GIAC Advanced Incident Handling and Hacker Exploits
Practical Assignment for SANS Parliament Hill
August 21-24, 2000
Version 1.3

Option 2: Document an exploit, vulnerability or malicious program

Cisco IOS type 7 password vulnerability

Submitted for: SANS Institute

Submitted by: Lee Massey

Submitted on: September 24, 2000

Exploit Details

Name:	Cisco IOS type 7 password vulnerability
Variants:	Riku Meskanen's perl version "ios7decrypt.pl", SPHiXe's 'C' version "ciscocrack.c", BigDog's Psion 3/5 OPL version "cisco.opl", Boson's Windows "GetPass! v1.1"
Operating System:	All Cisco router IOS software
Protocols/Services:	Not applicable
Brief Description:	An exploitation of weak encryption that allows programming code to take an encrypted Cisco IOS type 7 password and compute the plaintext password.

Protocol Description

This vulnerability does not use a particular protocol or service. This vulnerability is not protocol related, but rather an exploitation of weak encryption.

What is Cisco IOS?

Cisco IOS stands for Cisco Internetworking Operating System. IOS can be thought of as Cisco's router operating system. Every Cisco router has a configuration file that instructs the router how it should interact with networks that are directly connected. This interaction will typically include the routing of packets and the exchange of routing information with other layer 3 devices. A sample copy of a simple configuration file from a Cisco 3640 router running Cisco IOS version 12.1 is shown below as Appendix A.

What are the different types of Cisco IOS passwords?

There are three different types of Cisco IOS passwords.

Type 1) Cisco IOS type 0 passwords

There is a command in Cisco IOS that can be issued to encrypt all passwords in the configuration file. If this command is not entered into the configuration file then all passwords (except for the enable secret password) will appear as plaintext as shown below (and in Appendix A):

```
username admin privilege 15 password 0 cisco
```

From the above lines in the Cisco IOS configuration file we can see that in this example the user **admin** has a password of **cisco**. The above passwords are noted as type 0 (zero) as shown by the zero that precedes the actual password. Type 0 passwords use no encryption.

Type 2) Cisco IOS type 7 passwords

The command that is issued to encrypt user passwords is "service password-encryption" and this command should be entered from the Cisco router configuration mode prompt. If the "service password-encryption" command is issued then all type 0 (zero) passwords are become encrypted as show below (and in Appendix B):

```
username admin privilege 15 password 7 0822455D0A16
```

We can now observe that the above passwords are encrypted and that the password type has changed. These encrypted passwords are noted as type 7 passwords.

Type 3) Cisco IOS type 5 passwords

The other type of Cisco password is type 5. This password type is encrypted using an MD5 hashing algorithm and is used by the Cisco IOS to encrypt the **enable secret** password as shown below (and in the attached configuration files):

```
enable secret 5 $1$2ZTf$9UBtjkoYo6vW9FwXpnbuA.
```

The type 5 password encryption uses a stronger method of encryption then that of the type 7 passwords.

Description of variants

There are several variants/code that take advantage of the Cisco IOS type 7 password vulnerability. All the variants listed above crack Cisco IOS type 7 passwords, however the main difference in the variants is the programming language in which they are coded. Judging from the number of available variants, it would lead us to believe that the encryption scheme used in Cisco IOS type 7 passwords is not very strong.

How the exploit works

This exploit works in a similar manner in which L0phtCrack decrypts Windows NT passwords. Rather than trying to obtain a copy of a Windows NT SAM file, an attacker tries to obtain a copy of the encrypted type 7 password from a Cisco router usually by obtaining the Cisco IOS configuration file. This section will not document how an attacker collects such information, only how this information would be used once an attacker gathers it.

In order to understand how a Cisco IOS type 7 password is cracked we will walk through manually cracking a password.

Here is how to break the encryption used for Cisco IOS type 7 passwords.

Assumptions: 1. The encrypted text is already obtained.

It is assumed that the attacker has already obtained the encrypted text and is ready to decrypt the password.

2. The constant value is known.

A constant value exists which provides a salt in an attempt to introduce randomness so that two identical passwords when encrypted will have different ciphertext if the salts are different (see Appendix C). For Cisco IOS type 7 passwords the constant is "tfd;kfoA,.iyewrkldJKD".[5] From what I understand, this constant was obtained by comparing a large number of Cisco IOS type 7 passwords to see if a pattern existed.

We will use the example of the user **admin** as taken from Appendix B. As we can see below the plaintext password that was previously **cisco** has been encrypted into a Cisco IOS type 7 password.

username admin privilege 15 password 7 **0822455D0A16**

Given the assumptions above, here is how to manually exploit the weakness of the poor encryption that is implemented in the Cisco IOS type 7 password.

Let xorstring[n] be the value of the nth character in the constant value that is stated in Assumption 2. For example xorstring[5] = k and xorstring[11] = i.

The encrypted string must be an even length of digits and the entire length of the plaintext password is equal to [(length of encrypted password) - 2] / 2. Thus in our example we can conclude the length of the plaintext password is [12-2] / 2 = 5.

***Note that when decrypting Cisco IOS type 7 passwords manually it is a good idea to have an ASCII chart available.

Step 1. Take the first two digits of the encrypted text.

In our example, the first two digits of the encrypted text is "08". This value is used as decimal representation of an index of where to start taking salts from the constant value.

Step 2. Obtain the current salt.

"A" is the eighth value in the constant value (tfd;kfo**A**,.iyewrkldJKD) as dictated by the first two digits of the encrypted text. Therefore our salt is xorstring[08] = "A".

Step 3. Take the next two digits of the encrypted text.

In our example, the next two digits of the encrypted text is "22". This value is the hexadecimal representation of the first character in the plaintext password XOR'd against the salt (in this case "A"). For a description of the operator XOR see Appendix D.

Step 4. Calculate the first plaintext character in the password.

If we take the hexadecimal representation of the first character in the plaintext password (as obtained in Step 3) we see that it is 0x22 which is the decimal equivalent of 34 ($2 * 16^1 + 2 * 16^0 = 34$). We also know that our salt in this case is "A" which is the decimal equivalent of 65. Now we perform the following operation to obtain the first character of the plaintext password:

0x22 XOR xorstring[08] = first character in plaintext password

Simplify using decimal values…

34 XOR 65 = first character in plaintext password

In order to easily compute the value of 34 XOR 65 we can perform the same operations as shown in Appendix D.

Decimal	Hex	Binary							
34	0x22	0	0	1	0	0	0	1	0
65	0x41	0	1	0	0	0	0	0	1
99	0x63	0	1	1	0	0	0	1	1

As we can conclude from above, 34 XOR 65 = 99 and the ASCII value of 99 is "c". Thus the first plaintext character in the Cisco IOS type 7 password is "c".

Step 5. Obtain the next salt.

Now we must increment the index value (originally 08) by 1. Thus we will use "," which is the ninth value in the constant value (tfd;kfoA**,**.iyewrkldJKD). Therefore our new salt is xorstring[09] = ",".

Step 6. Take the next two digits of the encrypted text.

The next two digits of the encrypted text is "45". This value is the hexadecimal representation of the second character in the plaintext password XOR'd against the new salt (in this case ",").

Step 7. Calculate the next plaintext character in the password.

If we take the hexadecimal representation of the second character in the plaintext password (as obtained in Step 6) we see that it is 0x45 which is the decimal equivalent of 69 ($4 * 16^1 + 5 * 16^0 = 69$). We also know that our salt in this case is "," which is the decimal equivalent of 44. Now we perform the following operation to obtain the first character of the plaintext password:

0x45 XOR xorstring[09] = second character in plaintext password

Simplify using decimal values…

69 XOR 44 = second character in plaintext password

Once again we can perform the same operations as shown in Appendix D to determine the value of 69 XOR 44.

Decimal	**Hex**	**Binary**							
69	0x45	0	1	0	0	0	1	0	1
44	0x2c	0	0	1	0	1	1	0	0
105	0x69	0	1	1	0	1	0	0	1

As we can conclude from above, 69 XOR 44 = 105 and the ASCII value of 105 is "i". Thus the second plaintext character in the Cisco IOS type 7 password is "i".

If we continue following steps 5,6,7 until the encrypted text is exhausted we will obtain the plaintext password. For the sake of brevity the remainder of the plaintext password will be quickly computed in Step 8.

Step 8. Compute the remainder of the plaintext password.

0x5D XOR xorstring[10] = next character in plaintext password

Simplify using decimal values…

93 XOR 46 = next character in plaintext password

Decimal	**Hex**	**Binary**							
93	0x5D	0	1	0	1	1	1	0	1
46	0x2E	0	0	1	0	1	1	1	0
115	0x73	0	1	1	1	0	0	1	1

As we can conclude from above, 93 XOR 46 = 115 and the ASCII value of 115 is "s". Thus the next plaintext character in the Cisco IOS type 7 password is "s".

0x0A XOR xorstring[11] = next character in plaintext password

Simplify using decimal values…

10 XOR 105 = next character in plaintext password

Decimal	Hex	Binary							
10	0x0A	0	0	0	0	1	0	1	0
105	0x69	0	1	1	0	1	0	0	1
99	0x63	0	1	1	0	0	0	1	1

As we can conclude from above, 10 XOR 105 = 99 and the ASCII value of 99 is "c". Thus the next plaintext character in the Cisco IOS type 7 password is "c". At this point we only have 1 plaintext character to decrypt.

0x16 XOR xorstring[12] = next character in plaintext password

Simplify using decimal values…

22 XOR 121 = next character in plaintext password

Decimal	Hex	Binary							
22	0x16	0	0	0	1	0	1	1	0
121	0x79	0	1	1	1	1	0	0	1
111	0x6F	0	1	1	0	1	1	1	1

As we can conclude from above, 22 XOR 121 = 111 and the ASCII value of 111 is "o". As expected the last plaintext character in the Cisco IOS type 7 password is "o". This gives us the expected plaintext password of **cisco** for the user **admin**.

Hence we can see how easy it is to exploit the poor encryption algorithm of Cisco IOS type 7 passwords.

Obviously manually decrypting Cisco IOS type 7 passwords is not a desirable scenario especially when computers are much better designed for brain-numbing calculations than humans. In this case it would be much better to write a script in 'c' or perl to do these calculations as you will see in the next section.

How to use ios7decrypt.pl and GetPass! v1.1

There are several programs that are available that will exploit this vulnerability however this paper will only show two of the several programs: ios7decrypt.pl and GetPass! v1.1.

ios7decrypt.pl

This program is a small perl script (see Appendix E) that takes input in the form of:

username admin privilege 15 password 7 0822455D0A16

and gives output in the form of:

username admin privilege 15 password 7 cisco

Here is an example of how this program appears when run from the prompt:

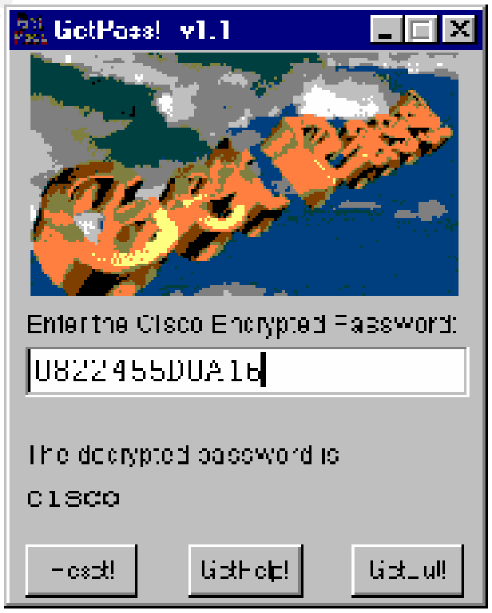
```
# perl ios7decrypt.pl
username admin privilege 15 password 7 0822455D0A16
username admin privilege 15 password cisco
#
```

In the above display of how ios7decrypt.pl functions, the text highlighted in *italics* is what was manually fed to the perl script and the text highlighted in **bold** was the output of the script.

As we can see, ios7decrypt.pl does an excellent job at decrypting Cisco IOS type 7 passwords which would otherwise be a manual painstaking task (see above ☺).

What's that you say? You don't have access to a machine running perl? Well you are in luck because there is a program called GetPass! v1.1 which runs on Windows 9X/NT. It doesn't get much easier then this…

Simply copy the Cisco IOS type 7 encrypted password and paste it into the box as shown in the screen shot below…


GetPass! v1.1

Enter the Cisco Encrypted Password:

0822455D0A16

The decrypted password is

cisco

…and voila! You now have the plaintext password. Although this program is extremely easy to use, one drawback is that it would be very painful to decrypt a large number of encrypted passwords.

Signature of the attack

If an attacker is using one of these programs to decrypt your passwords then it is already too late. The key is to ensure that the Cisco IOS configuration files are secured in such a manner so that an attacker cannot obtain any encrypted Cisco IOS type 7 passwords. I can think of three main methods that an attacker would try to obtain the Cisco IOS configuration file.

1) Poll Cisco IOS configuration file via SNMP

   In this scenario the attacker could try to download the Cisco IOS configuration file via SNMP. There are several ways to do this ranging from custom written code to specific applications such as Solarwinds' SNMP Brute Force Attack (http://www.solarwinds.net/Tools/Security/SNMP%20Brute%20Force/index.htm). This would allow the attacker to gain the configuration file from the Cisco router and then quickly decrypt any Cisco IOS type 7 password. In this case, a network administrator should be looking for any authorized SNMP polling from either the log that resides locally in the Cisco routers' buffer or from a syslog host.

2) Attack the tftp server

   In this scenario the attacker could try to attack and gain access to a tftp server in order to gain access to several Cisco configuration files. Why attempt to gain access to one Cisco router when you could have access to many! ☺ Given the numerous methods to break into servers, the network administrator should always be looking for any suspicious actions or log entries. To fully document how to secure a server from the various types of attack is beyond the scope of this document.

3) Watch for e-mail sent to the Cisco Technical Assistance Center (TAC)

   In many cases when a network administrator has a network problem that might be related to a Cisco router a case is opened with the Cisco TAC who often ask for a copy of the output from a "show tech-support" command. This command outputs almost everything about the router, including the configuration file. If an attacker was able to break in to the network administrator's Internet SMTP server (i.e. sendmail server) undetected, then the attacker could monitor and capture messages bound for user@cisco.com. If the attacker wasn't particularly patient then the attacker could always create network problems in a hope that this would increase the chances of a network administrator opening a case with the Cisco TAC.

Of course the attacker could use tactics such as social engineering, shoulder surfing or using a sniffer to obtain passwords in order to access the router but that would not be exploiting the poor encryption algorithm implemented in Cisco IOS type 7 passwords.

<u>How to protect against it?</u>

There is no way to protect Cisco IOS type 7 passwords from being easily decrypted due to the nature of the weak reversible algorithm that is implemented.

"Cisco has no immediate plans to support a stronger encryption algorithm for Cisco IOS user passwords. If Cisco should decide to introduce such a feature in the future, that feature will definitely impose an additional ongoing administrative burden on users who choose to take advantage of it".[1]

Cisco does make some good recommendations on how to protect against this type of exploit. In summary… don't use Cisco IOS type 7 passwords. "Cisco recommends that all Cisco IOS devices implement the authentication, authorization, and accounting (AAA) security model. AAA can use local, RADIUS, and TACACS+ databases"[1]. This is a good recommendation because it centralizes user management (easier maintenance) and removes the risks of using Cisco IOS type 7 passwords. This method of protecting against this vulnerability could also be complimented by having the authentication portion of the AAA security model be passed on to a device that supports one-time passwords like Security Dynamics SecurID.

If it is necessary to implement Cisco IOS type 7 passwords on your Cisco devices then here are some suggestions that you can use to protect from the 3 scenarios discussed above:

1) Poll Cisco IOS configuration file via SNMP

To protect against this type of attack the network administrator has a few options.

i) Do not implement SNMP. If your device does not respond to SNMP polling then the attacker cannot download the configuration file. This however is not often a feasible solution since the network administrator often needs SNMP to provide network statistics.

ii) Implement SNMP access lists. If you must use SNMP then you should configure access lists that restrict which hosts can poll for SNMP related data. Example:

```
access-list 1 permit 1.1.1.1
access-list 1 permit 2.2.2.2
snmp-server community private RW 1
```

Using the above configuration in your Cisco IOS configuration file, only hosts 1.1.1.1 and 2.2.2.2 are allowed privileged SNMP access to your device. Of course you would use a much more secure SNMP community string than "private". ☺

2) Attack a tftp server

In order to protect your tftp server from attack you must secure the server itself both physically and logically. In this case the administrator should harden the OS (i.e. Unix, Linux, Windows NT, etc.) and ensure that all necessary OS and application patches are installed. The administrator should also regularly port scan this server to ensure that only the necessary services are running. Scanning the server regularly should also alert the administrator to any possible backdoors if suddenly a high port is open!

3) Watch for e-mail sent to the Cisco Technical Assistance Center (TAC)

In this situation the e-mail server administrator be watching for suspicious activity as well as following the steps outlined in scenario 2 above to ensure that the possibility of having the SMTP server compromised is reduced. Also if the Cisco IOS configuration file needs to be sent to anyone, the network administrator should ensure that the file is properly sanitized (i.e. removal of all password and security related information). Another way to avoid an attacker from obtaining this information in this manner is to use a more secure transport. For example use secure copy rather than e-mail.

Conclusion

If at all possible try to avoid using Cisco IOS type 7 passwords. It seems like Cisco has no plans of making improvements to their encryption algorithm that is used to encrypt type 7 passwords. Implementing an AAA security model is step in the right direction using TACACS+ or RADIUS which will remove the possibility of an attacker decrypting your Cisco IOS type 7 passwords and infiltrating your network.

Source code/ Pseudo code

Links to where the various source code/programs can be found:

SPHiXe's 'C' version:	ciscocrack.c
Riku Meskanen's perl version:	ios7decrypt.pl
BigDog's Psion 3/5 OPL version:	cisco.opl
Major Malfunction's Palm-Pilot 'C' port:	ciscopw_1-0.zip
Boson's Windows GetPass:	GetPass
L0pht's Palm Pilot version:	Cisco Type 7 Password Decryptor

Additional Information

References and Links to additional information:

1) Cisco IOS Password Encryption Facts
   http://www.cisco.com/warp/public/701/64.html
2) Useful Cisco Password Utilities
   http://www.alcrypto.co.uk/cisco/
3) The PC ASCII Chart
   http://a1computers.net/pcascii.htm
4) Mudge's explanation of this vulnerability
   http://www.alcrypto.co.uk/cisco/mudge.txt
5) Cisco.txt - Text file from Mudge's Cisco Type 7 Password Decryptor
   http://www.l0pht.com/~kingpin/cisco.zip
6) The Bitwise XOR Operator
   http://www.webreference.com/js/tips/991017.html

Notes

Cisco IOS type 7 passwords have a maximum length of 25 characters. Recall that the first two digits of the encrypted text represent an index into the constant value (tfd;kfoA,.iyewrkldJKD) which range between 00-15. In the case where the index is 15, then we must move to the 15th character in the constant value, namely "r". When manually decrypting passwords we incremented by one for each character in the plaintext password. For example, to calculate the next character in the plaintext character we would use "k" (xorstring[16]). However we can see that in the case where the plaintext password is long (i.e. longer than 8 characters) we run out of values in the constant string. I did not have a chance to investigate this issue. I am not sure if once you used "D" in the calculations, if you would start again at "t" or if perhaps the information that I obtained regarding the constant value was incomplete. I obtained the constant value string from the Cisco.txt file that is in the following zip file:

http://www.l0pht.com/~kingpin/cisco.zip

No diagrams were included in this document because it was not necessary given the type of vulnerability.

Appendix A - Cisco IOS Configuration file with "no service password-encryption"

**** Notice the command "no service password-encryption" and the clear text passwords.*

```
Current configuration : 656 bytes
!
version 12.1
service timestamps debug uptime
service timestamps log uptime
no service password-encryption
!
hostname Router
!
enable secret 5 $1$2ZTf$9UBtjkoYo6vW9FwXpnbuA.
!
username admin privilege 15 password 0 cisco
!
!
!
!
ip subnet-zero
!
ip audit notify log
ip audit po max-events 100
!
!
!
interface Ethernet0/0
 ip address X.X.X.X X.X.X.X
!
interface Serial0/0
 no ip address
 shutdown
!
interface Serial0/1
 no ip address
 shutdown
!
ip classless
ip route 0.0.0.0 0.0.0.0 X.X.X.X
no ip http server
!
no cdp run
!
line con 0
 transport input none
```

```
line 33 64
line aux 0
line vty 0 4
 login local
!
end
```

Appendix B - Cisco IOS Configuration file with " service password-encryption"

**** Notice the command "service password-encryption" and the encrypted passwords.*

```
Current configuration : 679 bytes
!
version 12.1
service timestamps debug uptime
service timestamps log uptime
service password-encryption
!
hostname Router
!
enable secret 5 $1$2ZTf$9UBtjkoYo6vW9FwXpnbuA.
!
username admin privilege 15 password 7 0822455D0A16
!
!
!
!
ip subnet-zero
!
ip audit notify log
ip audit po max-events 100
!
!
!
interface Ethernet0/0
 ip address X.X.X.X X.X.X.X
!
interface Serial0/0
 no ip address
 shutdown
!
interface Serial0/1
 no ip address
 shutdown
!
ip classless
ip route 0.0.0.0 0.0.0.0 X.X.X.X
no ip http server
!
no cdp run
!
line con 0
 transport input none
```

```
line 33 64
line aux 0
line vty 0 4
 login local
!
end
```

Appendix C - Introducing randomness with the use of salts

We can see below that by using a salt (the first two digits in the encrypted text) that the same password can be encrypted in several different ways. As an example, 3 different users with the same password are shown below but we can see that the ciphertext is different in each case due to the salt.

```
username admin1 privilege 15 password 7 05080F1C2243
username admin2 privilege 15 password 7 0822455D0A16
username admin3 privilege 15 password 7 094F471A1A0A
```

In each of the above cases the password is **cisco**. As an exercise, try running the above encrypted text through GetPass! v1.1 to verify the results.

Appendix D - Explanation of XOR

The operator XOR (exclusive OR) compares two bits and assigns 1 as the result when the two operands are different. Here is the truth table for the operator XOR:

Bit1	**Bit2**	**Bit1 ^ Bit2**
0	0	0
0	1	1
1	0	1
1	1	0

Example:

Let A = 35 and Let B = 94.

Name	**Decimal**	**Hex**	**Binary**							
A	35	0x23	0	0	1	0	0	0	1	1
B	94	0x5E	0	1	0	1	1	1	1	0
A ^ B	125	0x7D	0	1	1	1	1	1	0	1

Hence we can see that when we take A XOR B we receive a decimal value equal to 125.

Appendix E - iosdecrypt.pl

```
#!/usr/bin/perl -w
# $Id: cisco.passwords.html,v 1.2 2000/08/18 00:43:54 fyodor Exp $
#
# Credits for orginal code and description hobbit@avian.org,
# SPHiXe, .mudge et al. and for John Bashinski <jbash@CISCO.COM>
# for Cisco IOS password encryption facts.
#
# Use for any malice or illegal purposes strictly prohibited!
#

@xlat = ( 0x64, 0x73, 0x66, 0x64, 0x3b, 0x6b, 0x66, 0x6f, 0x41,
          0x2c, 0x2e, 0x69, 0x79, 0x65, 0x77, 0x72, 0x6b, 0x6c,
          0x64, 0x4a, 0x4b, 0x44, 0x48, 0x53 , 0x55, 0x42 );

while (<>) {
        if (/(password|md5)\s+7\s+([\da-f]+)/io) {
            if (!(length($2) & 1)) {
                $ep = $2; $dp = "";
                ($s, $e) = ($2 =~ /^(..)(.+)/o);
                for ($i = 0; $i < length($e); $i+=2) {
                    $dp .= sprintf "%c",hex(substr($e,$i,2))^$xlat[$s++];
                }
                s/7\s+$ep/$dp/;
            }
        }
        print;
}
# eof
```