available at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/cose

Computers & Security

DNS-based email sender authentication mechanisms: A critical review

Amir Herzberg

Bar Ilan University, Computer Science Department, Ramat Gan 52900, Israel

ARTICLE INFO

Article history:
Received 21 January 2009
Received in revised form
9 April 2009
Accepted 1 May 2009

Key words:
Email
Spam
Phishing
SPF
DKIM
Sender-ID
SIDF
SMTP
Internet security

ABSTRACT

We describe and compare three predominant email sender authentication mechanisms based on DNS: SPF, DKIM and Sender-ID Framework (SIDF). These mechanisms are designed mainly to assist in filtering of undesirable email messages, in particular spam and phishing emails. We clarify the limitations of these mechanisms, identify risks, and make recommendations. In particular, we argue that, properly used, SPF and DKIM can both help improve the efficiency and accuracy of email filtering.

1. Introduction

The Internet facilitates efficient, low-cost communication worldwide, allowing universal connectivity for many purposes, and providing great value to society and users. In particular, Internet electronic mail (*email*) has extremely low per-message cost, especially when sending 'bulk' mail (to many recipients).

Unfortunately, the design of many basic Internet protocols often did not consider abusive, malicious use of the protocols. This especially holds for the email system, whose basic design is quite vulnerable. Currently, the most significant threats involving email users appear to be *spam*, *phishing* and *denial of service*:

Spam: there are different definitions of spam; we use the term *spam* to refer to email sent to many recipients ('bulk'), without appropriate label(s) allowing quick filtering by receivers (or receiving domains) not desiring to receive email of this class. Spam annoys users, wastes recipient's time and network resources, and reduces the reliability and the functionality of email, but serves the interests of the spammer. Spammers often abuse computers and/or email accounts of unaware users, e.g. using *spambot* malware. The costs in time and resources spent on spam, are probably much higher than the revenues to the spammers; however, as long as almost all of these costs are on users and operators but not on spammers, and costs for spammers are below spammer revenues, spamming will remain profitable and common.

E-mail address: herzbea@cs.biu.ac.il
0167-4048/$ – see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cose.2009.05.002

Phishing: phishing emails are intended to mislead the user, to believe they came from a trusted organization, typically a financial institution, while actually originating from the attacker. Phishing emails can cause significant harm, such as disclosure of credentials (e.g. password) or installation of malware (e.g. virus).
Denial of Service (DoS): DoS emails are sent with the intention to cause harm to the operation of the email system of the recipient and/or of a third party. The most common email DoS attack is by *clogging*, i.e. sending many emails with the goal of causing excessive overhead, to the recipient and/or to a third party. In particular, a *Joe job attack* is a special form of clogging, which sends many emails with incorrect, spoofed sender address, with the goal of causing the recipients to send error ('bounce') messages to the spoofed sender address, thereby overloading it with processing of these bounces.

There are many techniques used to defend against these (and other) email threats. We focus on *DNS-based email sender authentication mechanisms*. These mechanisms use the DNS system (in different ways), to identify the sender – i.e., the user sending the message and/or the domain from which the message originated. Email authentication mechanisms allow receiving mail agents to accept mail from known good senders, reject mail from known bad senders (e.g. spammers), or use reputation mechanisms such as blacklists and greylists (Harris, 2003) to decide how to handle mail from other senders. Many email systems employ email authentication, often in combination with reputation and classification mechanisms, to filter out spam, phishing and other abusive email, using architectures as shown e.g. in Herzberg (2009) and Leiba and Borenstein (2004).

There is a rather large number of proposals, mechanisms and standards for email sender authentication, beginning from the very early days of email (Thomas, 1974). Few proposals aim at providing general security mechanisms, including confidentiality (encryption) and long-term authentication (signatures), in particular S/MIME (Ramsdell, 2004) and OpenPGP (Elkins et al., 2001); other proposals focus on specific aspects such as validation of bounce messages, e.g. BATV (Crocker, 2005). Some solutions also involve extensions to the email protocols, e.g. the SMTP Auth extension (Siemborski and Melnikov, 2007). A complete, in-depth survey of all these mechanisms is beyond the scope of this article.

Instead, we focus on three predominant email sender authentication proposals, whose main motivation is to identify and block spam, phishing and other abusive emails, and that are all based on the Domain Name System (DNS):

SPF: The Sender Policy Framework (SPF) (Wong and Schlitt, 2006) that validates that the domain indicated in the mail from address, has indicated that the IP address used by the sending mail agent, is (or is not) allowed to send email with this mail from address. Since the mail from address is not visible to the recipient, it follows that SPF does not prevent display of spoofed sender address.
SIDF: The Sender-ID Framework (SIDF) (Lyon and Wong, 2006) that attempts to extend SPF by authenticating (also) recipient-visible sender addresses. SIDF, like SPF, is also based on security of routing and DNS.
DKIM: The DomainKeys Identified Mail (DKIM) design (Allman et al., 2007; Lieba and Fenton, 2007) allows a mail agent to authenticate email by validating a digital signature over the email, computed by some signer, typically a sending domain assuming responsibility and identifying itself as the origin of the message. DKIM signatures are computed over the message itself and over some headers fields, such as the FROM email message header (which is mandatory to sign); this allows recipients to detect unauthorized modification of the message or the signed header fields, from the values sent with the message by the signer. Notice that SIDF also shared the goal of ensuring integrity of header fields which the receiving MUA may present to the user to identify the message origin or sender.

When discussing each of the three proposed DNS-based email authentication mechanisms, we present usage recommendations and explain the security properties of the mechanisms, and clear up some possible misconceptions and unjustified expectations. The following is a summary of our main observations and recommendations:

SIDF considered harmful. We argue that SIDF does not deliver its promised added value over SPF, and on the contrary, it has significant drawbacks. We therefore recommend to avoid using it.
Use SPF and DKIM to efficiently identify legitimate emails. We argue that all three mechanisms can be useful to avoid 'false positives', i.e. prevent blocking of legitimate emails from known, trusted senders. This can be especially useful when the sender changes to a new outgoing mail server address. There is also a side-benefit: eliminating unnecessary filtering, esp. the computationally-intensive and imprecise content-based filtering, for properly authenticated emails (using SPF and/or DKIM).
Do not block email based only on SPF/SIDF/DKIM failure. Many senders may not use SPF/DKIM. Furthermore, even email from senders using SPF/DKIM, may fail to pass validation due to mangling. Such failures should therefore be used only as indicators, together with other indications such as content filtering, e.g. as in Herzberg (2009).
Senders should use both SPF and DKIM. We recommend to use *both* SPF and DKIM, since either one of them may be 'mangled' as email passes between servers.
Email envelope should signal use of SPF. Since often recipients may utilize SPF only to identify legitimate emails (avoid false positives and filtering), it is recommended that senders would mark their use of SPF in the email envelope, to ensure recipients are aware of the availability of the SPF records. This requires adoption of some standard signaling mechanism, e.g. using an EHLO SMTP service extension (Klensin, 2008).
Use DNS security if possible. All three mechanisms rely on the security of the DNS system, which currently is rather weak. DNS security can be improved using DNS security (Arends et al., 2005).
Limit DNS queries per SPF validation, use separate server. SPF policies may be abused as a part of a DNS poisoning attacks, such as described by Kaminsky (2008), or for Denial of Service (DoS) attacks. To address this threat, we recommend that incoming mail servers using SPF will use a separate DNS proxy

server, and severely limit their rate of DNS queries and the total number of DNS queries per SPF validation (even more than the limits recommended by Wong and Schlitt (2006) (see Otis (2006)).

Note that our scope does not include review of, or comparison to, other email authentication mechanisms and standards, e.g. S/MIME (Ramsdell, 2004) and OpenPGP (Callas et al., 2007). Specifically, S/MIME and OpenPGP are designed as a security service for the end user, often not only authenticating the sender to the recipient but also providing encryption (for confidentiality) and digital signatures (allowing the recipient to convince a third party of the identity of the sender).

Organization. In the next section, we present brief background on relevant Internet protocols (IP, DNS and SMTP), necessary to understand email authentication mechanisms, and explain the relevant adversary models, vulnerabilities and abuses. In the following sections, we discuss three proposed standards for providing (different forms of) authentication of senders of email: Sender Policy Framework (SPF) (Wong and Schlitt, 2006) in Section 3, Sender-ID Framework (SIDF) (Lyon and Wong, 2006) in Section 4, and DomainKeys Identified Mail (DKIM) (Allman et al., 2007; Lieba and Fenton, 2007) in Section 5.

2. Background: Internet Protocols (IP, DNS, SMTP) and Email Vulnerabilities

In this section we provide brief relevant background on Internet protocols which are highly relevant to email security, mainly IP, DNS and SMTP. We also explain the main adversary models: spoofing (blind) adversary, eavesdropping adversary, Man In The Middle (MITM) adversary, client-only adversary and DNS-poisoning adversary. Finally, we discuss email vulnerabilities and potential abuses.

2.1. The Internet Protocol (IP) and basic adversary models: MITM, spoofing and eavesdropping

The Internet is a loose collection of networks, connected by routers. Communication between hosts (computers) on different networks in the Internet, is done by sending Internet Protocol (IP) packets. The packets traverse from the sending host to a router connected directly (over same local network) to the sending host, and from it, via other routers, till reaching the destination. Routers run routing protocols to determine the best route (sequence of routers) from sending host to destination.

IP packets contain two *IP addresses*, identifying the purported sender (source) of the packet, and the intended destination. IP addresses are assigned hierarchically, to service providers, organizations and ultimately individual computers (users). This hierarchical IP Address assignment is usually done in 'blocks', where the IP addresses in each block have a common left-hand prefix, allowing efficient routing based on the left-hand prefix of the destination IP address.

The Internet is a huge, decentralized network, with many vulnerabilities, and subject to a wide range of attacks and threats. In particular, attackers often send packets with *spoofed/fake* source IP address, i.e. a packet purportedly originating from a certain IP address, while in reality it is sent by an attacker, who was not assigned this address. Many Internet Service Providers (ISPs) filter packets their customers send to the Internet, and block any that are using incorrect source address (*ingress filtering*, see Ferguson and Senie (2000)), however, some providers do not do this. As a result, it is relatively common for attackers to be able to send spoofed packets, i.e. IP packets with fake source address. We call such attackers (IP) *spoofing adversaries*.

It is less common that an attacker has the ability to *intercept* packets sent to an IP address which does not belong to the attacker. This is due to the fact that packets are sent by the routers using the route they consider best toward the destination (based on the destination IP address). Therefore, to intercept a packet whose destination IP address is IP_{victim}, the attacker must either control a router or a computer on the 'best' route to IP_{victim}, or be able to disrupt the routing mechanisms so that the packet is routed via a router controlled by the attacker, rather than routed via the 'best' route. Such attacks are possible (Bellovin, 1989), but are quite challenging and not very common.

We refer to such an attacker with the ability to intercept packets as a *Man In The Middle (MITM) adversary*. More precisely, an MITM adversary is able to both intercept packets with IP_{victim} as destination address, and to spoof packets with arbitrary sender IP address. In practice, it is much more common for attackers to have spoofing capabilities then to have to have MITM capabilities. Tunneling may be used to identify and discard spoofed packets sent by a spoofing-but-not-MITM adversary, e.g. see Gilad and Herzberg (2009), however such mechanisms are not widely deployed so far.

Sometimes, attackers may be able to expose the content of packets directly by eavesdropping on the communication between hosts and/or routers, e.g. by electromagnetic emanation. For example, this may be possible when using (unencrypted) wireless local area network. An attacker which is only able to eavesdrop (and not to inject spoofed packets) is called an *eavesdropping adversary*. Of course, if an attacker can both eavesdrop and inject spoofed packets, then it is an MITM adversary.

2.2. The Domain Name System (DNS) and DNS Poisoning Adversaries

IP addresses are structured and allocated to optimize their use in routing packets, and hence are not mnemonic. Therefore, users and applications rarely use IP addresses directly; instead, they use more user-friendly and meaningful *domain names*. Domain names are formed as a sequence of alphanumeric strings separated by dots, e.g. www.foo.org. If α is a domain name and β is an alphanumeric string, then $\beta.\alpha$ is a domain name, which is a *subdomain* of α. The subdomain to domain relationship reflects organizational relationships, rather than routing considerations (as for IP addresses). The rightmost string is called *top level domain (TLD)*, and it is from a well-defined set (COM, NET, ORG, UK, IL,…). Top level domains are managed by *domain registrars*, assigned by the Internet Assigned Numbers Authority (IANA).

A domain manager (or 'owner') can assign subdomains, to parts of its organization or to other organizations. Domain names are often related to trademarks and common names for organizations and individuals. Domain registrars, and also courts of law, often prevent the use of domain names which conflict with intellectual property or trademark rights, or that may otherwise mislead users.

The basic task of the *domain name system (DNS)* is to *resolve* (map) domain names into IP addresses, e.g. www.foo.org may resolve to 132.70.6.252. Domain owners specify the mapping of their domains and subdomains in *authoritative DNS Servers*; namely, the authoritative name server for a domain knows and provides the IP address of names in the domain; for addresses in subdomains of the domain, it may also provide the address of an appropriate authoritative server. Therefore, the authoritative domain name servers form a hierarchy; at the top is a set of *root DNS servers*, in thirteen fixed, well-known IP addresses. The root DNS servers know and provide the IP addresses of the top level domain (TLD) authoritative servers, which in turn know and provide the IP addresses of authoritative name servers for each of their subdomains. DNS mappings are sent in special format, called DNS resource records (RR), containing name, value, type and expiration time.

Each computer connected to the Internet runs a DNS client, that has the IP address of at least one *local DNS server*. The DNS client uses the local DNS server to resolve domain names. The local DNS servers know the addresses of the root servers; each local DNS server also maintains a cache of DNS records it received, until their expiration time. If the required record (mapping) is not in the cache or has expired, then the local DNS server will send a request for the record to other DNS servers.

The caching and hierarchical organization of the DNS provide an efficient, decentralized, highly available directory service, for several types of records, and with multiple applications. The basic resource record is type A (or AAAA for IPv6), used to map domain names to IP addresses. Other records related to email spam and phishing prevention include:

MX (incoming mail server) records: the MX record of a domain contains the address of the *incoming* Simple Mail Transfer Protocol (SMTP) server of that domain. An SMTP-Sender (server sending email) uses this query to locate the SMTP-Receiver (server receiving email).

Reverse DNS (rDNS) records: reverse DNS (rDNS) records are associated with an IP address. The rDNS record of an IP address identifies a domain 'owning' that IP address (using the PTR DNS record type). When an organization, such as a corporation or Internet Service Provider (ISP), receives authority for a range of IP addresses, it normally receives also the ownership of the corresponding rDNS domain, allowing it to assign domain names to IP addresses in this range. Notice that some ISPs may provide IP addresses to customers (higher-tier ISPs, corporations or individuals), without giving them the ability to administer the corresponding rDNS entries.

DNS Blacklist records: DNS blacklist records are 'regular' DNS records, e.g. of type A, associated with subdomains of a 'blacklist domain', e.g. ExampleBL.org. Such blacklist DNS record (subdomain of ExampleBL.org) is defined for 'blacklisted' IP addresses, blocks of IP addresses or domain names. Typically, upon connection from SMTP-Sender with IP address aa.bb.cc.dd, the SMTP-Receiver makes a DNS request (usually for record type A, i.e. IP address) of the specially crafted domain name dd.cc.bb.aa.ExampleBL.org. By reversing the order of the bytes of the IP address, we allow a response for a 'higher level domain' e.g. cc.bb.aa.ExampleBL.org; such response can indicate that the entire IP address block aa.bb.cc.0/24 is blacklisted. The value returned (as an IP address) from the resolution can be interpreted as additional information, e.g. the cause of blacklisting.

Policy DNS records: domains can publish different policies in DNS records of different types, typically TXT or policy-specific types e.g. SPF. Policy DNS records are used in SPF and SIDF to identify authorized *outgoing* mail servers (in contrast to the MX records for incoming servers); see Sections 3 and 4. DKIM also uses a domain policy record, called the *author domain signing practices (ADSP)* policy (Allman et al., 2009), to specify whether a domain attaches DKIM signatures to all its outgoing emails; see Section 5. Domain policy records (and public key records, described next) may use the 'general-purpose' text record type (TXT), or dedicated record types (e.g. SPF).

Public key records: public key records publish the public keys of a domain, used to validate signatures by the domain owner. Public key records are used in DKIM, see Section 5.

DNS Poisoning. The DNS provides an open directory service, without attempting to restrict access to the information, therefore confidentiality is usually not a concern. However, DNS is a critical service, hence its integrity and availability are crucial. DNS is designed to ensure integrity and high availability in spite of failures, but may not resist a determined, malicious and powerful attacker.

In particular, standard DNS records are distributed without any modification detection mechanism, such as a digital signature (unless using Secure DNS (Arends et al., 2005)). As a result, a Man In The Middle (MITM) adversary can easily provide incorrect responses (resolutions) to a DNS query. The result is an incorrect resolution (typically of domain name to IP address), often cached by DNS servers and/or clients; we refer to this result as *DNS poisoning*.

In principle, it should be impossible for an IP spoofing adversary (without MITM capabilities), to poison the DNS record of domain www.foo.org kept by some DNS client or server. This is provided, of course, that the adversary does not control one of the relevant authoritative DNS servers: the root DNS servers, the org DNS servers or the foo.org DNS servers. However, due to protocol and implementation errors, often, determined attackers can cause DNS poisoning even without MITM capabilities (and without controlling a higher level name server). One common technique is by sending spoofed DNS responses (without intercepting the corresponding query). Many of these attacks exploit the fact that DNS typically runs without a connection (i.e. over UDP rather than TCP), and often implementations do not take sufficient measures to identify correct responses vs. spoofed responses. For example, many implementations rely only on a 16-bit identifier field copied from DNS query to DNS response, which may not suffice to foil a determined spoofing adversary; see more details e.g. in Bellovin (1989) and Kaminsky (2008).

We refer to an adversary which can poison DNS records (typically via IP spoofing), as *DNS poisoning adversary*. A DNS poisoning adversary may often have similar capabilities to an MITM adversary, with respect to applications that rely on DNS to identify servers, such as email (SMTP). Specifically, by spoofing the MX record, a DNS poisoning adversary can capture SMTP connections destined to some victim server; and by spoofing a domain policy record or a public key record, the adversary can circumvent DNS-based policy mechanisms such as these used by SPF, SIDF and DKIM, or send a fake public key, e.g. to circumvent public key signature validation by DKIM.

Secure DNS (Arends et al., 2005) is an enhancement to DNS, where responses are digitally signed, thereby preventing DNS poisoning by MITM and weaker adversaries. However, secure DNS has significant performance penalty, and is not yet universally deployed.

2.3. *The email system and its vulnerabilities*

Users compose, read and manage their email using an interface module usually referred to as the *Mail User Agent (MUA)*. The MUA may be a program running on a multi-user computer (such as Unix mail), a dedicated mail client (such as Outlook or Thunderbird), or simply a web client (browser).

Mail User Agents are usually invoked as needed by the user, and do not run continuously. Hence, email is not sent directly from the sender's MUA, say MUA_{Alice} to the recipient's MUA, say MUA_{Bob}. Instead, MUA_{Alice} sends the email to Bob's *Mail Delivery Agent* (MDA_{Bob}), which delivers it to Bob's *mailbox* until MUA_{Bob} picks it up.

Bob's MUA, MUA_{Bob}, picks up email placed in the mailbox by Bob's MDA. When the MUA is an application running on the same machine as the MDA, mail pickup may be done via the file system. More often, the MUA runs on a separate machine, and picks up mail either via an appropriate *web-mail* form or via a *mail pickup protocol*, typically either the *Post Office Protocol (POP)* (Myers and Rose, 1996) or the *Internet Message Access Protocol (IMAP)* (Crispin, 2003).

A different protocol, the *Simple Mail Transfer Protocol (SMTP)* (Klensin, 2008), transfers the email from the sending MUA, e.g. MUA_{Alice}, to the recipient's MDA, e.g. MDA_{Bob}, typically via *Mail Transfer Agents (MTA)*. Typically, the email first flows through MTAs in the sender's domain, say *a.com*, until it reaches the *outgoing (border) MTA* of *a.com*. Then, the outgoing MTA identifies the incoming MTA of the destination domain, e.g. *b.org*, by an MX DNS query; finally, the email flows inside the destination domain *b.org* till it reaches the MDA.

All of the three standard email protocols we mentioned – POP, IMAP and SMTP – were originally designed for a benign environment, with very limited security mechanisms. In particular, none of them protect the confidentiality of email against an eavesdropper. Complementing standards and protocols, such as S/MIME (Ramsdell, 2004) and OpenPGP (Elkins et al., 2001), use public key encryption and signatures and can provide end-to-end protection of the confidentiality and authenticity of email, even against an MITM attacker. However, these standards are applied only to a small fraction of email messages. Still, it is not very common for attackers to be able to eavesdrop on communication over the Internet; and when this is a concern, the sender and recipient can simply use S/MIME or OpenPGP.

However, email has several additional vulnerabilities, which can be exploited even by attackers who only have capabilities of regular users (often by controlling many *zombies*, i.e. compromised machines of benign users). Table 1 lists some of the most important vulnerabilities, with the resulting threats, the adversary capabilities required to exploit the vulnerability, and the main countermeasures and tools to address the vulnerability. We focus on the two bottom rows of the table.

The designers of SMTP did not include authentication mechanisms intentionally, since this allows maximal connectivity. Namely, SMTP does not require prior arrangement or relationship between sending and receiving domains. In particular, the incoming MTA of *b.org* is not necessarily aware of the IP address of the outgoing MTA of *a.com*. Therefore, unless appropriate sender authentication mechanisms are used, an adversary can send (spoofed) email, pretending it is sent from the outgoing MTA of *a.com*, simply by opening an SMTP connection to *b.org* and sending the domain *a.com* in its HELO (or EHLO) message.

Furthermore, email messages are often associated with several 'sender addresses'. Most obviously, there is an address which the recipient's MUA shows to the user, as the sender (or 'from') address. Specifically, this is usually the address in the FROM email message header, or if such is not available, in the SENDER header (Resnick, 2008). As we discuss later, some MUAs may display two addresses (often as 'from f@foo.org on behalf of accts@VIC-Bank.com'). A possibly different address, is used by mail servers to send delivery status notifications (DSNs), to report of (possible) delivery problems; this address is often referred to as the MAIL FROM address (since it is carried in the MAIL FROM SMTP directive); and several more sender-related addresses may exist.

SMTP allows an outgoing server belonging to one domain, e.g. foo.org, to deliver messages with sender addresses in other domains, e.g. VIC-Bank.com. Therefore, adversaries may be able to send email with spoofed sender addresses, even when they do not have IP spoofing or DNS poisoning capabilities. The ease of email spoofing can help in phishing.

Notice that unlike DNS, the SMTP protocol always runs on top of the reliable connection-oriented TCP protocol. TCP connections begin with a 'three-way handshake' phase, where each party sends a random 32-bit Initial Sequence Number (ISN) to the other party, who should respond with ISN + 1. A spoofing adversary cannot intercept the ISN sent by the other party, therefore it can only guess the ISN + 1 expected in response; therefore it is hard for a spoofing adversary to inject or modify data on SMTP (and other TCP) connections (cf. e.g. to DNS). However, as explained above, SMTP spoofing does not require the adversary to inject or modify data on a (legitimate) connection, or to send packets with spoofed source IP address; an email spoofing adversary can simply open a new connection to incoming MTA of the destination.

There are, however, some possible obstacles to email spoofing. In particular, the incoming MTA of a domain normally waits for incoming SMTP connections only on port 25. Companies and Internet Service Providers (ISPs) often place

Table 1 – Email vulnerabilities, threats and countermeasures.

Vulnerability	Threat	Adversary	Countermeasure	Tools
Sent in clear	Exposure	Eavesdropper, MITM	Encryption	S/MIME (27), OpenPGP (11)
Cheap to send	Spam	Client	Domain reputation	Blacklists, rDNS, captcha
No content validation	Phishing, fraud, hoaxes, malware	Client	Filter, limit, indicate, educate	Content filters, user agents
Sender identity not validated	Phishing, spam, fraud, malware	Client	Authenticate sender and/or domain	SPF (31), DKIM (1), SIDF (24)
Bounce address not validated	'Joe-job': clog bounce recipient	Client	Authenticate MAILFROM	SPF (31), BATV (8)

a packet filtering rule blocking access from their network to TCP port 25 of external hosts, allowing such access only to their outgoing border MTA. This allows to deliver email to incoming MTAs of other domains only from the outgoing border MTA of the domain. This (widely deployed) technique is recommended by Lindberg (1999), and referred to as *port 25 blocking*. Port 25 blocking prevents domain users from acting as outgoing MTAs, and allows the domain administrators to apply restrictions and filtering mechanisms, in particular to deal with spam or phishing email originating from users of the domain. Notice that users may still be able to submit mail to their Mail Submission Agent (MSA) when they reside in a separate network, by submitting to a different port (587 by default, see Lindberg (1999)). Unfortunately, not all companies and ISPs enforce port 25 blocking. Furthermore, even providers that block port 25, and allow only sending email via their outgoing MTA, do not necessarily prevent spoofing of different sender-related addresses in the email, such as the FROM email message header (Resnick, 2008) or the MAIL FROM SMTP directive (Klensin, 2008).

To conclude this brief review of the email system, we note that email is often *mangled* in transit. *Email mangling* refers to 'harmless' modifications of messages, e.g. insertion of 'white space' such as <cr><lf> to break long lines; such modifications do not change textual email. Furthermore, sometimes mail contents are changed beyond mangling, e.g. by adding prefix and/or suffix text ('this message was scanned by…'). Finally, mail agents often add header fields (e.g. RECEIVED:), and may also remove header fields (e.g. for efficiency or to hide sensitive route information), modify or reorder them. In addition to such email mangling, email messages are often slightly modified, esp. to add annotations of services performed on the email, e.g. scanning to detect viruses or other problems; results are usually appended to the original message. All forms of email mangling, including appending of annotations, can cause failure of solutions based on cryptographic authentication of the email, e.g. by digital signatures. In particular, DKIM has some mechanisms to allow authentication to remain valid in spite of certain limited mangling and annotations; see details in Section 5.

3. Sender Policy Framework (SPF)

In this section, we discuss the Sender Policy Framework (SPF) (Wong and Schlitt, 2006). SPF allows the incoming MTA of an organization that receives email from the outgoing MTA of another organization, to confirm that this outgoing MTA is authorized to send mail with the MAIL FROM address specified in the email.[1] SPF relies on the secure distribution of the DNS record with the sending domain SPF policy, which is defined using either the (dedicated) SPF record type, or the (general-purpose) TXT record type.

SPF allows an administrator of a domain, e.g. VIC-Bank.com, to publish the domains' *email authorization policy*, specifying which IP addresses can be used by outgoing border MTA(s) using (MAIL FROM) address in the domain VIC-Bank.com. Namely, SPF allows a domain, e.g. VIC-Bank.com, to specify which IP addresses it intends to use for its outgoing mail servers, i.e. it authorizes (only) these IP addresses to send email with a MAIL FROM address in domain VIC-Bank.com. Suppose that an incoming border MTA, say Bob, has an SMTP connection from some IP address, say aa.bb.cc.dd, and the connection uses a MAIL FROM address in domain VIC-Bank.com. If Bob supports SPF, then it should retrieve VIC-Bank.com's SPF policy record, by an appropriate DNS query (for record of type TXT or SPF, of domain name VIC-Bank.com). This policy record can indicate whether aa.bb.cc.dd is 'allowed' to send email with MAIL FROM address in domain VIC-Bank.com.

The policy record identifies the use of SPF (and version), and a list of terms. Terms are of two types, called *mechanisms* and *modifiers* (see Wong and Schlitt (2006)). Terms are evaluated from left to right, until reaching a term whose predicate holds for the sending MTA's IP address; the predicate of the last term is usually all, which is true for all IP addresses. For example, the record spf1 +IP4:1.2.3.4 −all is an SPF policy record for SPF version 1.0, and has two terms: the term +IP4:1.2.3.4, with the *allow* qualifier denoted +, which approves the sending MTA if its IP address is 1.2.3.4, and the term −all, with the *hard fail* qualifier denoted − which forbids any other IP address. The allow (+) qualifier is the default, so it is often omitted. In addition to allow (+) and hard fail (−) qualifiers, SPF has two more qualifiers: *soft-fail* qualifier denoted ~, indicating that the address is 'to be suspected' – not authorized but not conclusively forbidding it either, and

[1] Delivery status notification (DSN) emails, e.g. 'bounce' email, contain a null MAIL FROM address, to avoid 'loop' of DSN emails to unreachable addresses. When receiving email with null MAIL FROM address, SPF validation is performed using the domain name specified by the sending MTA in the HELO directive, identifying the domain that the sending MTA belongs to.

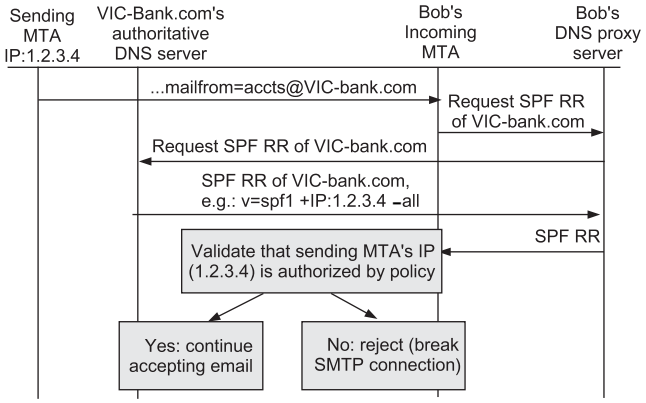

Fig. 1 – Retrieving and using SPF policy. Notice policies are cached by the proxy DNS server (if policy record is already in cache, then the cache does not need to request it from VIC-Bank.com's DNS server).

the *experimental* qualifier denoted ?, indicating that the term (or entire SPF record) is still experimental and should not be relied upon.

The receiving mail agent performing SPF validation, evaluates the terms from left to right, to identify if the SMTP-Sender (e.g. at IP address aa.bb.cc.dd) is allowed to send email with MAIL FROM address in the domain, e.g. VIC-Bank.com. For example, suppose VIC-Bank.com returns record spf1 +IP4:1.2.3.4 −all as a result of a DNS query to records of type TXT or SPF, when Bob's server queries to validate email received from IP address aa.bb.cc.dd. Then, the result is 'allow' if aa.bb.cc.dd = 1.2.3.4, and 'hard fail' otherwise. See example in Fig. 1; for complete details and more examples, see (Wong and Schlitt, 2006).

When the SPF result is not 'allow', then it is possible that the email was not sent by VIC-Bank.com. In this case, if Bob's MTA accepts and transfers the email (e.g. not validating SPF or ignoring the (soft/hard) fail result), then this email may eventually be bounced to VIC-Bank.com, although VIC-Bank.com has never sent this message; such 'fake bounces' are sometimes referred to as 'backscatter', and could be exploited for Denial-of-Service attack on VIC-Bank.com (a Joe-job attack). Avoiding such fake bounces is one of the motivations for using SPF. Hence, publishing an SPF policy record which returns 'hard fail' (except for legitimate IP addresses for an outgoing mail servers of VIC-Bank.com), could allow VIC-Bank.com to avoid receiving fake bounces, if SMTP-Receivers (e.g. Bob) would block email for which the SPF result is 'hard fail'.

However, notice that the mere fact that SPF validation resulted in 'allow' should *not* be taken as an indication that the email is 'good' (not spam or phishing). Nothing prevents spammers from setting SPF records for various domains under their control; indeed it seems that a huge fraction of spam messages pass SPF validation, since they use a spammer-controlled MAIL FROM domain. Email agents should not give any positive credit ('points') for SPF validated emails, except if the domain is considered trustworthy.

On the contrary, when SPF validation results in 'hard fail', it is reasonable to consider the message as 'suspect'. However, unfortunately, in the current deployment of email mechanisms, it is quite possible and even common for a mail server to accept benign (non-phishing/spam) email, from a mail server for which the SPF validation returns 'hard fail'. The risk of rejection of valid email ('false positive'), in this case, is often considered much higher than the potential benefit of blocking email with a fake bounce address, usually spam. Similarly, the fear of such false positives motivates domains to avoid publishing SPF policies with 'hard fail' rules. In an informal study of SPF records by the author and students, we found that the vast majority of deployed SPF records do not contain 'hard fail', and furthermore we found that many mail servers apparently do not block email 'just' since the SPF validation returns 'hard fail'.

Why SPF may fail for benign email ('false positive')? One reason is when the email was sent by a (legitimate) mail forwarding services. There are several proposals to 'fix email forwarding' to avoid this problem, e.g. the Sender-Rewrite Scheme (SRS) (Wong et al., 2004). However, none of these proposals, including SRS, was so far widely adopted. Most of these schemes, including SRS, work by changing the MAIL FROM address to that of the forwarding service, e.g. f@foo.org, and allowing foo.org to forward any DSN to the original MAIL FROM address. This approach would prevent SPF rejection by recipients (since the email from foo.org have MAIL FROM address f@foo.org).

However, this implies that the recipient will consider this email as originating from foo.org. Therefore, recipients relying on SPF as an indication that the domain in the MAIL FROM is 'responsible' for the email, e.g. to skip content-filtering for email from non-spamming domains, may want to ignore the result for SRS-formatted *mail from* addresses. Furthermore, if a recipient applies content-classification to the email and thereby detect that the email appears (by content) to originate from VIC-Bank.com, but has MAIL FROM of foo.org, then the email may be suspected as 'phishing' (false positive).

Merely blocking emails for whom SPF fails may result in 'false negatives', i.e. may pass spam/phishing emails, in one of two cases. First, this will happen if foo.org fails to perform SPF validation. Second, spam/phishing emails may uses a MAIL FROM address not of VIC-Bank.com, and thereby pass SPF validation by foo.org; when foo.org forwards the email, the MAIL FROM is of foo.org, therefore may be considered 'good' by the filtering

mechanisms (and end-users). We conclude that to provide good defense against phishing, SPF must be used together with reputation and classification mechanisms, e.g. as in Herzberg (2009).

This problem can be addressed in the typical case where foo.org is a forwarding service trusted by the recipient. In this (typical) case, foo.org may communicate the result of its (SPF or other) authentication validation, e.g. using an agreed-upon email header such as AUTHENTICATION-RESULTS or SENDER-AUTH-HEADER (Kucherawy, 2009). Since the MAIL FROM address is now of foo.org SPF validation will pass, avoiding 'false positive' (incorrectly identifying valid mail as spam/phishing); and since foo.org has indicated the results of its own SPF validation in the agreed-upon header, e.g. AUTHENTICATION-RESULTS, the receiver can still decide if to block due to SPF validation failures (detected by foo.org).

A positive ('allow') SPF indication can help to establish the reputation of an (new) outgoing MTA, on the reputation or identity of the domain (e.g. VIC-Bank.com). Namely, if VIC-Bank.com suddenly changes to a new outgoing MTA (using a new IP address), then an SPF record can show that this new IP is valid for VIC-Bank.com. Similarly, if a domain e.g. foo.org has good reputation (as not sending spam or other phishing email), and its SPF record indicates that the Sender-SMTP (at IP aa.bb.cc.dd) is allowed to send email with MAIL FROM address in domain foo.org, then Bob may receive the email, e.g. without subjecting it to content-classification. Notice that similar reputation mechanism can also work based only on the IP address of the outgoing MTA, however, using SPF (and hence basing the reputation on the domain name) allows using the same reputation record for multiple IP addresses, including extending the reputation for a new IP address.

Such positive SPF indication may be useful even to recipients that do not rely on SPF's negative indications (to avoid false positives e.g. due to forwarding, as discussed above). These recipients may only check for SPF records randomly (or for updates to known SPF records); there may be value in a convention allowing senders to specify existence and location of the SPF record, for the benefit of such recipients (interested only in 'positive' SPF indicators)[2]; similar indicators exist in Sender-ID and DKIM (for DKIM, this is the signature message header, and in Sender-ID, this is the SUBMITTER keyword of the MAIL FROM SMTP directive). We note, however, that this (simple) proposal has not been proposed before, and in particular not included in Wong and Schlitt (2006).

SPF uses the Domain Name System (DNS) to distribute the domain policy record, since DNS is controlled by the owner of the domain, and is a convenient and efficient distribution mechanism. Notice that DNS poisoning adversaries (and Man In The Middle (MITM) adversaries) can cause false positive and false negative failures for policy-based authorization mechanisms, and in particular SPF, which are based on retrieving the policy record from DNS. Policy records may be signed, e.g. by using secure DNS (Arends et al., 2005), to prevent this threat; in reality, this is usually considered an overkill for the limited goals of policy-based authorization mechanisms.

Furthermore, SPF may help an attacker facilitate DNS poisoning and Denial of Service attacks. Specifically, an attacker can construct SPF policy which causes multiple DNS lookups in arbitrary domains, e.g. of a victim DNS server. This threat was raised in Otis (2006), and addressed in the SPF specifications (Wong and Schlitt, 2006), by setting a limit of ten mechanisms causing DNS lookups during validation of a single SPF policy, and a limit of ten DNS records lookups per mechanism. Otis (2006) shows that the combination of these two restrictions allows a single SPF validation to result in up to 100 DNS resolutions, allowing a relatively weak attacker to cause significant overhead. This technique allows an attacker to cause the receiving MTA to generate a series of a large number of DNS queries controlled by the attacker. Attackers can use this to 'clog' the recipient (with a large number of DNS responses) or to cause high load on some victim DNS server(s).

Furthermore, this can help an attacker to poison the DNS proxy used by the receiver MTA, extending on the attack of Kaminsky (2008). First, the attacker can cause many requests and thereby slow down responses from authoritative DNS servers. Second, the attacker can spoof the responses to the SPF requests (controlling the contents and timing of the requests is very helpful). Finally, the attacker can generate a query to a machine it controls, allowing it to synchronize with the timing of the queries (to increase its chances of spoofing the response before the real response is received). Note that these attacks can be more severe if multiple agents perform SPF validation, and especially if the recipient's Mail User Agent (mail client) performs SPF validation (which is a non-standard use of SPF validation, but deployed by some mail clients).

Due to these potential abuses of SPF policies, we recommend that SPF validation would be performed only by receiving MTA (and not by MUA), and only after blocking blacklisted senders. The receiving MTA performing SPF validation should use a dedicated DNS proxy, so that even if it is poisoned, the impact is limited (to DNS lookups by the receiving MTA) and cannot help in attacks against other computers and services. Furthermore, recipients may consider limiting the number of DNS queries sent when validating a single SPF policy (even more stringent than the limits defined by Wong and Schlitt (2006)). Such limits can be imposed in the SPF validating process, and/or in the dedicated DNS proxy (as recommended above). Finally, these SPF threats to DNS poisoning, provide yet additional incentive for protecting DNS against poisoning, by performing DNS queries over TCP or SCTP, and by using DNS security (Arends et al., 2005).

To summarize, SPF allows an incoming MTA to validate that the outgoing MTA sending the email, is authorized to send using the given MAIL FROM address; this validation is secure against email spoofing adversaries and IP spoofing adversaries, but not against MITM adversaries or DNS poisoning

[2] SPF requires placing its policy record as a TXT or SPF resource record of the domain name, in the authoritative name server of the domain. This ensures that recipients can easily locate such record, without relying on any 'hint' in the email. However, this may cause overhead in two ways: first, the domain name may have other TXT records already; second, when there is no such SPF record, the receivers still have to make two DNS queries – for both the TXT and SPF records. When the receiver only needs the policy record for 'positive' indication, i.e. to avoid false positives or skip unnecessary validation work, as suggested above, we can place the policy record in an arbitrary DNS record controlled by the sender, and indicate this record to the recipient e.g. in an email message header.

adversaries. SPF validation may fail if (legitimate) email was forwarded, unless the forwarding service is trusted and signals the result of SPF validation, as discussed above.

4. Sender-ID Framework (SIDF)

SPF validates only that the sending mail server is authorized by the owner of the domain of the MAIL FROM address (or if absent, the HELO address). However, these addresses are not necessarily related to the sender email address presented by the mail reader to the user, which is usually the FROM or SENDER address of the original sender as specified in the corresponding email message headers, or sometimes the address of a mailing list or similar service used to forward a message from one sender to a list of recipients. Therefore, SPF validation does not provide direct defense against spoofed (phishing) emails.

The Sender-ID Framework (SIDF) (Lyon and Wong, 2006) is an extension of SPF, that attempts to provide (some) defense against phishing, specifically against unauthorized use of user-visible sender addresses.

This goal is challenging. Email messages may contain multiple headers identifying entities involved in sending the message, including the FROM header (for 'entity responsible for the message'), the SENDER header (for 'entity who entered the message into the mail system'), the RESENT FROM, RESENT SENDER headers (used by some mailing lists and other forwarding services), and more. Furthermore, different email readers may present different sender addresses to the user.

The SenderID Framework addresses this challenge by defining a new identifier, the *Purported Responsible Address* (PRA) (Lyon, 2006). The PRA is not an SMTP directive (like MAIL FROM) or an email header field (like the FROM header). Instead, the PRA is a result of a function, specified by Lyon (2006), applied to existing email message header fields (Resnick, 2008). As the name implies, the Purported Responsible Address (PRA) is supposed to be the identification of the party which is 'responsible' for the email, i.e. whose policy was used to authenticate the mail server sending the email and whose reputation should be used to filter the email; this is also the party who should be 'blamed' if the email is found to be spam or phishing. Namely, the PRA should identify the last mailing list or forwarding service which forwarded the message; for this purpose, the PRA is defined as the most recently added resent-from or resent-sender email message header. This assumes that all mail forwarding agents add such resent fields; notice that this assumption is *not* always valid; in fact, the use of these fields by SIDF conflicts with their definitions by Resnick, 2008. For email that was never forwarded, the PRA should simply identify the email sender, usually identified by the FROM or SENDER email message headers.

Preferably, to provide defense against phishing, the mail readers should present the PRA to the user, hopefully allowing the user to identify phishing email by its provision via untrusted server. However, when email was forwarded (e.g. by forwarding service of mailing list), users are usually most interested in the identity of the original sender. When the PRA differs from the email sender as identified in the email headers, a Sender-ID compatible mail user agent (MUA) may display both identities, typically by text such as FROM: f@foo.org ON BEHALF OF: accts@VIC-Bank.com. This could help users using MUAs supporting SIDF to identify phishing emails, if users would securely validate both sender and PRA identifiers (and never trust email sent from a domain other than VIC-Bank.com, or forwarded by an untrusted agent), and if the PRA (e.g. f@foo.org) were indeed responsible to only forward properly authenticated email with FROM or SENDER from VIC-Bank.com. Unfortunately, neither assumptions seem likely to hold:

- Based on experimental works on phishing (Dhamija et al., 2006; Herzberg and Jbara, 2008), detection rates of spoofed emails may be low, even when the email is sent with false sender identification. In fact, we expect even lower detection rates when presenting both the PRA and the claimed FROM address, as above, due to confusion and misunderstanding of the 'from… on behalf of…' issue. Furthermore, email addresses can contain arbitrary 'comments' in addition to the actual address, which can be abused to create highly misleading addresses. Further experimental work should confirm (or refute) these expectations.
- Existing mail forwarding services do not perform the sender validation as required by SIDF. Furthermore, even if a forwarding service performs such sender validation, it cannot truly validate the sender address, in case it receives email forwarded by some *other* forwarding service. Therefore, even when Bob sees 'FROM: f@foo.org ON BEHALF OF: accts@VIC-Bank.com', and even if Bob completely trusts f@foo.org, and f@foo.org indeed performs SIDF validation, this may still be a phishing email (i.e. not from accts@VIC-Bank.com). This concern may be addressed when the forwarding service, e.g. foo.org, is trusted by the recipient, and it authenticates the sender and signals the results of this authentication, e.g. via the SENDER-AUTH-HEADER email header (Kucherawy, 2009), exactly as discussed above for SPF. However, here there is no real advantage of SIDF compared to SPF.

SIDF reuses most of the policy mechanisms of SPF. In particular, entities define their SIDF policies as DNS records, usually of type TXT, and even beginning with the same identifying string: V=SPF. The rest of the policy also uses exactly the same syntax and semantics as in SPF, except for the characters immediately following the V=SPF prefix. In SPF, this string is followed by the version number, which is currently only defined as 1; i.e. SPF records always begin with the six letters V=SPF1. SIDF records are identified by as if they are version 2.0 of SPF, i.e. by a record beginning with V=SPF2.0. This (controversial) use of the prefix V=SPF2.0 to identify SIDF policy records, is a remnant to the failed effort by the MARID IETF working group to merge the SPF and SenderID proposals (Contributors, 2008).

The difference between SIDF and SPF is in the identities to which the policy is applied. SPF policies, as explained above, refer to the MAIL FROM address (or if absent from the email, to the HELO address). In contrast, SIDF policies can refer to the MAIL FROM address, the PRA address, or to both. SIDF records specify the relevant identifier (or identifiers) by the keywords MFROM and/or PRA, immediately following the V=SPF2.0 record prefix; for example, when including both keywords, as in V=SPF2.0/MFROM,PRA, this specified that the policy applies to both

the MAIL FROM address and to the PRA address. A controversial recommendation in the SIDF specification, instructs recipients to interpret 'classical SPF' records, i.e. records beginning with V=SPF1, as equivalent to V=SPF2.0/MFROM,PRA; arguably, this may cause recipients to reject legitimate mail, e.g. if an organization uses one mail server for sending its own emails (specified in the SPF record) and a different mail server when forwarding email (or running a list-server), where the address of this second server is not included in the SPF record.

To conclude, we do not see how the PRA authentication facility of SIDF, offers significant additional defense against phishing, compared with the use of the MAIL FROM authentication (provided also by SPF).

5. Cryptographic sender authentication and DKIM

There are several proposals and mechanisms for cryptographic authentication of email, mostly using digital signatures. In all proposals, the authentication tag is appended to the email and sent with it, and validated by the recipient (and possibly intermediaries). Some of these email authentication techniques are established standards, most notably OpenPGP (Callas et al., 2007; Zimmerman, 1995) and S/MIME (Ramsdell, 2004); both provide general-purpose security for email, with support for confidentiality as well as authentication. However, both assume that the MUA of the email recipient, will support the corresponding standard, and know how to handle and display signed messages; this causes a usability problem for users of non-conforming mail readers. In addition, both proposals required complex key management, and do not support signing by a domain on outgoing messages of its customers. Finally, these standards do not define how a sender can publish that all of its email is signed; this information is critical to allow recipients to suspect unsigned email claiming to come from that sender.

We focus on DomainKeys Identified Mail (DKIM) (Allman et al., 2007), a newer proposal for digital-signature-based email sender authentication, which is focused on prevention of phishing and spam. DKIM allows an incoming MTA, or any other mail agent (including the recipient's MUA or MDA), to authenticate email by using cryptographic sender signature, validated using the sender's public key.

DKIM avoids message modification or new MIME types, by placing the signature in email header fields, which are typically not shown to end-users by MUAs. Specifically, DKIM adds a new, special header field, DKIM-SIGNATURE; since unknown email headers are silently ignored, this does not cause a usability problem, even for users of non-conforming mail readers. DKIM also has the following additional features: (a) simple default key management based on DNS, with ability to specify other key management mechanisms, (b) ability to sign emails of an entire organization or domain, (c) limited robustness to email mangling, and (d) support for signing a signer-chosen selection of email headers.

In particular, DKIM allows signing of email headers which the Mail User Agent presents to the user to identify the sender. Furthermore, DKIM specifies that the FROM header field *must* be signed; other significant headers, that may mislead the user, should also be signed, e.g. SUBJECT, REPLY-TO and DATE, but headers that may change in transit should not be signed (since this may invalidate the signature).

Signing user-visible headers can protect from misleading users in different ways, e.g. from responding to the attacker (with forged REPLY-TO header, causing response to be sent to the attacker). However, this does not prevent an attacker from sending email with misleading identifiers, either unsigned or signed by the attacker.

As (partial) defense against this threat, DKIM also defines an *author domain signing practices (ADSP)* policy (Allman et al., 2009), which is kept as a special DNS record in the sender domain. The ADSP is an efficient, standard mechanism for senders to signal whether they sign *all* of their outgoing emails, and furthermore whether they recommend to discard any unsigned mail which indicates a user in this domain in the FROM or SENDER headers; essentially, this is a (much simpler) variant of the sender's policy in SPF and Sender-ID. The DKIM specification (Allman et al., 2007) recommends that an incoming border MTA will request the ADSP from the domain of the address in the FROM email header, which is usually also the 'from/sender' address displayed to the user.

However, as indicated above, ADSP offers only limited protection against phishing, for three reasons. First, ADSP security relies on the security of the Domain Name System, rather than relying on the security of digital signatures; this can be fixed by using secure DNS (Arends et al., 2005) to sign the ADSP record (or to authenticate the fact there is no ADSP record). Second, ADSP relies on recipients retrieving and validating ADSP records for all incoming emails; due to its limited value and deployment, many recipients may decide not to do this. Finally, experiments on user ability to detect phishing attacks (Dhamija et al., 2006; Herzberg and Jbara, 2008), seem to indicate that users have very limited ability to detect phishing emails, based on incorrect sender address; we believe a much better way to use DKIM to detect phishing emails would be by combining it with classification mechanisms, as in Herzberg (2009).

DNS-based authentication of SPF and SIDF could also be used in combination with content-based classification to detect phishing. The differences are in *method*, *efficiency* and overhead, *security* against attackers with different capabilities, and *robustness* to handling by email agents. Details follow.

Method and efficiency. DKIM authenticates the original sender of the email, or of some other mail agent who signed the message while forwarding it, by validating its digital signature over the message; unlike the IP address of the sending MTA, used by SPF and Sender-ID, the signature should not change even when the email is forwarded. Hence, DKIM is robust to email forwarding, in contrast to SPF and Sender-ID, which often 'break' on forwarding, and at best require cooperation and trust of the forwarding agents.

Security. DKIM sender authentication can be secure against MITM and DNS-poisoning adversaries; this improves upon SPF and Sender-ID, which are both vulnerable to DNS-poisoning and MITM adversaries. However, this requires deviation from the default deployment as specified in (Allman et al., 2007), or usage of secure DNS (Arends et al., 2005). The default DKIM deployment uses and depends on DNS for distribution of both the sender's public validation key and of

the sender's author domain signing practices (ADSP) policy (Allman et al., 2009). The motivation for this is simplicity – the DNS system already provides some level of control for each domain over its records – and efficiency of DNS query mechanisms. However, this implies vulnerability of default DKIM, even to the (relatively weak) DNS-poisoning adversary. This weakness can be avoided, by using secure public key distribution (e.g. using certificates, like in SSL/TLS (Dierks and Rescorla, 2006)), or using secure DNS (Arends et al., 2005). If using ADSP, it should also be signed.

Robustness to handling by email agents. DKIM is robust to forwarding of the message, in contrast to SPF and SIDF. However, DKIM's use of cryptographic signatures implies that its authentication can break even for 'minor' email mangling, as well as addition of trailing text as done e.g. by some email systems to signal scanning against viruses or other services. DKIM deals with email mangling and addition of (trailing) text, with two mechanisms: *canonicalization* and *limited scope*. By limiting the scope of the signed message parts, we avoid parts which we consider non-essential and modifiable. Namely, many email signing standards apply the authentication only to specific parts of the message, e.g. defined as *scope*(*m*). DKIM allows the signer to extend the scope to include arbitrary header fields, and to specify the number of bytes in the body of the email that are included in the computation of the tag, using the l attribute. When the l = *l* is included, only the first *l* bytes of the email are authenticated, allowing intermediate agents to append trailers such as added by some mailing lists and anti-viruses.

Canonicalization deals with mangling of the signed message parts by different mail agents along the path, potentially invalidating the signature. For most messages and signers, some common modifications such as white-space replacement and wrapping of long header lines do not materially change the message, and can be permitted. Such senders may use the RELAXED canonicalization function (Allman et al., 2007). In other cases, any modification may not be acceptable, in which case the default, SIMPLE canonicalization algorithm is appropriate. Other canonicalization functions can be defined.

Summary. DKIM provides an effective mechanism for authenticating emails sent by a domain, essentially without risk of 'false negatives' – there does not seem a way that an email can pass DKIM validation, if it was not sent via an authorized sending server. DKIM use of digital signatures is very different from SPF's reliance on security of DNS and of routing; however, the two systems have much in common. First, the fact that an email passes DKIM (and SPF) validation does not indicate that the email is 'good'; spammers can easily generate valid messages using their own domains. This validation is useful *only* when there are reasons to trust emails from that domain. Second, both DKIM and SPF validation can fail – even for legitimate mail; in DKIM case, this is mainly due to 'email mangling'. We conclude that mail agents should not conclude that a message is spam merely based on DKIM (or SPF) failure; this can be only a very limited indicator. Another conclusion is that mail senders are advised to publish *both* SPF and DKIM policies (and apply DKIM to their outgoing mail), to increase the likelihood that at least one of the two mechanisms will remain intact when the message is validated.

Acknowledgments

Many thanks to Nathaniel (Nathan) Borenstein, Dave Crocker, Jim Fenton, John Leslie, John Levine, Chris Lewis, Amit Klein, der Mouse, Douglas Otis, Haya Shulman, Alessandro Vesely and the anonymous referees, for their helpful and constructive comments. This work was supported by Israeli Science Foundation grant ISF 1014/07.

REFERENCES

Allman E, Callas J, Delany M, Libbey M, Fenton J, Thomas M. DomainKeys Identified Mail (DKIM) Signatures. RFC 4871 (Proposed Standard), http://www.ietf.org/rfc/rfc4871.txt; 2007.

Allman E, Fenton J, Delany M, Levine J. DKIM Author Domain Signing Practices (ADSP). Internet draft draft-ietf-dkim-ssp-10; 2009.

Arends R, Austein R, Larson M, Massey D, Rose S. DNS security introduction and requirements. RFC 4033 (Proposed Standard), http://www.ietf.org/rfc/rfc4033.txt; 2005.

Bellovin SM. Security problems in the TCP/IP protocol suite. Computer Communication Review 1989;19(2):32–48.

Callas J, Donnerhacke L, Finney H, Shaw D, Thayer R. OpenPGP message format. RFC 4880 (Proposed Standard), http://www.ietf.org/rfc/rfc4880.txt; 2007. Updated by RFC 5581.

Contributors: MARID. Wikipedia, The Free Encyclopedia; 2008.

Crispin M. Internet message access protocol – VERSION 4rev1. RFC 3501 (Proposed Standard), http://www.ietf.org/rfc/rfc3501.txt; 2003. Updated by RFCs 4466, 4469, 4551, 5032, 5182.

Crocker D. Challenges in anti-spam efforts. The Internet Protocol Journal 2005;8(4):2–14.

Dhamija R, Tygar D, Hearst M. Why phishing works. Montreal, Quebec, Canada. In: Proceedings of the conference on human factors in computing systems (CHI2006); 2006. p. 581–90.

Dierks T, Rescorla E. The Transport Layer Security (TLS) Protocol Version 1.1. RFC 4346 (Proposed Standard), http://www.ietf.org/rfc/rfc4346.txt; 2006. Obsoleted by RFC 5246, updated by RFCs 4366, 4680, 4681.

Elkins M, Torto DD, Levien R, Roessler T. MIME security with OpenPGP. RFC 3156 (Proposed Standard), http://www.ietf.org/rfc/rfc3156.txt; 2001.

Ferguson P, Senie D. Network ingress filtering: defeating denial of service attacks which employ IP source address spoofing. RFC 2827 (Best Current Practice), http://www.ietf.org/rfc/rfc2827.txt; 2000. Updated by RFC 3704.

Gilad Y, Herzberg A. Lightweight Opportunistic Tunneling (LOT). In: 14th European symposium on research in computer security. ESORICS; 2009.

Harris E. The next step in the spam control war: greylisting, http://projects.puremagic.com/greylisting/whitepaper.html; 2003.

Herzberg A. Combining authentication, reputation and classification to make phishing unprofitable. In: Proceedings of the IFIP 24th international information security conference, IFIP SEC 2009; 2009.

Herzberg A, Jbara A. Security and identification indicators for browsers against spoofing and phishing attacks. ACM Transactions on Internet Technology 2008;8(4).

Kaminsky D. It's the end of the cache as we know it. Online at:. In: Black Hat conference http://www.doxpara.com/DMK_BO2K8.ppt; 2008.

Klensin J. Simple mail transfer protocol. RFC 5321 (Draft Standard), http://www.ietf.org/rfc/rfc5321.txt; 2008.

Kucherawy M. Message header field for indicating message authentication status. RFC 5451 (Proposed Standard), http://www.ietf.org/rfc/rfc5451.txt; 2009.

Leiba, B., Borenstein, N.S. A multifaceted approach to spam reduction. In: CEAS 2004-First Conference on Email and Anti-Spam; 2004.
Lieba B, Fenton J. DomainKeys identified mail (DKIM): using digital signatures for domain verification. In: CEAS 2007: the third conference on email and anti-spam; 2007.
Lindberg G. Anti-spam recommendations for SMTP MTAs. RFC 2505 (Best Current Practice), http://www.ietf.org/rfc/rfc2505.txt; 1999.
Lyon J. Purported responsible address in e-mail messages. RFC 4407 (Experimental), http://www.ietf.org/rfc/rfc4407.txt; 2006.
Lyon J, Wong M. Sender ID: authenticating e-mail. RFC 4406 (Experimental), http://www.ietf.org/rfc/rfc4406.txt; 2006.
Myers J, Rose M. Post office protocol – version 3. RFC 1939 (Standard), http://www.ietf.org/rfc/rfc1939.txt; 1996. Updated by RFCs 1957, 2449.
Otis D. SPF DoS Exploitation. Internet draft draft-otis-spf-dos-exploit-01; 2006.
Ramsdell B. Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.1 Message Specification. RFC 3851 (Proposed Standard), http://www.ietf.org/rfc/rfc3851.txt; 2004.
Resnick P. Internet message format. RFC 5322 (Draft Standard), http://www.ietf.org/rfc/rfc5322.txt; 2008.
Siemborski R, Melnikov A. SMTP service extension for authentication. RFC 4954 (Proposed Standard), http://www.ietf.org/rfc/rfc4954.txt; 2007. Updated by RFC 5248.
Thomas R. On the problem of signature authentication for network mail. RFC 644, http://www.ietf.org/rfc/rfc644.txt; 1974.
Wong M, Schlitt W. Sender policy framework (SPF) for authorizing use of domains in e-mail, version 1. RFC 4408 (Experimental), www.ietf.org/rfc/rfc4408.txt; 2006.
Wong MW. SPF, MTAs and SRS. Linux Journal 2004;2004(121).
Zimmerman PR. The Official PGP User's Guide. MIT Press; 1995.

Prof. Amir Herzberg has over 25 years of experience in computer programming and engineering, focused on security and networking. He received B.Sc. (Computer Engineering), M.Sc. (Electrical Engineering) and D.Sc. (Computer Science), from the Technion, Israel, at 1982, 1987 and 1991, respectively. Since 2002, he is with the Computer Science department of Bar Ilan University. During 1991–2000, Prof. Herzberg filled research and management positions in IBM Research (New York and Israel), and previously he filled R&D and management positions in the Israeli Defense Forces and companies. His current research interests include secure computing and communication, including applied cryptography, secure usability, and efficiency and reliability of communication, esp. in emerging areas such as vehicular networking.