Lab 2: Diffie-Hellman, Public Key and Private Key

You will be allocated an instance of the Cloud. It is recommended that you use Linux Kali for the following, but you can also use the Windows version of Openssl (http://asecuritysite.com/openssl.zip). Demo: https://youtu.be/3n2TMpHqE18

1 Diffie-Hellman

No	Description	Result

1	Bob and Alice have agreed on the values:
	
	G=2879, N= 9929
	Bob Select x=6, Alice selects y=9
		Now calculate (using the Windows calculator):
		
		Bob's A value (G^x mod N):
		
		Alice's B value (G^Y mod N):

2	Now they exchange the values. Next calculate the shared key:	Bob's value (B^x mod N):
		
		Alice's value (A^Y mod N):
		
		Do they match? [Yes] [No]

3	If you are in the lab, select someone to share a value with. Next agree on two numbers (G and N). You should generate a random number, and so should they. Do not tell them what your random number is. Next calculate your A value, and get them to do the same. Next exchange values.	Numbers for G and N: Your x value: Your A value: The B value you received:

		Shared key: Do they match: [Yes] [No]

B Private Key

Download openssl from the following link and open-up a console window.

http://asecuritysite.com/openssl.zip

No	Description	Result

1	Use:
	
	`openssl list-cipher-commands`
	
	`openssl version`
		Outline five encryption methods that are supported:
		
		Outline the version of OpenSSL:

2	Using openssl and the command in the form: `openssl prime –hex 1111`	Check if the following are prime numbers: 42 [Yes][No] 1421 [Yes][No]
2	Now create a file named myfile.txt (either use Notepad or another editor). Next encrypt with aes-256-cbc `openssl enc -aes-256-cbc -in myfile.txt -out encrypted.bin` and enter your password.	Use the following command to view the output file: `cat encrypted.bin` Is it easy to write out or transmit the output: [Yes][No]

3	Now repeat the previous command and add the –base64 option. `openssl enc -aes-256-cbc -in myfile.txt -out encrypted.bin –base64`	Use following command to view the output file: `cat encrypted.bin` Is it easy to write out or transmit the output: [Yes][No]
4	Now Repeat the previous command and observe the encrypted output. `openssl enc -aes-256-cbc -in myfile.txt -out encrypted.bin –base64`	Has the output changed? [Yes][No] Why has it changed?
5	Now let’s decrypt the encrypted file with the correct format: `openssl enc -d -aes-256-cbc -in encrypted.bin -pass pass:napier -base64`	Has the output been decrypted correctly? What happens when you use the wrong password?
6	If you are working in the lab, now give you private key to your neighbour, and get them to encrypt a secret message for you.	Did you manage to decrypt their message? [Yes][No]
7	Now encrypt a file with Blowfish and see if you can decrypt it.	Did you manage to decrypt the file? [Yes][No]
8	Now encrypt a file with 3DES and see if you can decrypt it.	Did you manage to decrypt the file? [Yes][No]
9	Now encrypt a file with RC2 and see if you can decrypt it.	Did you manage to decrypt the file? [Yes][No]

C Public Key

We will using Openssl to perform the following (try and do it from a console window):

No	Description	Result
1	First we need to generate a key pair with: `openssl genrsa -out private.pem 1024` This file contains both the public and the private key.	What is the type of public key method used: How long is the default key: How long did it take to generate a 1,024 bit key? Use the following command to view the keys: `type private.pem` (or `cat private.pem` in Linux)
2	Use following command to view the output file: `type private.pem`	What can be observed at the start and end of the file:
3	Next we view the RSA key pair: `openssl rsa -in private.pem -text`	Which are the attributes of the key shown: Which number format is used to display the information on the attributes:

4	Let’s now secure the encrypted key with 3-DES: `openssl rsa -in private.pem -des3 -out key3des.pem`	
4	Next we will export the public key: `openssl rsa -in private.pem -out public.pem -outform PEM -pubout`	View the output key. What does the header and footer of the file identify?
5	Now we will encrypt with our public key: `openssl rsautl -encrypt -inkey public.pem -pubin -in myfile.txt -out file.bin`	
6	And then decrypt with our private key: `openssl rsautl -decrypt -inkey private.pem -in file.bin -out decrypted.txt`	What are the contents of decrypted.txt
7	If you are working in the lab, now give you public key to your neighbour, and get them to encrypt a secret message for you.	Did you manage to decrypt their message? [Yes][No]

D Storing keys

We have stored our keys on a key ring file (PEM). Normally we would use a digital certificate to distribute our public key. In this part of the tutorial we will create a crt digital certificate file.

No	Description	Result

1	If you are using Windows (if you are using Linux, go the following step):
	• First download a conf file which will define the defaults for our digital certificate: http://asecuritysite.com/openssl.txt
	• Store this in your openssl folder (rename it back to openssl.conf).
	
	Next create the crt file with the following:
	`openssl req -new -key private.pem -out cert.csr -config openssl.conf`
	`openssl x509 -req -in cert.csr -signkey private.pem -out server.crt`
	
	Next create the crt file with the following:
	`openssl req -new -key private.pem -out cert.csr -config /etc/ssl/openssl.cnf`
	`openssl x509 -req -in cert.csr -signkey private.pem -out server.crt`
		What is the type of public key method used:
		
		View the certificate file and determine:
		
		The size of the public key:
		
		The encryption method:

E Test

Now take the test at:

http://asecuritysite.com/tests/tests?sortBy=crypto02