner, C. S. 2004. Method of tracking and authenticating emails. United States Patent Application 20050188077A1.

252. Rao, J. R. and Rohatgi, P. 2000. Can pseudonymity really guarantee privacy? In Proceedings of the Ninth USENIX Security Symposium, pp. 85–96. USENIX.

253. Ravikumar, P. and Cohen, W. W. 2004. A Hierarchical Graphical Model for Record Linkage. In Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence (UAI '04), Banff Park Lodge, Banff, Canada.

254. Reddy, K. S., Dash, S. K. and Pujari, A K. 2006. New Malicious Code Detection Using Variable Length n-grams. In Bagchi, A. and Atluri, V. (Eds): ICISS 2006, *Lecture Notes in Computer Science*, Vol. 4332, pp. 276–288.

255. Ressler, R. K., Burgess, A. W., Hartman, C. R., Douglas, J. E. and McCormack, A. 1986. Murderers who rape and mutilate. *Journal of Interpersonal Violence*, Vol. 1, pp. 273–287.

256. Saferstein, R. 2001. *Criminalistics: An introduction to forensk science* (7th ed). Pearson Prentice Hall.

257. Sager, G 1998. Security Fun with OCxmon and cflowd. Internet-2 Measurement Working Group.

258. Salton, G. and McGill, M. J. 1983. *An Introduction to Modern Information Retrieval*. McGraw-Hill, 1983.

259. Sanchez, L. A., Milliken, W. C., Snoeren, A. C., Tchakountio, F., Jones, C. E., Kent, S. T., Partrige, C. and Strayer, W. T. 2001. Hardware Support for a Hash-Based IP Traceback. In Proceedings of the Second DARPA Information Survivability Conference and Exposition (DISCEX II), Anaheim, CA, US, pp. 146–152.

260. Santilla, P., Hakkanen, H., Alison, L. and Whyte, C. 2003. Juvenile firesetters: Crime scene actions and offender characteristics. *Legal and Criminological Psychology*, Vol. 8, pp. 1–20.

261. Savage, S., Wetherall, D., Karlin, A. and Anderson, T. 2000. Practical Network Support for IP Traceback. In Proceedings of the 2000 ACM SIGCOMM Conference, Stockholm, Sweden, pp. 295–306.

262. Savage, S., Wetherall, D., Karlin, A. and Anderson, T. 2001. Network Support for IP Traceback. *IEEE/ACM Transactions on Networking (TON)*, Vol. 9, No. 3.

263. Sebastiani, F. 2002. Machine Learning in Automated Text Categorization. *ACM Computing Surveys*, Vol. 34, No. 1, pp. 1–47.

264. Schler, J., Koppel, M., Argamon, S. and Pennebaker, J. 2006. Effects of Age and Gender on Blogging. In Proceedings of AAAI Spring Symposium on Computational Approaches for Analyzing Weblogs.

265. Information Warfare Monitor and Shadowserver Foundation 2010. Shadows in the cloud: Investigating Cyber Espionage 2.0.`http://shadows-in-the-cloud.net`.

266. Shevertalov, M., Kothari, J., Stehle, E. and Mancoridis, S. 2009. On the use of discretised source code metrics for author identification. In Proceedings of the First International Symposium on Search Based Software Engineering, Windsor, United Kingdom, pp. 69–78.

267. Skarda, B. E. 2008. Operationalizing Offensive Social Engineering for the Air Force. Master Thesis. Air Force Institute of Technology, Air University, Ohio, USA.

268. Slade, R. M. 2004. *Software Forensics: Collecting Evidence from the Scene of a Digital Crime*. McGraw Hill Professional.

269. Smith, R. M. 2000. Microsoft Word Documents that Phone Home. Privacy Foundation.

270. Snoeren, A. C., Partridge, C., Sanchez, L. A., Jones, C. E., Tchakountio, F., Kent, S. T. and Strayer, W. T. 2001. Hash-Based IP Traceback. `http://www.acm.org/sigcomm/sigcomm2001/p1.html`.

271. Song, D. X., and Perrig, A. 2001. Advanced and authenticated marking schemes for IP traceback. In Proceedings of the Conference on Computer Communications.

272. Spafford, E. H. 1989. The internet worm program: An analysis. *ACM SIGCOMM Computer Communication Review*, Vol. 19, No. 1, pp. 17–57.

273. Spafford, E. H. and Weeber, S. A., 1993. Software forensics: tracking code to its authors. *Computers and Security*, Vol. 12, pp. 585–595.

274. Spinellis, D. 2009. Job security. *IEEE Software*, Vol. 26, No. 5, pp. 14–15.

275. Spitzner, L. 2003a. Honeypots: Catching the insider threat. In Proceedings of the Ninteenth Annual Computer Security Applications Conference (ACSAC), pp. 170–179.

276. Spitzner, L. 2003b. Honeytokens: The other honeypot. Technical report, SecurityFocus.

277. Stamatatos, E., Fakotakis, N. and Kokkinakis, G. 2000. Automatic text categorization in terms of genre and author. *Computational Linguistics*, Vol. 26, No. 4, pp. 471–495.

278. Stamatatos, E., Fakotakis, N. and Kokkinakis, G. 2001. Computer-based authorship attribution without lexical measures. *Computers and the Humanities*, Vol. 35, No. 2, pp. 193–214.

279. Stamatatos, E. 2009. A Survey of Modern Authorship Attribution Methods, *Journal of the American Society for information Science and Technology*, Vol. 60, No. 3, pp. 538–556.

280. Staniford-Chen, S. G. 1995. Distributed Tracing of Intruders. Master's Thesis, University of California.

281. Staniford-Chen, S. and Heberlein, L. 1995. Holding Intruders Accountable on the Internet. In Proceedings of the 1995 IEEE Symposium on Security and Privacy (SandP), Oakland, CA, US, pp. 39–49.

282. Stein, B. and Eissen, S. M. 2007. Intrinsic plagiarism analysis with meta learning. In Proceedings of the First International Workshop on Plagiarism Analysis, Authorship Identification, and Near-Duplicate Detection, , Amsterdam, Netherlands, pp. 45–50.

283. Sterne, D., Djahandari, K., Wilson, B., Babson, B., Schnackenberg, D., Holliday, H. and Reid, T. 2001. Autonomic Response to Distributed Denial of Service Attacks. *Lecture Notes in Computer Science* (LNCS), Vol. 2212, pp. 134–149.

284. Stone-Gross, B., Moser, A., Kruegel, C., Almaroth, K. and Kirda, E. 2009. FIRE: FInding Rogue nEtworks, 25th Annual Computer Security Applications Conference (ACSAC), Honolulu.

285. Symantec 2009. The Symantec Public Report on Rogue Security Software.

286. Symantec 2010. The Nature of Cyber Espinonage: Most Malicious File Types Identified and Encrypted Spam from Rustock. MessageLab Intelligence.

287. Tang, J., Li, H., Cao, Y. and Tang, Z. 2005. Email Data Cleaning. In Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD'2005), Chicago, Illinois, USA, pp. 489–499.

288. Thonnard, O. and Dacier, M. 2008. Actionable knowledge discovery for threats intelligence support using a multi-dimensional data mining methodology. In Proceedings of the Eight IEEE International Conference on Data Mining series, Pisa, Italy.

289. Thonnard, O., Mees, W. and Dacier, M. 2009. Addressing the attack attribution problem using knowledge discovery and multi-criteria fuzzy decision-making. In Proceedings of the Fifteen ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Workshop on CyberSecurity and Intelligence Informatics, 2009, Paris, France , pp. 11–21.

290. Thomson, R. and Murachver, T. 2001. Predicting gender from electronic discourse. *British Journal of Social Psychology*, Vol. 40:, pp. 193–208.

291. Turvey, B. E. 2008. *Criminal Profiling: An Introduction to Behavioral Evidence Analysis*. Elsevier.

292. Tweedie, F. J., Singh, S. and Holmes, D. I. 1996. Neural Network applications in stylometry: The Federalist Papers. *Computers and the Humanities*, Vol. 30, No. 1.

293. Ukkonen, E. 1985. Finding approximate patterns in strings. *Journal of Algorithms*, Vol. 6, No. 1, pp. 132–137.

294. Uzuner, O. and Katz, B. 2005. A comparative study of language models for book and author recognition. In Proceedings of the Second International Joint Conference on Natural Language Processing, Berlin, Germany: Springer, pp. 969–980.

295. Vamplew, P. and Dermoudy, J. 2005. An anti-plagiarism editor for software development courses. In Proceedings of the Seventh Australasian Conference on Computing Education, pp. 83 90, Newcastle, Australia. Australian Computer Society.

296. Vrable, M., Ma, J., Chen, J., Moore, D., Vandekieft, E., Snoeren, A. C., Voelker, G. M. and Savage, S. 2005. Scalability, fidelity, and containment in the potemkin virtual honeyfarm. *SIGOPS Operating Systems Review*, Vol. 39, No. 5, pp. 148–162.

297. Wang, G., Chen, H. and Atabakhsh, H. 2004. Criminal Identity Deception and Deception Detection in Law Enforcement. *Group Decision and Negotiation*, Vol. 13, pp. 111–127.

298. Wang, G., Chen, H. and Atabakhsh, H. 2006. A Multi-layer Naive Bayes Model for Approximate Identity Matching. *Lecture Notes in Computer Science*, Vol. 3975, pp. 479–484.

299. Wang, G., Kaza, S., Joshi, S. and Chang, K. 2007. The Arizona IDMatcher: Developing an Identity Matching Tool for Law Enforcement. In Proceedings of the 8th Annual International Digital Government Research Conference, Philadelphia, USA, pp. 304–305.

300. Wang, X., Reeves, D. S. and Wu, S. F. 2002. Inter-Packet Delay Based Correlation for Tracing Encrypted Connections Through Stepping Stones. In Proceedings of the Seventh European Symposium on Research in Computer Security (ESORICS).

301. Wang, X. and Reeves, D. S. 2002. Robust Correlation of Encrypted Attack Traffic Through Stepping Stones by Manipulation of Inter-packet Delays. In Proceedings of the 2003 ACM Conference on Computer and Communications Security (CCS), pp. 20–29.

302. Wang, X. , Reeves, D. S. , Wu, S. F. and Yuill, J. 2001. Sleepy Watermark Tracing: An Active Network-Based Intrusion Response Framework. In Proceedings of 16th International Conference on Information Security (IFIP/Sec01).

303. Watanbe, S. 1985. *Pattern Recognition: Human and Mechanichal*. John Wiley, New York.

304. Webb, A. 2002. *Statistical Pattern Recognition*. John Wiley, New York.

305. Weber, H. 1812. *The Works of Beaumont and Fletcher*. Edinburgh.

306. Wei, Ch., Sprague, A., Warner, G. and Skjellum, A. 2008. Mining spam email to identify common origins for forensic application. In Proceedings of the 2008 ACM symposium on Applied computing, New York, USA.

307. Wei, C., Sprague, A., Warner, G. and Skjellum, A. 2009. Characterization of spam advertised website hosting strategy. In Proceedings of Sixth Conference on Email and Anti-Spam, California, US.

308. Wheeler, D. A. and Larsen, G. N. 2003. Techniques for cyber attack attribution. Technical Report, Institute for Defense Analysis, pp. 37–92.

309. Williams, C. B. 1940. A Note on the Statistical Analysis of Sentence-length as a Criterion of Literary Style. *Biometrika*, Vol. 31, pp. 363–390.

310. Willems, C., Holz, T. and Freiling, F. 2007. Toward Automated Dynamic Malware Analysis Using CWSandbox. *IEEE Security and Privacy Magazine*, Vol. 5, No. 2, pp. 32–39.

311. Witten, I. H., Frank, E. and Hall, M. A. 2011. *Data mining: practical machine learning tools and techniques*. Morgan Kaufmann.

312. Yetiser, T. 1993. Polymorphic Viruses - Implementation, Detection, and Protection. `http://vx.netlux.org/lib/ayt01.html`.

313. Yoda, K. and Etoh, H. 2000. Finding a Connection Chain for Tracing Intruders. In Proceedings of the Sixth European Symposium on Research in Computer Security (ESORICS).

314. Yule, G. U. 1939. On Sentence-length as a Statistical Characteristic of Style in Prose, with Application to Two Cases of Disputed Authorship. *Biometrika*, Vol. 31, pp. 356–361.

315. Yule, G. U. 1944. The Statistical Study of Literary Vocabulary. *Cambridge University Press*.

316. Yuill, J., Denning, D. and Feer, F. 2006. Using deception to hide things from hackers : Processes, principles, and techniques. *Journal of Information Warfare*, Vol. 5, No. 3, pp. 26–40.

317. Zhao, Y. and Zobel, J. 2007. Searching with style: Authorship attribution in classic literature. In Proceedings of the 30th Australasian Computer Science Conference, New York: ACM Press, pp. 59–68.

318. Zax, D. 2011. *You can't keep your secrets from Twitter*. FastCompany. `http://YouCan'tKeepYourSecretsFromTwitter`.

319. Zhang, C., Chen, W.-B., Chen, X and Warner, G. 2009. Revealing common sources of image spam by unsupervised clustering with visual features. In Proceedings of the 2009 ACM symposium on Applied Computing, New York, NY, USA.

320. Zhang,Y. and Paxson, V. 2000. Detecting Stepping Stones. In Proceedings of the Ninth USENIX Security Symposium, pp. 171–184.

321. Zhang, L. , Persaud, A., Johson, A. and Guan, Y. 2006. Stepping Stone Attack Attribution in Non-cooperative IP Networks. In Proceedings of the 25th IEEE International Performance Computing and Communications Conference (IPCCC 2006), Phoenix, AZ.

322. Zheng, R., Li, J., Chen, H. and Huang, Z. 2006. A Framework for Authorship Identification of Online Messages: Writing-Style Features and Classification Techniques. Journal of the American Society for Information Science and Technology , Vol. 57, No. 3, pp. 378–393.

323. Zheng, R., Qin, Y., Huang, Z., and Chen, H. 2003. Authorship analysis in cybercrime Investigation. Proceedings of the 1st NSF/NIJ Symposium, ISI2003 (pp. 5973). Springer-Verlag.

324. Zhou, C. V., Leckie, C., Karunasekera, S. and Peng, T. 2008. A self-healing, self-protecting collaborative intrusion detection architecture to trace-back fast-flux phishing domains. In Proceedings of the Second IEEE Workshop on Autonomic Communication and Network Management (ACNM 2008).

DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION DOCUMENT CONTROL DATA	1. CAVEAT/PRIVACY MARKING

2. TITLE	3. SECURITY CLASSIFICATION
Attribution of Spear Phishing Attacks: A Literature Survey	Document (U) Title (U) Abstract (U)
4. AUTHOR	5. CORPORATE AUTHOR
Van Nguyen	Defence Science and Technology Organisation PO Box 1500 Edinburgh, South Australia 5111, Australia

6a. DSTO NUMBER	6b. AR NUMBER	6c. TYPE OF REPORT	7. DOCUMENT DATE
DSTO–TR–2865	015-662	Technical Report	August, 2013

8. FILE NUMBER	9. TASK NUMBER	10. TASK SPONSOR	11. No. OF PAGES	12. No. OF REFS
2012/1105154/1	N/A	N/A	154	324

13. URL OF ELECTRONIC VERSION	14. RELEASE AUTHORITY
`http://www.dsto.defence.gov.au/publications/scientific.php`	Chief, Cyber and Electronic Warfare Division

15. SECONDARY RELEASE STATEMENT OF THIS DOCUMENT

Approved for Public Release

OVERSEAS ENQUIRIES OUTSIDE STATED LIMITATIONS SHOULD BE REFERRED THROUGH DOCUMENT EXCHANGE, PO BOX 1500, EDINBURGH, SOUTH AUSTRALIA 5111

16. DELIBERATE ANNOUNCEMENT

No Limitations

17. CITATION IN OTHER DOCUMENTS

No Limitations

18. DSTO RESEARCH LIBRARY THESAURUS

spear phishing, attribution, email and website authorship, malware authorship, adversary identification and characterisation

19. ABSTRACT

Spear phishing involves the use of social engineering and contextual information to entice a targeted victim into unwitting leakage of sensitive information for purposes of identity crime or espionage. The high success rate together with the potential scale of damage caused by spear phishing attacks has motivated cyber researchers and practitioners to investigate more effective and strategic defensive, deterrent and offensive mechanisms against spear phishers. Obviously, the practicability of any such defence mechanism depends on the extent to which a defender has knowledge of the adversary behind a spear phishing attack. This necessitates the defending party to perform attribution in order to identify the spear phisher and/or his/her accomplice. In this survey, I broadly define attribution of spear phishing as any attempt to infer the identity or characteristics of the source of the attack which may include a machine, a human individual, or an organisation. Though highly desirable, this attribution mission is a very challenging task. This survey represents an initial step in this direction. Ultimately, the survey aims to sketch the landscape of attribution methods pertaining to spear phishing, as well as to provide constructive remarks and relevant recommendations for an organisation wishing to perform this attribution mission.