Uncovering identities: A study into VPN tunnel fingerprinting

Vafa D. Izadinia, D.G. Kourie and J.H.P. Eloff

Abstract

Operating System fingerprinting is a reconnaissance method which can be used by attackers or forensic investigators. It identifies a system's identity by observing its responses to targeted probes, or by listening on a network and passively observing its network ‘etiquette’. The increased deployment of encrypted tunnels and Virtual Private Networks (VPNs) calls for the formulation of new fingerprinting techniques, and poses the question: “How much information can be gleaned from encrypted tunnels?” This paper investigates IPSec VPN tunnel-establishment and tear-down on three IPSec implementations: Microsoft Windows 2003, Sun Solaris 9 x86, and *racoon* on Linux 2.6 kernel. By analysing each platform's Internet Key Exchange (IKE) messages, which negotiate the IPSec tunnel, we identify a number of discriminants, and show that each of these platforms can be uniquely identified by them. We also show that the nature of some encrypted traffic can be determined, thus giving the observer an idea of the type of communication that is taking place between the IPSec endpoints.

Article Outline

1. Introduction

The aim in developing protocols and cryptographic algorithms, is to increase the overall security of a system. More often than not, these protocols and algorithms are technically sound and astonishingly elegant in their simplicity (Schneier and Ferguson, 2003, Schneier, 1996 and Koblitz, 1994); however, they are implemented in ways they were not intended to be, or with minor 'enhancements' that digress from the original design, often resulting in unanticipated weaknesses in the system. These two: the implementation, and the relaxed (or *uncommitted*) adherence to the original standards, account for weaknesses in vendors' products. It comes as no surprise then, when products boasting strong encryption, are bypassed, or broken-into. Whether it be intruders attempting to break a system, designers trying to devise a way to better protect the information, or investigators putting together clues in the aftermath of a break-in; the more is known of the system as a whole, the more detailed the image that can be constructed.

One method for acquiring this information is termed as Operating System (OS) fingerprinting (Yarochkin, 1999, Nazario, 2000, Veysset et al., 2002, Beverly, 2004 and Lee et al., 2002). One form of OS fingerprinting, probes the target system by sending well-known, specially crafted, or otherwise unexpected information to the target, in an attempt to elicit a response from it. This response, in turn, differs from operating system to operating system (and even between different versions of the same OS); thus enabling the interested party to determine what OS the target system is running. Fingerprinting can be used by intruders to determine what avenue of attack to employ when breaking into a system; it can be used by investigators to piece together the network path along which an attack took place; and it can also be used by system designers and developers in an attempt to better disguise the identity of a system.

OS fingerprinting can be used by legitimate network administrators, as well as by individuals of ill-intent, wishing to obtain information otherwise hidden behind a firewall. This is known as network mapping (Broido and Claffy, 2001 and Lowekamp, 2003) or topology discovery (Huffaker et al., 2001). Once the fingerprinting process has produced a list of machines with matching OS details, an attacker may determine which machines are most vulnerable, and prepare to launch an attack, exploiting the machines' weaknesses.

The work presented in this paper, is intended to highlight a new method of OS fingerprinting, which can be used in a complementary fashion, to those already in existence.

1.1. Motivation

As Virtual Private Network (VPN) technology is becoming ubiquitous, it follows that new avenues of attack are being devised, and with them, new methods of traffic analysis.

When deploying VPN tunnels, the endpoints are often placed behind firewalls, effectively bypassing them. Being able to glean information about the devices between

which the tunnel is being established (i.e., the tunnel endpoints) may enable an attacker to bypass restrictive firewall Access Control List (ACL) clauses. Fig. 1 shows a typical deployment of VPN gateways (terminators/concentrators) placed behind firewalls in two communicating corporate networks, resulting in the VPN tunnel being established through existing firewall policies. The encapsulated IPSec traffic appears as a payload to regular IP traffic.[1]

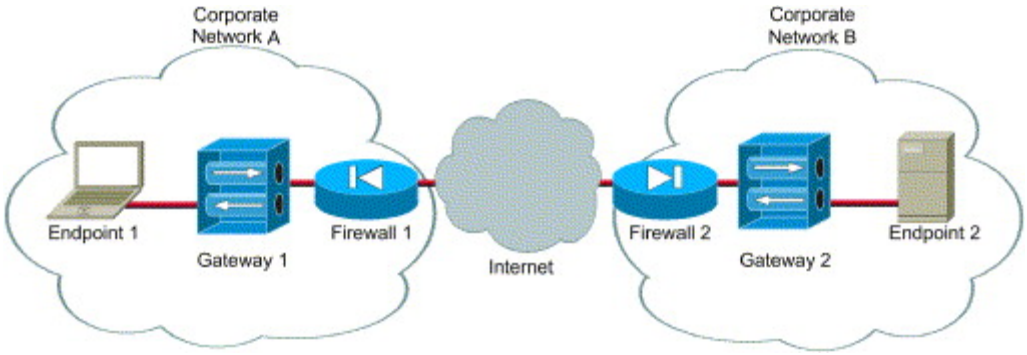

Fig. 1. Depiction of a VPN tunnel established *through* firewalls, initiating and terminating on gateways behind the firewall in each corporate network.

This paper presents the findings of a study involving three IPSec VPN implementations, available on widely used workstation and server OSs. These are Windows 2003, Linux (2.6 kernel), and Solaris 9 x86. It explores the question of whether Internetworking protocols and cryptographic algorithms that are relied on for end-to-end security are weakened by variations in their commercial implementation. The information presented here is derived from work reported in Izadinia (2004).

1.2. Related work

Up until now, fingerprinting has been done by one of the following methods:

• Observing the behaviour of the TCP/IP stack in the target host (Yarochkin, 1999); that is, observing how the stack reacts to malformed IP datagrams, to IP datagrams with invalid lengths, to TCP packets with an incorrect combination of flags set, as well as other typical TCP/IP behaviour such as default TCP windows sizes, and initial sequence number (ISN) analysis.

• Banner-grabbing (f0bic, 2001); that is, making FTP, telnet, and HTTP connections to the target machine, and recording (or ‘grabbing’) the default banner displayed (such as *Welcome to Machine-Name, running RedHat Linux 8, Kernel 2.4-19*). This is one of the earliest types of OS fingerprinting. It is included here for historical completeness.

• Gleaning information by attacking specific services on the target machine, such as the SMTP service (Bordet, 2002), and observing its response in the face of errors; such as

sending a MAIL FROM without a HELO, using MAIL FROM <> with an empty address, or using invalid source addresses, to name but a few.

• Observing the target machine's Address Resolution Protocol (ARP) etiquette (<concept@ihug.com.au>, 2000); that is, noting the target machine's Layer 2 reaction to spoofed frames (frames with incorrect source addresses), and the period and frequency of the subsequent ARP requests.

• By observing Internet Control Message Protocol (ICMP) responses from the target hosts (Arkin, 2001 and Arkin and Yarochkin, 2001) which are, more often than not, a result of vendors' non-adherence to RFCs.

• And lastly, by initiating a connection to an IKE daemon on the targeted machine, not completing the Phase 1 negotiation, and observing the manner in which the server tries to recover from this error condition, until it eventually times out (Hills, 2003).

The work presented in this paper approaches the problem in a novel way, in that it performs identification without targeted probes, but rather by observing the Layer 4 exchange between 2 devices wishing to establish an IPSec tunnel.

1.3. Layout

Section 2 begins with a background to IPSec and its key-exchange protocols. Section 3 presents the laboratory layout used to conduct the fingerprinting tests and explains the test methodology developed and employed in this project. Section 4 presents the test criteria, and Section 5 discusses results and research findings. Finally Section 6 wraps things up with a summary and a glimpse into future work.

2. Background

Tunneling can be viewed as a process whereby the entire encapsulat*ed* datagram becomes the payload of the encapsulat*ing* datagram, and is transported through the network using the encapsulating datagram's header information. Adapting slightly from the notation used by Aqun et al. (2000), this can be described as: [Y[tunnel header[X]]], where Y is the encapsulat*ing* protocol, and X is the encapsulat*ed* protocol.

2.1. IKE

The history of IKE's development is peppered with political squabbling, and even after its release as an RFC, it has been the subject of harsh criticism. It derives from several other *key management* protocols: the Internet Security Association and Key Management Protocol (ISAKMP), Oakley, Photuris, and SKEME. Often considered unsuitable to fulfill the task for which it was designed, the IETF convened a working group to ‘fix’ the shortcomings present in IKE; this was named the Son of IKE (SOI), and later IKEv2. The present overview of IKE only discusses IKEv1, since at the time of writing, IKEv2 has

not been ratified as a standard as of yet, and thus should not have found its way into any consumer products.

IKE consists of two distinct phases of operation, named *IKE Phase 1* and *IKE Phase 2*. Phase 1 establishes an ISAKMP Security Association (SA). Phase 2 uses this SA to derive keying material, and set up IPSec SAs. An SA is "the method by which traffic traveling between two endpoints will be protected" (Dunbar, 2001). There are two kinds of SAs in IPSec talk: ISAKMP SAs (sometimes referred to as IKE SAs), and IPSec SAs. IPSec SAs are simplex, or unidirectional (that is, two of them have to be set up for any duplex connection), whereas ISAKMP SAs are bidirectional. In short, an IKE Phase 1 exchange establishes an encrypted (*secure*) channel which is then used in the negotiation of the IPSec SAs (in Phase 2).

Phase 1 has two "modes" (Harkins and Carrel, 1998): *aggressive* mode and *main* mode. Main mode consists of 6 messages: messages 1 and 2 negotiate the cryptographic protocols and parameters, messages 3 and 4 conduct the Diffie-Hellman (DH) exchange, and messages 5 and 6 provide Perfect Forward Secrecy (PFS). Aggressive mode consists of three messages, since it does not offer PFS. In aggressive mode, the Diffie-Hellman exchange takes place in the first two messages, and the second and third messages serve for each side to prove that they know the DH value and their respective secret (Perlman and Kaufman, 2000) (that is, the second message carries more information in aggressive mode than it does in main mode). Phase 2 is referred to as Quick Mode, and is an exchange consisting of only three messages, all of which are encrypted.

3. Laboratory layout

The aim of laboratory design was to represent the minimum topology that would approximate a typical branch office VPN scenario, where two disparate networks are connected by setting up a tunnel through the Internet (Fig. 2).

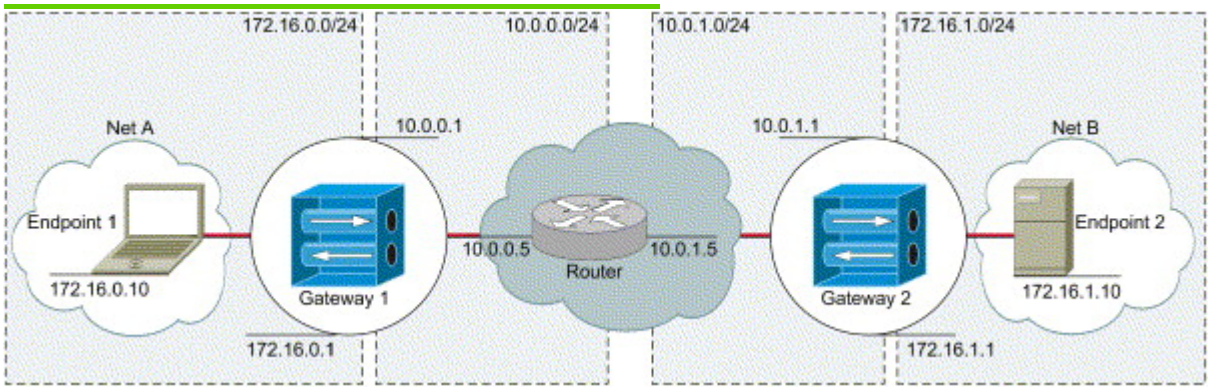

Fig. 2. Laboratory layout.

In order to configure the IPSec implementations on each gateway machine, official documentation on the topic was obtained from the corresponding vendors (Microsoft, Sun Microsystems), and from the official developers' sites (*isakmpd* and *racoon*), and followed to the letter. The ICSA Labs recommendations delineated in ICSA-Labs (2003) were followed as far as cryptographic options to Phase 1 and Phase 2 were concerned.

Two tests were conducted. The first consisted in sending an ICMP ECHO REQUEST (*ping*) to endpoint 2 from endpoint 1, causing gateway 1 to establish an encrypted tunnel through the router to gateway 2, and allow the ECHO REQUESTS to traverse the tunnel to endpoint 2, and return the ECHO REPLYs from endpoint 2. As soon as 5 echo replies had been received, the *ping* would be stopped, and the tunnel torn down manually to mark the end of the test.[2] The second test consisted in using the telnet application to connect from endpoint 1, to the day-time port (TCP port 13) on endpoint 2. This would cause the gateways to set up the tunnel, with endpoint 2 returning the current date and time, and closing down the telnet connection. The tunnel would then be torn down on each gateway.

Searching for fingerprints was broken down into three steps:

• Analysis of the connection summary: this provided an overview of the packet exchange between the two gateway machines (source and destination addresses, protocol type and protocol information).

• Analysis of the IKE exchange: this consisted in scrutinizing the ISAKMP payload of each IP datagram involved in the IKE Phase 1 and Phase 2 exchange, and looking for anomalies, differences between the various implementations, and contrasting this information to the IPSec Domain of Interpretation (DOI), ISAKMP, and IKE RFCs (2407, 2408, and 2409, respectively).

• Analysis of the ESP-protected IP datagrams: the last step examined the encrypted IP datagrams which contained the ICMP ECHO REQUEST and REPLY messages, and the TCP handshake and disconnect messages, in order to determine what could be inferred from their encrypted payload.

4. Fingerprinting test results

The process of fingerprinting an encrypted tunnel endpoint is a tricky one. Following the initial messages of the Key Exchange, all the communication are encrypted. It is for this reason that the process could be better defined as fingerprinting the Key Exchange process that takes place between the two endpoints. This Key Exchange, ISAKMP/IKE in IPSec, is handled on the gateway system by a daemon process that listens for connections on a particular port (UDP 500 for IKE). In the case of each of the implementations tested, fingerprinting the IKE daemon was analogous to fingerprinting the IPSec implementation – or the OS, for that matter, since each OS has its own IKE daemon. Linux is the notable exception. There are currently two IKE daemons available for the Linux 2.6 platform: *isakmpd* and *racoon. isakmpd* is the OpenBSD ISAKMP/IKE daemon, which has been

ported to Linux, and *racoon* is the ISAKMP/IKE daemon developed by the KAME (2000) project. Both make use of the IPSec implementation on the 2.5.x and 2.6.x Linux kernels. The Linux tests were conducted using the 2.6.8.1 kernel. Due to space constraints, results from the Linux *isakmpd* implementation are not discussed in this paper.

At the outset of the testing phase, it was decided that IKE implementations of three OSs would be observed, and that the IPSec tunnels would be configured to use pre-shared keys (PSKs) and x.509 certificates (*certs*) as per the ICSA Labs recommendations. Analysing the traffic showed that the ISAKMP/IKE exchanges differed little whether PSKs were used, or certs. The one clear difference was the *Authentication Method* type: it was PSK(1) when pre-shared keys were used, RSA-SIG(S) when certs were used in all OSs other than Solaris 9, and RSA-ENC(4) when certs were used in Solaris 9. Due to the fact that the results of using PSKs were near identical to when certs were used, the findings discussed below are only those of ISAKMP/IKE tunnels configured using x.509 certificates.

Following this test phase, 18 discriminants were identified, a combination of which can be used to single out each IPSec implementation. All the discriminants are detailed below, categorized into the three steps that were taken to find them, as mentioned at the close of Section 3. Thereafter, they are summarized in Table 1. Each discriminants is explained in a subsection below.

Table 1.

IKE/IPSec fingerprint table

	Windows 2003	**Solaris 9 x86**	**Linux 2.6 *racoon***
DF bit set (in IP header)	No	Yes	Yes
Any message in Key Exchange is fragmented	Yes MM5 and MM6	No	No
Proposal Payload SPI Size = 0	Yes	No (8)	Yes
Commit Bit set in QM	Yes (QM2)	No	No
Use of Vendor-ID payload	Yes (4)	No	No
ISAKMP Info. between MM and QM	No	Yes (1)	Yes (2)
ISAKMP Info. at tunnel tear-down	Yes (4)	No	Yes (2)
ISAKMP Info. sent from Initiator to	Yes	No (R to I)	Yes

	Windows 2003	**Solaris 9 x86**	**Linux 2.6 *racoon***
Responder (I–R)			
Three messages in QM as per RFC 2409	No (4)	Yes	Yes
ESP traffic before QM completion	No	Yes	No
Unique order of SA Attribute Type value in Transform Payload of MM1 (TAVO fingerprint)	No [1,2,4,3,11,12] same as Windows 2000	Yes [3,2,1,4,11,12]	Yes [11,12,1, 3,2,4]
Unique number of payloads in MM4	Yes [4,10,7,130,130]	Yes [10,4,5]	No [4,10,7]
Key Exchange (4) is first payload type in MM3 and MM4	Yes	No; MM3: hash (8), MM4: nonce (10)	Yes
Identification (5) is first payload type in MM5 and MM6	Yes	No hash (8)	Yes
Certificate Request (7) payload type is visible in MM4	Yes	No UNKNOWN-ID-TYPE: 155,151,55	Yes
Encrypted *ping* discernible	Yes (108 bytes)	Yes (108 bytes)	Yes (108 bytes)
Encrypted TCP handshake discernible	Yes (84B, 84B, 76B)	Yes (84B, 84B, 76B)	Yes (84B, 84B, 76B)
Encrypted TCP Close discernible	Yes (76B each)	Yes (76B each)	Yes (76B each)

It should be noted that for brevity, the initials MM and QM have been used below and in Table 1 as per the ICSA Labs recommendations (ICSA-Labs, 2003), to denote IKE Main

Mode (Phase 1) and IKE Quick Mode (Phase 2), respectively. The message number in each exchange, where the discriminant was observed, is appended to the initials; that is, MM4 refers to the fourth message in the Main Mode exchange, QM2 refers to the second message in the Quick Mode exchange, and so on.

4.1. Discriminants from connection summary

In order to study the connection summary, the Ethereal traffic analysis tool was used. The discriminants are described below, italicized to correspond to the entries in Table 1.

To begin, the test traffic was analysed for the presence or absence of the *DF bit in IP header*. IP fragmentation is unsuitable or undesirable in some environments; this is due to the computing overhead required to fragment and reassemble an IP datagram, the memory required to buffer the fragments until they can be reassembled, and the need to retransmit the entire IP datagram if any fragment is lost (Cisco Systems, Inc., 2004). Whenever IP fragmentation is not desired, the Don't Fragment (DF) bit on the IP header can be set.

In each OS tested, the connection summary was consulted to establish whether there were *ISAKMP Informational messages sent between Main Mode and Quick Mode*. Implementing ISAKMP Informational messages is a requirement, as per RFCs 2408 and 2409, but initiating an ISAKMP Informational exchange after Main Mode and before Quick Mode is merely allowed as an option, and its adherence or non-adherence by a vendor does not place it in violation of the standard.

The connection summary for each OS was reviewed for *ISAKMP Informational messages sent at tunnel tear-down*. As per the RFCs, sending an ISAKMP Informational message when the IPSec tunnel is being torn down, is allowed, and not in violation of the standard. Sending one or more messages in response to the ISAKMP Informational messages is, however, discouraged by the standard.

In several places within RFCs 2407, 2408, and 2409 reference is made to the fact that ISAKMP Informational messages can be sent by either the Initiator or the Responder. Two such references are “once established, *either party* may initiate Quick Mode, Informational, and New Group Mode Exchanges” (Harkins and Carrel, 1998) (Author's emphasis), and “When creating a Notification Payload, the transmitting entity (*Initiator or Responder*) MUST do the following…” (Maughan et al., 1998) (Author's emphasis). *ISAKMP Informational messages originating from either the Initiator or from the Responder* are therefore allowed, and not in violation of the standard. The presence and direction of these messages were observed in the connection summary of each OS.

Connection summaries from all OSs were checked to observe *the number of messages in Quick Mode*. All diagrams describing Phase 2 (Quick Mode) in RFC 2409 depict it as consisting of three messages, in a request–response pattern, originating from the Initiator; and as per the RFCs, IKE exchanges consist of a fixed number of messages. Therefore,

an IKE/IPSec implementation employing anything other than three messages for IKE Phase 2 Quick Mode, would be in violation of the standard.

The connection summary from each OS test was examined for *encrypted exchanges prior to the completion of phase negotiation*. Referring to the use of the Commit Bit in either Main Mode or Quick Mode exchanges, RFC 2408 says that it "is used to ensure that encrypted material is not received prior to completion of the SA establishment" (Maughan et al., 1998), suggesting that it is undesirable that encrypted communication should take place before the (successful) completion of the phase negotiation. From this, it is assumed that transmitting encrypted material during a phase negotiation, which is not implicitly part of the exchange, is not in violation of the standard, but could constitute bad practice.

In the case of each OS tested, the connection summary was examined for *IP fragmentation during phase negotiation*. Since the RFCs make no mention of fragmentation, and more specifically, no mention of fragmentation during phase negotiation, it can be assumed that an implementation allowing fragmentation is not in violation of the standard, but in the view of secondary literature (Cisco Systems, Inc., 2004) it can be seen as undesirable, and could be considered bad practice. Although the secondary literature referenced only makes mention of fragmentation of IPSec traffic in general, this reservation can be more specifically applied to phase negotiation.

4.2. Discriminants from IKE exchange

In order to study the IKE exchange, Ethereal was used, paying attention to each IP datagram's payload.

It is understood from RFC 2408 that any value for SPI Size is valid, but given the explanation of MUST in RFC 2119 (that MUST denotes an absolute requirement of the specification), an implementation would be in violation of the standard if it does not ignore the contents of the SPI field in the event of the *SPI Size value being anything other than zero*. The value of the SPI Size observed in each implementation formed a discriminant.

As per RFC 2408, *the use of the Commit Bit* is optional. However, in the event that it be implemented, it is an absolute requirement that the party which did not set the Commit Bit has to wait for an ISAKMP Informational message containing the CONNECTED Notify Message, and can therefore be considered in violation of the standard if it does not do so. This behaviour was observed, and each vendor's peculiarities formed a discriminant.

The definition of the Vendor-ID payload in RFC 2408 makes it clear that the *use of Vendor-ID payloads* is optional. The presence or absence of the Vendor-ID payload was seen as a discriminant.[3]

Attribute Values are classified as either Basic (B) or Variable Length (V), and the only restriction imposed by RFC 2407 is that “Attributes described as basic MUST NOT be encoded as variable. Variable length attributes MAY be encoded as basic attributes if their value can fit into two octets.” Furthermore, in RFC 2408 the SA Attributes field is depicted as variable in length, suggesting that any number of attributes can be included. RFC 2407 adds the only limitation, stating that “An SA Life Duration attribute MUST always follow an SA Life Type which describes the units of duration.” Sample Attribute Classes, Values and Types can be seen in Table 2. The *specific order of SA Attribute Type values in the MM1 Transform Payload* was used as a discriminant.

Table 2.

SA Attribute Classes, Values and Types, as per RFC 2409 (Harkins and Carrel, 1998)

Class	**Value**	**Type**	**Class**	**Value**	**Type**
Encryption algorithm	1	B	Group curve A	9	V
Hash algorithm	2	B	Group curve B	10	V
Authentication Method	3	B	Life type	11	B
Group description	4	B	Life duration	12	V
Group type	5	B	PRF	13	B
Group prime/irreducible polynomial	6	V	Key length	14	B
Group generator one	7	V	Field size	15	B
Group generator two	8	V	Group order	16	V

The task of ordering the SA Attribute Type values can then be understood to have been left to the implementers, the only absolute condition being that the Life Type attribute must always be followed by a Life Duration attribute.

RFC 2408 states “An ISAKMP message has a fixed header format … followed by a variable number of payloads.” RFC 2409 further states “The SA payload MUST precede all other payloads in a Phase 1 exchange. Except where otherwise noted, there are no requirements for ISAKMP payloads in any message to be in any particular order.” RFC 2408 later adds “While the ordering of payloads within messages is not mandated, for processing efficiency it is RECOMMENDED that the Security Association payload be the first payload within an exchange.”

It can thus be assumed that the ordering of ISAKMP Payloads is left up to the vendor, but that there "may exist valid reasons in particular circumstances" to position the SA payload as the first one. This *ordering of ISAKMP payloads in MM4* formed the basis for a discriminant.

The diagrammatic description of "IKE Phase 1 authenticated using signatures" in RFC 2409 shows that in the second message from the Initiator, the second item is the Key Exchange, suggesting that this should be the first payload in MM3. (Indeed, this is how most vendors interpreted it.) RFC compliance may therefore be understood by a vendor as depending on 'Key Exchange' being the first payload in MM3. The ambiguity created by the RFC regarding this topic, however, may lead a vendor to argue that a certificate is an optional case in Phase 1 as a revised mode of public key encryption. As will be shown later, all but one vendor interpreted the first meaning to be correct.

This extends to the use of 'Key Exchange' as the first payload of the second message from the Responder (MM4) also. If Section 5.1 of RFC 2409 was followed, then 'Key Exchange' would be the first payload, otherwise 'Nonce' would be the first payload. Indeed, both these possibilities could be argued to be RFC-compliant. The *first payload type in MM3 and MM4 being Key Exchange (4)* was used as a discriminant.

The argument presented above also applies to 'Identification' as the first payload of the third message from the Initiator as well as that of the Responder (MM5 and MM6). If Section 5.1 of RFC 2409 was followed, then 'Identification' would be the first payload, but if Section 5.3 was followed, 'HASH' would be the first payload. The *first payload type in MM5 and MM6 being Identification (5)* was used as a discriminant.

The same is true for the 'Certificate Request' payload. If the authentication is viewed as consisting of x.509 certificates, then a 'Certificate Request' should be visible in MM4, but if the authentication is viewed as a special case of a revised mode of public key encryption, then MM4 would contain <Nr_b>PubKey_i, <KE_b>Ke_r, <IDir_b>Ke_r, as per RFC 2409.[4] Whether or not a *Certificate Request payload type (7) was visible in MM4* was taken as a discriminant.

4.3. Potential discriminants from ESP traffic

ESP-protected traffic is encrypted, and it would seem logical that its perusal should reveal nothing. This proved not to be the case, however. Analysing the ESP-protected traffic resulted in successfully identifying patterns within the encrypted stream, which corresponded to the plaintext traffic. This could, to a small measure, defeat the purpose of encrypting the traffic in the first place. These results contributed to fingerprinting in that they partially laid bare the behaviour of each implementation's TCP/IP stack, irrespective of encryption. At this stage, these should be regarded as partial discriminants. Observing the behaviour of other IKE/IPSec implementations on other platforms will determine whether these *partial* discriminants can be considered *complete* discriminants in the future. These tests nevertheless showed that wrapping the contents of an IP datagram in cryptography does not necessarily hide its contents.

The tests conducted showed that when running ping without command-line options across the IPSec tunnel, the ESP-enabled (encrypted) ICMP communication appeared at regular intervals on the network, and that its datagram size was consistently unchanged (108 bytes). These results raise two important points. Firstly identifying encrypted traffic as being an ICMP ECHO REQUEST or ECHO REPLY opens the door to the possibility that the contents of other types of encrypted traffic may also be identified (and thus potentially lead to the mounting of a *known-plaintext* attack). Secondly, when working through large network traces looking for anomalous traffic patterns, the ability to isolate any single portion of the traffic, aids the investigator by narrowing down the search space. In a real-world scenario, bits of information appearing at near-regular intervals can be routing protocol updates, for example. The *discernible encrypted ping* was taken as a partial discriminant.

Either by observing whether the DF bit is set or not, or by hypothesizing that the messages exchanged during the tests are small enough not to need fragmentation, it can be inferred that an encrypted TCP handshake would consist of three distinct packets, of near-close or identical size. The tests conducted showed that the TCP handshake was easily identifiable among the encrypted traffic, and that the size of the three messages (84 bytes for the first two messages, and 76 bytes for the third) was consistent across all platforms tested. The *identifiable TCP handshake* was taken as a partial discriminant.

Analysis of the encrypted traffic showed the four messages of a TCP Close[5] clearly discernible in all of the implementations tested. The size of these messages was also consistent throughout all implementations: 76 bytes each. Although the tests conducted did not take into account the possibility of inferring TCP Close by observing the 2MSL timeout, it is likely that in real-world TCP traffic, the 2MSL timeout will be clearly visible amidst the encrypted traffic, and that it will help in pinpointing the location of the TCP Close. The *identifiable TCP Close* was taken as a partial discriminant.

5. Fingerprint of IKE/IPSec implementations

What follows, is the collection of discriminants which form the fingerprint for each IKE/IPSec implementation.

Tests showed that the order of SA Attribute Type values in the Transform Payload of MM1 can alone serve as a fingerprint. This discriminant, christened the *TAVO fingerprint* (for Transform-payload Attribute Value Order), is also presented for each of the platforms tested by indicating the order of the respective attributes listed in Table 2.

This section closes with a table representing all fingerprints.

5.1. Microsoft Windows 2003 enterprise server

The most significant finding resulting from the analysis of the Microsoft 2003 IKE/IPSec implementation was that the Phase 2 (Quick Mode) consisted of four messages instead of

the RFC-defined 3. This is a major deviation from the standard, and is likely to cause interoperability problems with other IPSec implementations.

The contents and purpose of this fourth Quick Mode message (QM4) were explained in *The Cable Guy* (Davies, 2002), a Microsoft TechNet publication: QM4 contains a NOTIFICATION from the Responder, the payload of which is a CONNECTED notification message. “[T]his message, which is used by IPSec peers running Windows XP or Windows 2000, is not required by the IKE standard. It is used to prevent the Initiator from sending IPSec-related packets to the Responder before the Responder is ready to receive them.” The standard offers a solution to this problem: use of the Commit Bit, which RFC 2408 states is used “to ensure that encrypted material is not received prior to completion of the SA establishment”. The standard further states that in the event that the Commit Bit is set, “the entity which did not set the Commit Bit MUST wait for an Informational Exchange containing a Notify payload (with the CONNECTED Notify Message) from the entity which set the Commit Bit”. The Microsoft Windows 2003 implementation sets the Commit Bit, but does not send the ISAKMP Informational message containing the CONNECTED notify message, rather choosing to communicate the CONNECTED message as the payload of a fourth message to Quick Mode.

These two choices, namely the use of the Commit Bit without the accompanying ISAKMP Informational, and the addition of a fourth message to Quick Mode, place this implementation in violation of the official IETF standard on two accounts. Although untested, it is likely that these two digressions from the standard may cause interoperability problems when trying to establish an IPSec VPN tunnel between a Microsoft gateway and another vendor's gateway. That said, it should also be noted that the net result of Microsoft's IKE/IPSec implementation achieves its purpose by means of this alteration to the standard and works well in practice; meaning that when setting up an IPSec tunnel between two peer Windows 2003 machines, no IPSec-encrypted traffic is exchanged before IKE Phase 2 Quick Mode is completed. Given that our tests showed other implementations which did not achieve this goal, it is commendable that Microsoft did, albeit by digressing from the standard.

For the purpose of fingerprinting, it can be noted that the Microsoft implementation was the only one which made use of the Commit Bit; it was also the only one which made use of the Vendor-ID payload. The Windows 2003 implementation contained four Vendor-ID payloads in MM1, namely: MS NT5 ISAKMPOAKLEY, Microsoft L2TP/IPSec VPN Client, draft-ietf-nat-t-ike-02, and Ox26244d38eddb61b3172a36e3d0cfb819 – the latter of which could not be decoded by Ethereal. Windows 2003 was also the only OS not to have the DF bit set in its IPSec communication. As a result of this, the last two messages in Phase 1 Main Mode (MM5 and MM6) were fragmented. The TAVO fingerprint for Windows 2003 is: [1,2,4, 3,11,12].

5.2. Sun Microsystems Solaris 9 x86

Sun Microsystems Solaris 9 for x86 presented a range of variations that set it apart from the other implementations, and was the only implementation holding different values to the rest in the case of five discriminants.

The Solaris 9 for x86 IKE/IPSec implementation was the only one holding a Proposal Payload SPI Size not equal to 0 (its SPI value size was 8). It was also the only one not sending ISAKMP Informational messages at tunnel tear-down. This, however, may be attributed to the design of IPSec on Solaris 9, which assumes the commencement of IPSec-encrypted traffic to be the machine bootstrap process. In contrast, other implementations allow the IKE daemon (Linux) or IPSec service (Windows) to be terminated or disabled – which results in ISAKMP Informational messages being issued. Solaris 9 was also the only implementation where ISAKMP Informational messages were sent from Responder to Initiator, the inverse of all the others.

To their credit, Solaris 9 implementers may well have been following RFC recommendations when choosing to interpret x.509 certificates as a special case of the revised mode of public key encryption. Referring to the advantages of authentication using the revised mode of public key encryption over that of authentication with signatures, RFC 2409 states “This authentication mode retains the advantages of authentication using public key encryption but does so with half the public key operations” and further “This solution adds minimal complexity and state yet saves two costly public key operations on each side. In addition, the Key Exchange payload is also encrypted using the same derived key. This provides additional protection against cryptanalysis of the Diffie-Hellman exchange” (Harkins and Carrel, 1998) suggesting that it may be the better option.

Two other peculiarities displayed by the Solaris 9 implementation were the fact that the tunnel set up showed IPSec-encrypted (ESP) communication being sent from Initiator to Responder prior to the completion of Phase 2 (Quick Mode). There was also one ISAKMP Informational message sent between the completion of Main Mode and prior to the start of Quick Mode. The TAVO fingerprint for Solaris 9 x86 is: [3, 2,1,4,11,12].

5.3. Racoon on Linux kernel 2.6

Racoon is part of the KAME (2000) project. Of the two IKE/IPSec implementations that were tested for Unix/Linux environment (Solaris 9 x86 and Linux *racoon*), *racoon* was the most straight-forward and easy to configure. It was the only non-Windows implementation *not* to allow encrypted communication prior to QM completion, and the ordering of payloads in MM4 was 4, 10, 7, similar to Windows 2003s 4, 10, 7, 130, 130. Racoon was the implementation which had the most attributes in common with those of other platforms, suggesting willing adherence to the standard. The TAVO fingerprint for *racoon* on Linux 2.4.6.8.1 is [11,12,1,3,2,4].

6. Conclusion

Fingerprinting can be thought of as a special case of traffic analysis. For an attacker, this could be a means to by which narrow down the search space for exploitable weaknesses on the target system. For forensic examiners, it could provide circumstantial evidence that would help determine *how* an attack was carried out, and how the vulnerable systems were identified, and could help to reconstruct a sequence of events that resulted in the cyber crime under investigation. Studying the research findings of studies in fingerprinting, could aid a software developer interested in adhering as closely as possible to the protocol standards, and in designing a secure system that leaks as little unnecessary information as possible: to coin a phrase, a *reticent communicator*.

The results obtained as a product of this present research contribute to the field of information gathering, specifically to the subject of traffic analysis, by exposing a new avenue of information gathering, namely that of fingerprinting the endpoints of a VPN tunnel. This study extracted discriminants from the IPSec VPN connection summary, from the IKE Phase 1 and Phase 2 exchanges, and pointed to three instances of encrypted traffic leaking traffic information. Tests were conducted on three operating systems, using three IKE/IPSec implementations to provide empirical data, and the positive outcome of these test suggests that further research is likely to produce finer-grained fingerprints. In addition to the discriminants extracted, which can be used in a decision tree in order to isolate any of the three implementations, a singular fingerprint was also presented, namely the Transform-payload Attribute Value Order (or TAVO) fingerprint, which can be used to uniquely identify an IKE/IPSec implementation in the OSs tested in this study. The complete findings, a subset of which was discussed in this paper, can be found in Izadinia (2004).

Future work on the topic of fingerprinting IKE/IPSec implementations can be undertaken in various ways. One, is to look for more discriminants, by combing through the RFCs which comprise the IPSec standard. Another potential avenue for research is the study of IKE/IPSec implementations in hardware VPN devices. A related area for future research is that of ways to counter fingerprinting, that is: how the discriminants can be rendered ineffectual.

References

Aqun et al., 2000 Z. Aqun, Y. Yuan, J. Yi and G. Guanqun, Research on tunneling techniques in virtual private networks, *Correspondence Journal* **21** (2000) (6) Note: Journal entry may not be accurate [Original in Chinese, translated by babelfish.altavista.com].

Arkin, 2001 Arkin O. ICMP usage in scanning: the complete know-how, version 3; 2001.

Arkin and Yarochkin, 2001 O. Arkin and F. Yarochkin, ICMP based remote OS TCP/IP stack fingerprinting techniques, *Phrack Magazine* **11** (2001) (57) [File 7 of 12].

Beverly, 2004 R. Beverly, A robust classifier for passive TCP/IP fingerprinting, *Passive and active network measurement, 5th international workshop, PAM 2004, Antibes Juan-les-Pins, France, April 19–20, 2004, Proceedings, Lecture notes in computer science* **vol. 3015**, Springer (2004).

Bordet, 2002 Bordet J. Remote SMTP server detection; 2002.

Broido and Claffy, 2001 Broido A, Claffy K. Internet topology: connectivity of IP graphs. In: Proceedings of SPIE international symposium on convergence of IT and communication; 2001.

Cisco Systems, Inc., 2004 Cisco Systems, Inc., IP fragmentation and PMTUD Document ID: 25885. Tech. rep., Cisco Systems (2004).

<concept@ihug.com.au>, 2000 <concept@ihug.com.au>. Detection with ARP. Napalm e-zineNote: the source for this reference was not readily available, however it was traced with help from ouah_@hotmail.com and datacide@beefed.org; 2000.

Davies, 2002 Davies J. IKE Negotiation for IPSec Security Associations. The cable guy; June 2002 [a Microsoft TechNet publication].

Dunbar, 2001 N. Dunbar, IPSec networking standards – an overview Tech. rep., Hewlett-Packard Company, Infrastructure Strategic Engineering Division (2001).

f0bic, 2001 f0bic. Examining remote OS detection using LPD querying; 2001.

Harkins and Carrel, 1998 D. Harkins and D. Carrel, RFC 2409 – the Internet key exchange (IKE) Tech. rep., IETF (1998).

Hills, 2003 Hills R. NTA monitor UDP backoff pattern fingerprinting white paper. Tech. rep. NTA; 2003.

Huffaker et al., 2001 B. Huffaker, D. Plummer, D. Moore and K. Klaffy, Topology discovery by active probing Tech. rep., Cooperative Association for Internet Data Analysis (2001).

ICSA-Labs, 2003 ICSA-Labs, IPSec technical product configuration guidelines Tech. rep., ICSA Labs – a division of TruSecure Corporation (2003).

Izadinia, 2004 Izadinia VD. Fingerprinting encrypted tunnel endpoints. MSc. dissertation. South Africa: Computer Science Department, University of Pretoria; 2004.

KAME, 2000 KAME. Racoon manual pages – racoon(8), racoon.conf(5), setkey(8); 2000.

Koblitz, 1994 N. Koblitz, A course in number theory and cryptography (2nd ed.), *Graduate texts in mathematics* **vol. 114**, Springer Verlag (1994).

Lee et al., 2002 D. Lee, J. Rowe, C. Ko and K. Levitt, Detecting and defending against web-server fingerprinting, *Proceedings of the 18th annual computer security applications conference, 2002*, IEEE Computer Society (2002).

Lowekamp, 2003 Lowekamp B. Combining active and passive network measurements to build scalable monitoring systems on the grid. ACM SIGMETRICS Performance Evaluation Review 30; 2003.

Maughan et al., 1998 D. Maughan, M. Schertler, M. Schneider and J. Turner, RFC 2408 – Internet security association and key management protocol (ISAKMP) Tech. rep., IETF (1998).

Nazario, 2000 Nazario J. Passive system fingerprinting using network client applications; 2000.

Perlman and Kaufman, 2000 R. Perlman and C. Kaufman, Key exchange in IPSec: analysis of IKE, *IEEE Internet Computing* **4** (2000) (6).

Schneier, 1996 B. Schneier, Applied cryptography (2nd ed.), John Wiley & Sons (1996).

Schneier and Ferguson, 2003 B. Schneier and N. Ferguson, Practical cryptography (1st ed.), John Wiley & Sons (2003).

Veysset et al., 2002 F. Veysset, O. Courtay and O. Keen, New tool for remote operating system fingerprinting Tech. rep., Intranode Software Technologies (2002).

Yarochkin, 1999 Yarochkin F. Remote OS detection via TCP/IP stack fingerprinting; 1999.

[1] This statement assumes that by designing the IPSec link in this way, the enterprise is agreeing to allow the IPSec traffic through the firewall. It should also be noted for completeness that IPSec traffic can be identified as such by the Encapsulation Security Payload (ESP) or Authentication Header (AH) headers following the regular IP headers. IPSec traffic is therefore not *indistinguishable* from regular IP traffic, but harmless content in legitimate IPSec packets *is* indistinguishable from malicious content.
[2] In UNIX and Linux machines this was done by sending the IKE daemon a TERM signal, and in Windows machines this was done by deactivating the IPSec policy.

[3] On a related note, if all IKE/IPSec implementations made use of the Vendor-ID payload, containing a vendor-specific string, this could by itself, constitute a fingerprint.
[4] The label Nr represents the Responder's nonce, PubKey_i is the Initiator's public key, KE is the Key Exchange payload, Ke_r is the key to the symmetric encryption algorithm negotiated earlier, IDir is the Initiator and the Responder's identification, the _b denotes the body of the payload, and the <x>y notation means x is encrypted using y.
[5] FIN + ACK, ACK, FIN + ACK, ACK.

Vitae

Jan Eloff is Head of Department and full professor in Computer Science at the Department of Computer Science, University of Pretoria. He is also Chair of the School of Information Technology. He is an expert member of Technical Committee 11 (Information Security) of the International Federation for Information Processing (IFIP). In 2001 he received the IFIP Silver Core and Outstanding Services Award for his long-term services to IFIP. He also serves as the South African representative on ISO (International Standards Organisation) contributing to the development of computer and information security standards. In October 2004 he was elected as the President of the South African Institute of Computer Scientists and Information Technologists (SAICSIT). He has published extensively in a wide spectrum of accredited international subject journals. Many acclaimed international and national conferences were organised and conducted under his leadership. He is an evaluated researcher from The National Research Foundation (NRF), South Africa. He is a member of the Council for Natural Scientists of South Africa.

Derrick Kourie is a professor in the Department of Computer Science at Pretoria University. He acted as the main supervisor of the MSc project upon which this article is based. His principle areas of interest include software engineering, software by construction and various theoretical aspects of computer science.

Vafa Izadinia is a graduate in Computer Science from the University of Pretoria, in South Africa. He obtained a BSc Hons. in Computer Science in 1999, following which he worked in Belgium, before taking academic leave to further his studies. He completed an MSc in Computer Science from the University of Pretoria in 2005, and currently works and lives in Belgium.