

Forum:[Hack Forums](#)

Thread: [PGP: Pretty Good Privacy | Illustrated & Explained For Newbs \[TUTORIAL \]](#)

Canonsx130 - January 7, 2017, 11:37 am

I felt this section could use some love with tutorials since I don't see many around, hence I decided to make this thread. This is my attempt at explaining PGP to those who are not familiar with it. PGP is a technology that is widely used these days yet most people I've encountered have never ever heard of it, or at least don't know what exactly it is even though they might have heard of it. So this will be a very introductory guide to PGP and how you can get started. I'll start by explaining some very basics of cryptography followed by the actual guide.

What Is Cryptography?

Let's get right into it with a classic example. Alice and Bob want to exchange messages between them, but they do not want Evil Tom to intercept or read their messages since he is still a minor. If they send the message directly as plain text, they know Tom will read it since he is skilled & perverted enough to intercept it. So they encrypt the message using a key pair. Alice & Bob both have a public key and a private key. Public key of both Alice & Bob is public, as the name suggests. However, the private key is private and known only by the owner of the private key.

Now when Alice wants to send a message to Bob, she encrypts the message using the public key of Bob. Now when Bob gets the message, he can use his private key to decrypt the message and read it! The beauty of it is, even if Tom intercepts the message, he cannot read it since the message can be decrypted only using the private key of Bob. Similarly when Bob sends a message to Alice which is encrypted using the public key of Alice, only Alice can read it since only she has access to the private key corresponding to the public key with which the message is encrypted. This is a very simple example of an asymmetric encryption algorithm where two separate keys are used to encrypt and decrypt the messages(public & private key). Another category of algorithms is symmetric encryption algorithms where the same key is used for both encryption and decryption. AES is a widely used example for symmetric key encryption and RSA is an example for asymmetric encryption.

Problems with Key-Pair Encryption

There are some problems with conventional asymmetric key encryption techniques which I'll explain by using the Alice & Bob example above. We saw how Alice & Bob can securely exchange messages between them using a public & private key pair. So Tom is no longer able to read the messages even if he intercepts them. Tom had to find a way to read the messages so he makes an account on Hack Forums and learn some Ub3R L33T H4CK1NG TECHNIQUES. With his newly acquired skills, he hacks Alice's machine and changes her public key to his own public key! Similarly he hacks Bob's machine and changes Bob's public key to his own public key! Poor Bob & Alice never knew about this since they had no antivirus installed to detect the anomalies. Now neither Bob nor Alice knew their public key had been replaced with the public key of Evil Tom. Now whenever Bob sends a message to Alice, Tom can decrypt them since they were encrypted using his public key in the first place. He decrypts and reads the message, re-encrypts it using the public key of Alice, and forwards it to Alice. Alice will never know the message was intercepted. This is the popular man-in-the-middle-attack. PGP(Pretty Good Privacy) is a technique used to overcome this problem and in the following sections, we'll see what PGP is and how we can use it.

PGP: Pretty Good Privacy Explained

Using conventional asymmetric encryption techniques, MITM attack was possible and a new method to prevent this was invented where keys were signed to prevent forgery. Signing is used for demonstrating the authenticity of a digital message or public key. Signing is actually the same as encrypting, with the difference that in signing the user encrypts the text with a private key instead of encrypting the text with a public key. The process of encrypting a digital message/key using a private key, with the purpose of demonstrating the authenticity of a digital message or public key, is called signing. Even though signing and encrypting both transform a plain text into a cipher text using a certain key, signing is not called encryption, to denote the difference in use of the public and private keys in the encryption process. The private key used for signing a message differs from the private key used to decrypt a message. Two communicating parties will thus each have a key pair to encrypt/decrypt and a key pair to sign.

How can this new signing technique be used by Alice and Bob? Well both Alice and Bob trusted Omniscient, the administrator of Hack Forums. So Bob and Alice asks Omniscient to sign their public encryption keys using his private signing key. Similarly, they also asks the Staffs to sign their public encryption keys using the staffs' private signing keys. Now when Alice or Bob receives an encrypted message, he/she can simply check whether it is signed by Omniscient or the Staffs. If not, then it is surely not a message sent by Bob/Alice, it was probably sent by Evil Tom.

In the above illustration(took fucking hours to make), you can see Bob and Evil Tom get their public keys signed. Alice only trust signatures from Omniscient and will reject public keys that aren't signed by Omniscient. A public key is usually published by means of a certificate. The certificate includes the name of the public key owner, the public key itself, and a validity date. The validity date indicates the final date until a key can be trusted. After that date, a new certificate should be used. When a user loses a private key for some reason, he can revoke his certificate. This revocation indicates that the public key in the certificate should no longer be trusted. A user can only notice the certificate revocation if he is synchronized with a public key server.

PGP has an internal certificate 'vetting scheme' to assist the users in verifying the integrity of a certificate, a trust model which has been called a web of trust. A given public key may be digitally signed by a third party user to attest to the association between someone and the key. As time goes on, you will accumulate keys from other people that you may want to designate as trusted introducers. Everyone else will each choose their own trusted introducers. And everyone will gradually accumulate and distribute with their key a collection of certifying signatures from other people, with the expectation that anyone receiving it will trust at least one or two of the signatures. This will cause the emergence of a decentralized fault-tolerant web of confidence for all public keys. I'm feeling sleepy now so I'll stop. This is all I feel like writing about PGP. If you are interested to read more, here are some links:

[Pretty Good Privacy: Wikipedia Page](#)

[Why I Wrote PGP: Philip Zimmermann \(Inventor of PGP\)](#)

[How PGP Works: Official Docs](#)

[MIT PGP Key Server](#)

randomUsername25214 - January 7, 2017, 3:41 pm

Thank you for explaining this it helped me alot.

Shanonnn - January 8, 2017, 3:15 am

OP delivers.

Great thread, very helpful for new people. +1

Canonsx130 - January 8, 2017, 4:08 am

Thanks guys. I just realized I didn't include the actual steps to use PGP. I shall edit it in the coming days with the additional information.

snowsoldier - January 10, 2017, 9:21 am

Thank you for explaining this, definitely helpful.

Capwin - January 10, 2017, 4:27 pm

That's some good information you gathered, I use PGP all the time.

BitcoinTalk - January 10, 2017, 4:28 pm

If you guys don't know, you can use KeyBase.io for PGP. It's got a nice design and it's really easy to use, does all the work for you, ect.

2SLiCkZ - January 10, 2017, 4:29 pm

(Jan 10, 2017 05:28 PM)BitcoinTalk Wrote: [\[CITE\]](#) If you guys don't know, you can use KeyBase.io for PGP. It's got a nice design and it's really easy to use, does all the work for you, ect.

Going to look into this, thanks!

kingdras - April 8, 2017, 11:43 pm

Really good stuff here. Thanks a lot!