Common VPN Security Flaws

Roy Hills, NTA Monitor Ltd.
http://www.nta-monitor.com/

January 2005

Abstract

This paper outlines some of the common VPN security flaws that NTA Monitor have found during the last three years while performing VPN security tests. The paper concentrates on remote access VPN configurations using the IPsec protocol, although some of the findings are also applicable to site-to-site VPNs.

Some of the problems that have been seen, such as the username enumeration issue, are new discoveries, while others are known limitations of the protocols, which are exposed due to poor configuration.

The paper shows that VPNs are far from the impenetrable systems that many people believe them to be, and that they can actually be the weak link in an otherwise secure system.

Contents

1 Introduction

In the three years since NTA Monitor started testing VPN security, they have tested many implementations from most of the major vendors. What has been found is quite shocking: most of the VPNs that were tested have had remotely exploitable vulnerabilities, and often these would allow an attacker to gain unauthorised access to the VPN, view or alter VPN traffic, or disrupt the VPN server.

Some of the vulnerabilities that have been discovered during the testing were previously unknown, and these were reported to the vendors in accordance with NTA Monitor's disclosure policy. After several such flaws had been reported, it was noticed that the same issues were occurring again and again in different vendors' products.

It was also found that the organisations being tested generally felt that their VPN was invisible and impenetrable, and that the VPN security testing was just a "tick in the box". After the testing they discovered that the VPN was actually the weakest point in their perimeter. These organisations now know the problems, and in most cases have fixed them. However it is suspected that many other organisations have vulnerable VPNs that they assume are secure.

This paper details the common VPN flaws and explains their root causes. It also gives examples using the *ike-scan* and *psk-crack* tools, which are part of the *ike-scan*[1] package, which demonstrate these flaws.

2 VPNs are Attractive Targets

Before discussing the issues themselves, it is worth pointing out that VPNs are attractive targets to hackers. The reasons include:

- VPNs carry sensitive information over an insecure network.
  The users generally trust the VPN to keep the information secure, which is understandable because that is what the VPN is designed to do. Because of this trust, the users will transfer sensitive data without using additional encryption, and use protocols that transmit authentication credentials in the clear.

- Remote Access VPNs often allow full access to the internal network.
  Many organisations configure their remote access VPNs to allow full access to the internal network for VPN users. This means that if the VPN is compromised, then the attacker gets full access to the internal network too.

[1] ike-scan is available from *http://www.nta-monitor.com/ike-scan/*

- VPN traffic is often invisible to IDS monitoring.
  If the IDS probe is outside the VPN server, as is often the case, then the IDS cannot see the traffic within the VPN tunnel because it is encrypted. Therefore if a hacker gains access to the VPN, he can attack the internal systems without being picked up by the IDS.

- Increasing security in other areas.
  As more organisations install firewalls, move Internet servers onto the DMZ, automatically patch servers etc., the VPN becomes a more tempting target.

3 Common VPN Flaws

3.1 VPN Fingerprinting

Most VPN servers can be fingerprinted either by UDP backoff fingerprinting[1], or Vendor ID fingerprinting. While this is not a problem by itself (and some vendors do not consider it a problem at all), it does give a potential attacker useful information.

Some systems will reveal the general type of device, e.g. "Cisco PIX" or "Nortel Contivity", whereas others will show the software version details as well. An example of the latter is given in
http://www.nta-monitor.com/news/checkpoint2004/index.htm [2].

3.2 Insecure Storage of Authentication Credentials by VPN Clients

Many VPN client programs offer to store some or all of the authentication credentials (e.g. username and password), and for some clients, this is the default setting. While this makes the VPN easier to use it also introduces security risks, especially if the credentials are not well protected.

The common client issues that have been seen are:

- Storing the username unencrypted in a file or the registry.
  Anyone with access to the client computer can obtain the username. If the VPN is using IKE Aggressive Mode, then knowledge of the username allows an offline cracking attack against the password. Figure 2 shows an example of the username *royhills@hotmail.com* stored in the registry.

- Storing the password in a scrambled form.
  This is often referred to as "encryption", but it is really obfuscation rather than encryption because there is no unique key needed to decrypt it. If the obfuscation algorithm becomes known, then it is a simple matter to obtain the password if you have access to the client computer. Figure 2 shows an example of an obfuscated password stored in the registry. In this case, the corresponding clear-text password is *W0ntGu355Th15*.

- Storing the plain-text password in memory.
  If storing an obfuscated version of the password in a file or registry is not bad enough, many clients decrypt this when they start up, and store a plain-text version of the password in memory. In this case, anyone with access to the client computer can obtain the password by starting the VPN client and then dumping the process memory with a tool such as *pmdump*, or crashing the computer to get a dump of physical memory. Figure 1 shows an example memory dump from a VPN client with the clear-text password *W0ntGu355Th15* highlighted. Notice that the last two characters of the password are repeated in the memory dump. This is repeatable behaviour for this VPN client, and may give some insight into the obfuscation mechanism.

- Weak registry or file permissions for stored credentials.
  It is a bad idea to cache credentials at all, but this is made worse if they are stored in a file or registry entry that is readable by everybody. This allows these details to be obtained from guest or anonymous network connections as well as via physical access to the client system.

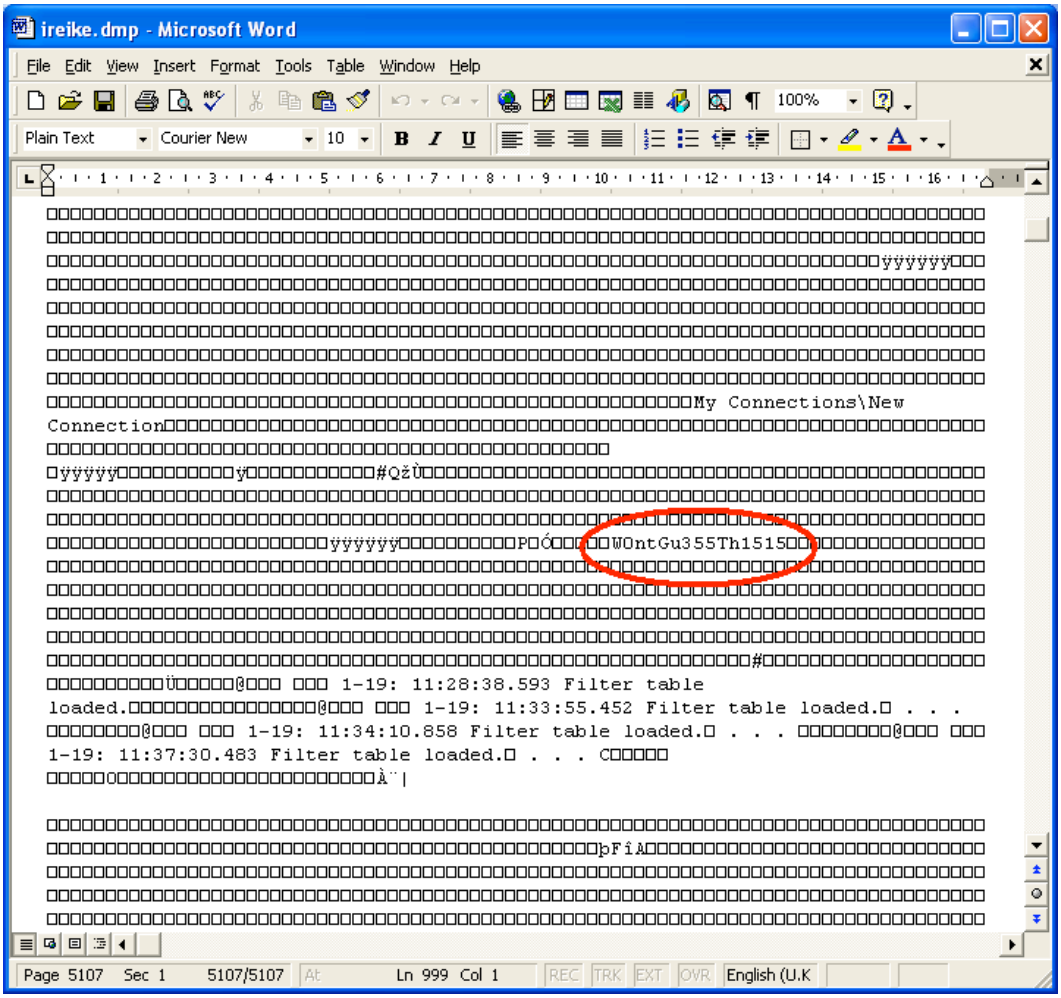

Figure 1: VPN Client Process Memory Dump Showing Plain Text Password

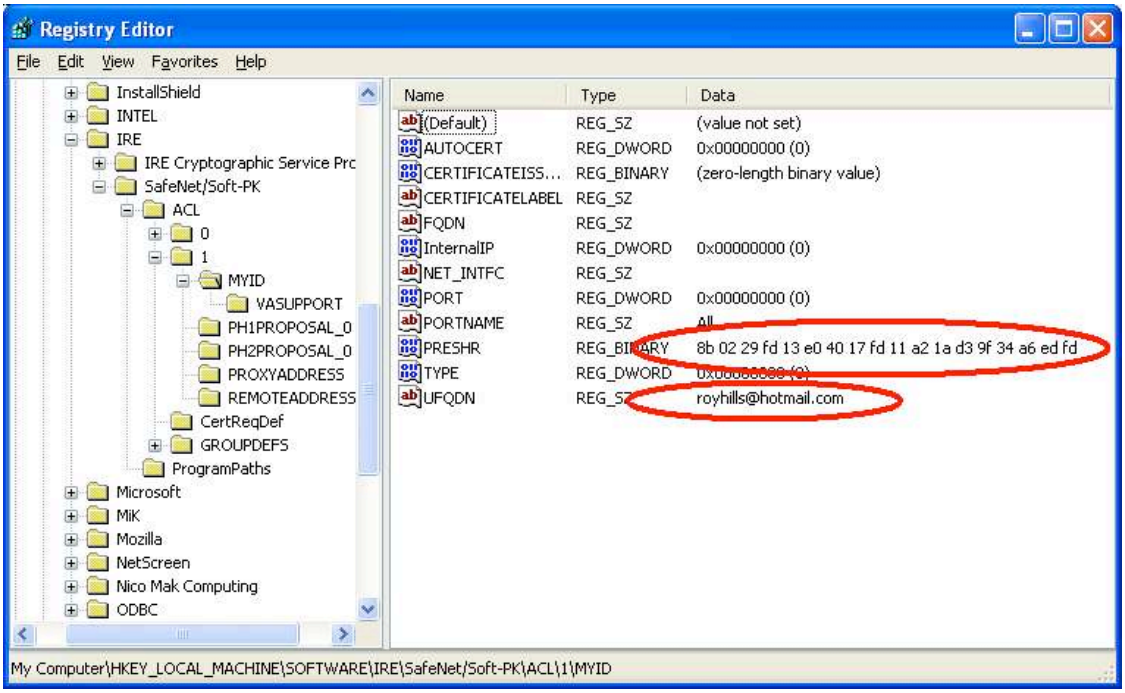

Figure 2: Username and Obfuscated Password Stored in Registry

3.3 Username Enumeration Vulnerabilities

Many remote access VPNs use IKE Aggressive Mode with pre-shared key (PSK) authentication as the default authentication method. The PSK authentication method is essentially the well-known username/password authentication scheme, although the terminology used can be different, for example the username is sometimes known as the *id* or *groupname*, and the password is sometimes referred to as the *secret* or *pre-shared key*.

One of the basic security requirements of a username/password authentication scheme such as this is that the response to an incorrect login attempt should not leak information about which of the credentials (username or password) was incorrect, because this would allow an attacker to deduce whether a given username is valid or not.

This requirement has been known for at least 25 years. The first known reference to this is in the November 1979 Morris Password Security paper[3] which discusses the authentication security of the Unix V7 operating system, which was released in January 1979. In this paper, he states:

> *To login successfully on the UNIX system, it is necessary after dialing in to type a valid user name, and then the correct password*

> *for that user name. It is poor design to write the login command in such a way that it tells an interloper when he has typed in a invalid user name. The response to an invalid name should be identical to that for a valid name.*
>
> *When the slow encryption algorithm was first implemented, the encryption was done only if the user name was valid, because otherwise there was no encrypted password to compare with the supplied password. The result was that the response was delayed by about one half second if the name was valid, but was immediate if invalid. The bad guy could find out whether a particular user name was valid. The routine was modified to do the encryption in either case.*

Although this security requirement has been known for decades, many implementations of PSK authentication do not abide by it and give a different response for an invalid username than for a valid one.

Figure 3 shows the initial packet exchange for aggressive mode PSK authentication. In this exchange, the client sends an IKE packet to the VPN server, and the VPN server responds with an IKE packet. Both packets contain several ISAKMP payloads, but the important ones for this discussion are the *Identity* payload sent by the client, which contains the username, and the *Hash* payload sent by the server, which is an HMAC[4] hash of various things including the password (pre-shared key)[2]. In a real authentication, the client would then respond with a third packet containing an HMAC hash of various things including the password, but this discussion is only concerned with the first two packets.

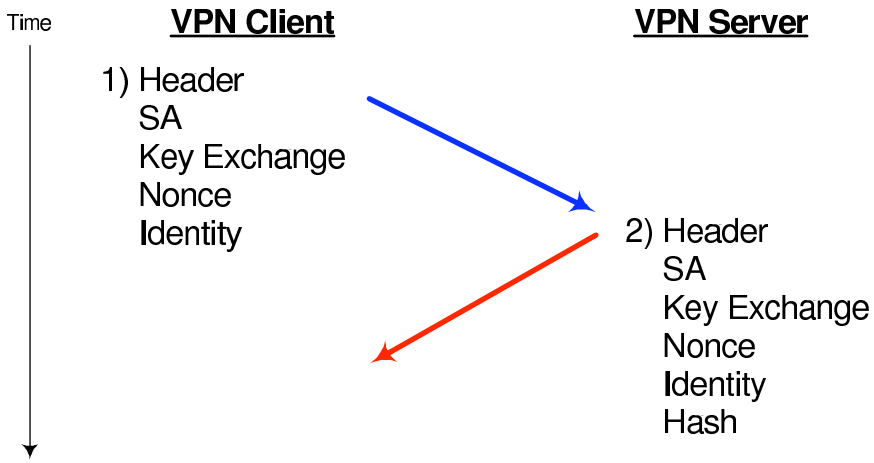

Figure 3: Packet Exchange for Aggressive Mode PSK Authentication

[2]The contents of the hash payload sent from the server are discussed in detail in section 3.4.

Three common faults were found in the way that VPN servers respond to the first packet from the client:

1. Some VPN servers only respond to the client if the username is valid, they do not respond at all to invalid usernames;

2. Some VPN servers will respond with a notification message, e.g. no-proposal-chosen, if the username is incorrect; and

3. Some VPN servers respond to both valid and invalid usernames, but the hash payload for invalid usernames is calculated using a null password, and it is simple for the client to determine this.

In all three cases, the response to an invalid username is different to that for a valid username, and this allows the client to determine if a given username is valid or not.

The correct way for the VPN server to respond to an invalid username is for it to respond using a random password for the hash payload. This is simple to implement, and does not allow the client to determine if a username is valid or not. It is therefore surprising that so many VPN implementations get this wrong.

An example of this issue is shown below. In this example, *ike-scan* is used to demonstrate that the VPN server responds to valid usernames normally, but to invalid usernames with a notify message. This shows that, for this VPN server, the username *fred* is valid, but the username *jim* is not.

```
$ ike-scan --aggressive --id=fred 172.16.2.2
Starting ike-scan 1.7 with 1 hosts (http://www.nta-monitor.com/ike-scan/)
172.16.2.2 Aggressive Mode Handshake returned
SA=(Enc=3DES Hash=SHA1 Auth=PSK Group=2:modp1024 LifeType=Seconds
LifeDuration(4)=0x00007080)
KeyExchange(128 bytes)
Nonce(20 bytes)
ID(Type=ID_IPV4_ADDR, Value=172.16.2.2)
Hash(20 bytes)

$ ike-scan --aggressive --id=jim 172.16.2.2
Starting ike-scan 1.7 with 1 hosts (http://www.nta-monitor.com/ike-scan/)
172.16.2.2 Notify message 14 (NO-PROPOSAL-CHOSEN)
```

This information leakage can be of great help to an attacker. Because usernames are often based on people's names or email addresses, it is easier than you would think to mount a successful dictionary attack. Once the first valid username is

found in this way, the format of the usernames is known, and it is even easier to find other valid usernames. During VPN security tests, it has often been possible to find many valid usernames in this way.

It is believed that this VPN username guessing issue is a new discovery, and NTA Monitor have notified several vendors about this issue. However, the vendors have not always implemented fixes after notification, so many systems are still vulnerable today.

3.4 Offline Password Cracking

Once a valid username is obtained using IKE Aggressive Mode, it is then possible to obtain a hash from the VPN server and use this hash to mount an offline attack to crack the associated password.

As shown in figure 3, the packet returned from the VPN server to the client contains, among other things, a *hash* payload which is known as the *responder hash* (because the client is the initiator and the VPN server is the responder) or $hash_r$.

The construction of $hash_r$ is defined by RFC 2409 [5] as:

$$hash_r = prf(skeyid, gx^r|gx^i|cky_r|cky_i|SAi_b|IDir_b)$$

and *skeyid* is defined as:

$$skeyid = prf(psk, Ni_b|Nr_b)$$

where the terms used are:

prf	The pseudo-random HMAC function
gx_r	The responder (VPN Server) public Diffie-Hellman value (in the key exchange payload)
gx_i	The initiator (VPN client) public Diffie-Hellman value (in the key exchange payload)
cky_r	The responder (VPN Server) ISAKMP cookie (in the ISAKMP header)
cky_i	The initiator (VPN client) ISAKMP cookie (in the ISAKMP header)
SAi_b	The body of the initiator (VPN client) SA payload
$IDir_b$	The body of the responder (VPN Server) ID payload
Ni_b	The body of the initiator (VPN client) nonce payload
Nr_b	The body of the responder (VPN Server) nonce payload

psk The Pre-Shared Key (group password)

All of the values above, apart from *psk*, are contained in the first initiator and responder IKE packets, which are not encrypted. So these values can be obtained from the packets and then the same functions used to calculate a hash using a pre-shared key of our choice and see if our calculated hash matches the one from the VPN server.

To perform the offline dictionary attack, a list of candidate passwords is taken, and each one is run through the hash function. The resulting hash is then compared with the hash that the server sent, and if they match then the correct password has been found. Because this is an offline attack, it does not cause any entries in the VPN server log, nor would it trigger account lockout. This attack is very fast: typically several hundreds of thousands of guesses per second. Some speed figures for the pre-shared-key cracking tool *psk-crack* are shown in table 1. The PSK cracking speed achievable depends mainly on the underlying hash performance: each PSK calculation consists of two HMAC calculations, and each HMAC calculation consists of two hash calculations (either MD5 or SHA1 depending on the hash algorithm used), therefore the PSK cracking speed should be approximately one quarter of the hash speed.

CPU type and speed	**MD5 attempts per second**	**SHA1 attempts per second**
Intel P3, 1.13GHz	153,000	88,000
Intel P4, 2.8GHz	264,000	136,000
AMD Athlon XP 2800+	315,000	212,000

Table 1: psk-crack Cracking Speeds

Table 2 shows the maximum time required for a brute-force attack against various password complexities using a single AMD Athlon XP 2800+ system.

Password Complexity	**Number of Combinations**	**Brute Force Time**
6 characters a-z	309 Million	16 minutes
6 characters a-z, A-Z, 0-9	57 Billion	2 days
8 characters a-z	209 Billion	8 days
8 characters a-z, A-Z, 0-9	218 Trillion	22 years

Table 2: Time required for brute-force cracking

Below is an example that shows how *ike-scan* can be used with a valid username (in this case the username *fred*, which we found earlier) to obtain the PSK parameters and write them to a file (fred.psk). The example also shows how *psk-crack* can be used to perform a dictionary attack against these PSK parameters to obtain the password.

```
$ ike-scan --aggressive --id=fred --pskcrack=fred.psk 172.16.2.2
Starting ike-scan 1.7 with 1 hosts (http://www.nta-monitor.com/ike-scan/)
172.16.2.2 Aggressive Mode Handshake returned
SA=(Enc=3DES Hash=SHA1 Auth=PSK Group=2:modp1024 LifeType=Seconds
LifeDuration(4)=0x00007080)
KeyExchange(128 bytes)
Nonce(20 bytes)
ID(Type=ID_IPV4_ADDR, Value=172.16.2.2)
Hash(20 bytes)

$ psk-crack fred.psk
Starting psk-crack in dictionary cracking mode
key "Liverpool" matches SHA1 hash 1f074be2ce5a3128aea49a4f4fb7752f9fe33670
Ending psk-crack: 10615 iterations in 0.052 seconds (204134.62 iterations
/sec)
```

Once a valid username and the associated password have been obtained, we can use this to complete IKE Phase-1 and establish an ISAKMP SA with the VPN Server. For some VPN servers, this is all that is required to gain access to the VPN, while others have an additional authentication step such as XAUTH[6] that must be completed after IKE Phase-1.

3.5 Man-in-the-Middle Attacks

If the VPN server is using IKE Aggressive Mode, and it is possible to determine a valid username and password, then an ISAKMP SA can be established to the VPN server. Even if the VPN server enforces a second level of authentication, this often relies on the security of this ISAKMP SA. In this case, if it is possible to establish an ISAKMP SA then the second level of authentication would not provide complete protection because it would be vulnerable to a man-in-the-middle attack. This risk is acknowledged in the XAUTH IETF draft[6]:

> *The protocol described in this memo strictly extends the authentication methods described in [IKE]. It does not in any way affect the authenticated nature of the phase 1 security association. In fact, this protocol heavily relies on the authenticated nature of the phase 1 SA. Without complete phase 1 authentication, this protocol does not provide **any** authentication at all, since it becomes easily vulnerable to Man-in-the-Middle (MitM) attacks.*

An example scenario showing how this could be exploited against a VPN server using XAUTH is given below:

1. Install the MitM system in the path of the VPN Client/Server traffic.
   Installing the system on an Ethernet link that the traffic flows over, and using ARP spoofing to re-direct the traffic could achieve this.
   In this position, the MitM system could sniff the usernames (which are passed in the clear) and crack the passwords using the information in the 1st and 2nd packets. Alternatively, it could be fed a list of usernames and passwords that had previously been obtained by group name enumeration and password cracking.

2. When the real client connects, allow them to establish an ISAKMP SA to the MitM system, and establish a second ISAKMP SA from the MitM system to the VPN server.
   The client user thinks he is connected to the VPN server, but is really connected to the MitM system.
   An ISAKMP SA can be established from the MitM system to the VPN server because the username and password are known.

3. The VPN server will issue an XAUTH challenge to the MitM system. The MitM system passes this on to the client.

4. The client sends the response (e.g. second username and SecurID PIN + passcode) to the MitM system, which passes it on to the VPN server.

5. Now the client is connected to the MitM system, and the MitM system is fully authenticated to the VPN server.
   At this point, the VPN security is breached. The MitM system has three options:

   (a) Intercept and log traffic between the client and VPN server;

   (b) Alter traffic between the client and VPN server; or

   (c) Drop the connection with the client and proceed to complete IKE Phase-2 with the VPN server and gain access to the internal resources.

3.6 Lack of Account Lockout

Most general purpose operating systems allow accounts to be locked out after a certain number of incorrect login attempts. However, many VPN implementations do not support this and allow an unlimited number of login attempts.

Why this should be the case is not clear because account lockout, like the prevention of username information leakage mentioned in section 3.3, has been a recognised part of authentication for decades (for example, it was used on VAX/VMS systems in the 1980s).

3.7 Poor Default Configurations

All too frequently, the default "out of the box" configuration for a VPN server is geared towards usability rather than security. Typically the default authentication method is IKE Aggressive Mode with pre-shared keys, even when stronger authentication methods such as Main Mode with certificates are available. IKE Aggressive Mode with pre-shared key authentication has known issues, some of which have been detailed in previous sections.

The end-users generally assume that the default configuration is secure because they trust the vendor to choose sensible defaults, and there is nothing to indicate to them that there is any security vulnerability unless they get tested or hacked.

The default configurations also normally include support for many different ciphers and modes, so you will often see a combination of strong and weak ciphers supported, or both ESP and AH (which does not encrypt at all) being supported. In these cases, someone with access to the client system (either directly, or over the network) could re-configure the client to use a weak cipher that could be easily cracked (it would not take long to crack a 40-bit export-grade cipher with modern equipment), or worse to use AH which passes the traffic in the clear. The user would almost certainly never notice the change because the VPN works just the same, and who ever bothers to manually check the tunnel mode and encryption ciphers after every connection?

3.8 Poor Guidance and Documentation

Many VPN implementations do not provide sufficient guidance and documentation to allow the end-user to make informed decisions about which configuration to use. Some examples of areas where guidance would be helpful but is very rarely given are:

- Use of weak ciphers such as export-grade or single DES, which can be cracked relatively easily;
- Use of weak authentication mechanisms such as pre-shared key with IKE Aggressive Mode, which transmits the username in the clear, and is vulnerable offline password cracking if a valid username is known; and

- Selection of the AH protocol, which doesn't encrypt the VPN traffic at all.

Typically there are no warnings about these things in the documentation, and usually there is no warning message when these options are selected in the configuration program either.

This means that users do not know what options are safe and what ones are risky. This is not a good state of affairs for such a critical part of the security perimeter. Vendors should not assume that the end-user will understand the details and security characteristics of IPsec and IKE.

4 Conclusions

VPN systems are not the invulnerable systems that they are often believed to be. The vast majority of remote access VPN systems that have been tested by NTA Monitor (about 90%) have had significant security issues, and it has been possible to demonstrate a full compromise on a large proportion of these vulnerable systems.

Some of the common issues that have been found are detailed in the earlier sections, but the root causes of these can be summarised as follows:

- The real-world VPN security issues are rarely in the cryptographic algorithms.
  A lot of attention is focused on the cryptography, but in practice, the security problems are not normally in the cryptographic algorithms that are used; the vulnerabilities are generally caused by poor configuration or bad implementation rather than the cryptography.
  This focus on cryptography is a problem for two reasons:
  - People frequently associate the security of the cryptographic algorithms with the security of the VPN system as a whole and therefore assume that the VPN is unbreakable "because it uses 3DES which would take billions of years to crack"; and
  - People spend too much time worrying about the cryptographic algorithms (e.g. "should we use 3DES or AES?") when they should be worrying about other areas.

- Well accepted security practices are not always used by VPN vendors.
  Things like not leaking information about valid usernames and locking out accounts after a number of failed attempts have been practiced by operating system login authentication for decades, so why do so many VPN implementations not bother?

- Default configurations are often chosen for ease-of-use rather than security. For example many remote access VPNs use IKE Aggressive Mode with pre-shared key authentication as the default configuration, although they may also support the stronger Main Mode with certificates. This is worrying because users will tend to use the default configurations, and they will understandably expect these defaults to be sensible and secure. It could be argued that vendors are failing their customers by choosing insecure default configurations.

- Customers do not always understand the configuration options.
  The configuration options are often difficult to understand by the end-user, and there is often no guidance as to what configurations are potentially insecure. For example, if you choose pre-shared key authentication with IKE Aggressive Mode, most implementations will not warn you of the known problems that are inherent in this method. There is also often little guidance in the documentation. See this bugtraq post for an example of this lack of documentation:
  http://www.securityfocus.com/archive/1/291340/2002-09-01/2002-09-07/2 [7]

- VPN servers have the same software problems as other complex software products.
  Although this paper has not talked in detail about buffer overflows and similar issues, some instances of software bugs which could have security implications have been found during VPN security testing. This should not come as a surprise to anyone, but it is worth pointing out that it should not be assumed that VPN implementations are somehow exempt from the software bugs that plague other complex products.

- VPNs should be tested to ensure that they are secure.
  In the three years that NTA Monitor have been performing VPN testing, about 90% of the sites with remote access IPsec VPNs that were tested had significant vulnerabilities. These were mainly large organisations including financial institutions who had their own in-house security teams. Given this situation, do not blindly trust that your VPN is secure.
  You can use *ike-scan* to help you test the VPN but, like many testing tools, you should be aware that it is quite a complex tool that needs to be fully understood in order for it to be used effectively.

References

[1] R. Hills, "NTA Monitor UDP Backoff Pattern Fingerprinting White Paper", *http://www.nta-monitor.com/ike-scan/whitepaper.pdf*, January 2003.

[2] R. Hills, "Firewall-1 Vendor ID Fingerprinting", *http://www.nta-monitor.com/news/checkpoint2004/index.htm*, May 2004.

[3] R. Morris and K. Thompson, "Password Security: A Case History", Communications of the ACM, Vol.22, No.11, November, 1979, pp.594-597.

[4] H. Krawczyk, M Bellare and R. Canetti, RFC 2104 "HMAC: Keyed-Hashing for Message Authentication", February 1997.

[5] D. Harkins and D. Carrel, RFC 2409 "The Internet Key Exchange (IKE)", November 1998.

[6] R. Pereira and S. Beaulieu, Extended Authentication within ISAKMP/Oakley (XAUTH), December 1999.

[7] R. Hills, "SecuRemote usernames can be guessed or sniffed using IKE exchange", Bugtraq Mailing List, September 2002.