

RE: PGP scripting...

Source: <http://www.derkeiler.com/Mailing-Lists/securityfocus/secprog/2003-01/0080.html>

From: Kenneth Buchanan (K.Buchanan@Kastenchase.com)

Date: 01/09/03

From: Kenneth Buchanan <K.Buchanan@Kastenchase.com>
To: 'Tom Arseneault' <TArseneault@counterpane.com>, "'jasonc@science.org'" <jasonc@science.org>,
Date: Thu, 9 Jan 2003 09:01:15 -0500

To be fair, it does depend on the cryptosystem you're using. Jason mentioned he wasn't clear on RSA, which indeed has a 'symmetry' between the keys that allows you to arbitrarily choose which is private and which is public.

But his original post was correct if you are speaking of Discrete Log-based cryptosystems, as opposed to Factoring-based cryptosystems. ElGamal crypto is based on DLP. So is Elliptic Curve Cryptography, which is a variant of ElGamal. In these systems divulging your private key compromises the public key as well.

Kenneth Buchanan
Software Developer
Kasten Chase
k.buchanan@kastenchase.com

"You do not really understand anything unless you can explain it to your grandmother."
— Albert Einstein

-----Original Message-----

From: Tom Arseneault [<mailto:TArseneault@counterpane.com>]
Sent: Wednesday, January 08, 2003 7:28 PM
To: 'jasonc@science.org'; Chris Matthews; 'Frank Knobbe'
Cc: secprog@securityfocus.com
Subject: RE: PGP scripting...

Not true, there is no relation between the keys in that way, you can't find one key from the other in any order. The only difference between the keys is that you keep the private key secret. Either key can be used to encrypt/decrypt messages. Here is an Algorithm for finding the public and private keys:

Algorithm:
Select two prime numbers p and q.

RE: PGP scripting...

Let $n=p.q$
Let $z=(p-1).(q-1)$
Choose a number d that does not divide z .
Choose a number e such that is a multiple of z plus 1.

e and n are published as the public key while d is kept secret as the private key.

Example:
 $p=3, q=11$
 $\rightarrow n=33, z=20$
Choose $d=7$
Choose $e=3$, i.e., $z+1$

As you can see d and e have no relation to each other. If your private key is compromised but somehow they do not have e , since d has no relation to z (hence n) you can not determine e from d . Also although e has a relation to z (hence n) there is still no relation to either d so your still safe.

Here is a quick over view of the public key encryption routines (the clearest I've yet seen) that explain the use of " n " in the above setup: Instead of sending plain text information P , transmitters compute the remainder C when Pe is divided by n . The receiver recovers the unencrypted message P by computing the remainder of Cd divided by n . (" P " stands for plain text, Pe is P modified by " e " (how exactly modified I don't recall) and Cd is C modified by " d ", Pd and Ce should also be valid combinations)

(The algorithm and example are taken off the web page
["http://thalia.spec.gmu.edu/~pparis/classes/notes_101/node63.html"](http://thalia.spec.gmu.edu/~pparis/classes/notes_101/node63.html))

However since you normaly will freely publish your public key then it can be assumed that once someone gets ahold of your private key he/she will now have both your keys, just not for the reason you describe.

As for the usage of the key in encryption and decryption, public key encryption is very compute intensive so while you could do bulk encryption with it whould be very slow.. The usual way things are done is that a symmetrical encryption will be used to encrypt a document (DES, 3DES, BLOWFISH, etc..., very fast) with a randomly generated key and that key is then encrypted with the public key of the person you sending the document to. Since only he, through the use of his private key, can decrypt the symmetrical key only he can decrypt the document.

For a signature, you first take a hash of the document (MD5, SHA1, etc...) and then you encrypt it with your private key so that anyone with your public key can decrypt the signature and verify the document (since only you, thru the use of your private key, could have created the signature they can be assured that the document has not changed in transit and you were the one to send it)

SecurityFocus SecProg: RE: PGP scripting...

Tom Arseneault
Security Engineer
Counterpane Internet Security.
"All humans are born Right-Handed...but the great ones overcome it."

-----Original Message-----

From: Jason Coombs [mailto:jasonc@science.org]
Sent: Wednesday, January 08, 2003 11:26 AM
To: Chris Matthews; 'Frank Knobbe'
Cc: secprog@securityfocus.com
Subject: RE: PGP scripting...

Aloha,

The public key is derived from the private key. Anyone in possession of the private key is by definition also in possession of the public key. The same is not true in reverse, a party can possess the public key without the ability to (reasonably) discover the matching private key.

The public key is normally used for encryption and the private key for decryption.

The private key is used only for producing digital signatures, and I'm not certain that the private key can even be used for bulk encryption, I'm still a little unclear on this point with respect to the RSA algorithm.

Sincerely,

Jason Coombs
jasonc@science.org

-----Original Message-----

From: Chris Matthews [mailto:chris@masc.ca]
Sent: Wednesday, January 08, 2003 4:14 AM
To: 'Frank Knobbe'
Cc: secprog@securityfocus.com
Subject: RE: PGP scripting...

-----Original Message-----

From: Frank Knobbe [mailto:fknobbe@knobbeits.com]

....

>So once the data has been encrypted on that box, the statement "If the
>system is compromised, they have all the data they
>need to get all the data." is not true since all they can get is the
encrypted data.

....

>Regards,
>Frank

<snip>

RE: PGP scripting...

SecurityFocus SecProg: RE: PGP scripting...

I believe the original question involved more of a dynamic modification of data on the machine's harddrive. If this is the case, and automatic encryption/decryption would require the public/private keys.

Another thought just occurred to me for Andrew:

Which key is being used to encrypt the data? If the public key is being used (and bear with me; my pgp theory is foggy this morning :), then technically anyone that has that public key can corrupt your encrypted data. If the private key was used, then anyone with the public key can easily decrypt it. This means that both keys need to be kept "secret", or am I mistaken on this?

Perhaps you should propose to your client a reevaluation of what exactly you're trying to protect and then try to find an encryption solution that more closely matches your requirements.

Cheers,
Chris