Pretty Good Privacy

Pretty Good Privacy (**PGP**) is an encryption program that provides cryptographic privacy and authentication for data communication. PGP is used for signing, encrypting, and decrypting texts, e-mails, files, directories, and whole disk partitions and to increase the security of e-mail communications. Phil Zimmermann developed PGP in 1991.[3]

PGP and similar software follow the OpenPGP, an open standard of PGP encryption software, standard (RFC 4880) for encrypting and decrypting data.

Contents

Design

Pretty Good Privacy

Original author(s)	Phil Zimmermann PGP Inc. Network Associates PGP Corp.[1]
Developer(s)	Symantec
Initial release	1991
Stable release	11.2.0 / April 16, 2018[2]
Written in	C
Operating system	Linux, macOS, Windows
Platform	Multi platform
Standard(s)	▪ OpenPGP RFC 4880 OpenPGP Message Format RFC 5581 The Camellia Cipher in OpenPGP RFC 6637 Elliptic Curve Cryptography (ECC) in OpenPGP ▪ PGP/MIME RFC 2015 MIME Security with Pretty Good Privacy (PGP)

PGP encryption uses a serial combination of hashing, data compression, symmetric-key cryptography, and finally public-key cryptography; each step uses one of several supported algorithms. Each public key is bound to a username or an e-mail address. The first version of this system was generally known as a web of trust to contrast with the X.509 system, which uses a hierarchical approach based on certificate authority and which was added to PGP implementations later. Current versions of PGP encryption include options through an automated key management server.

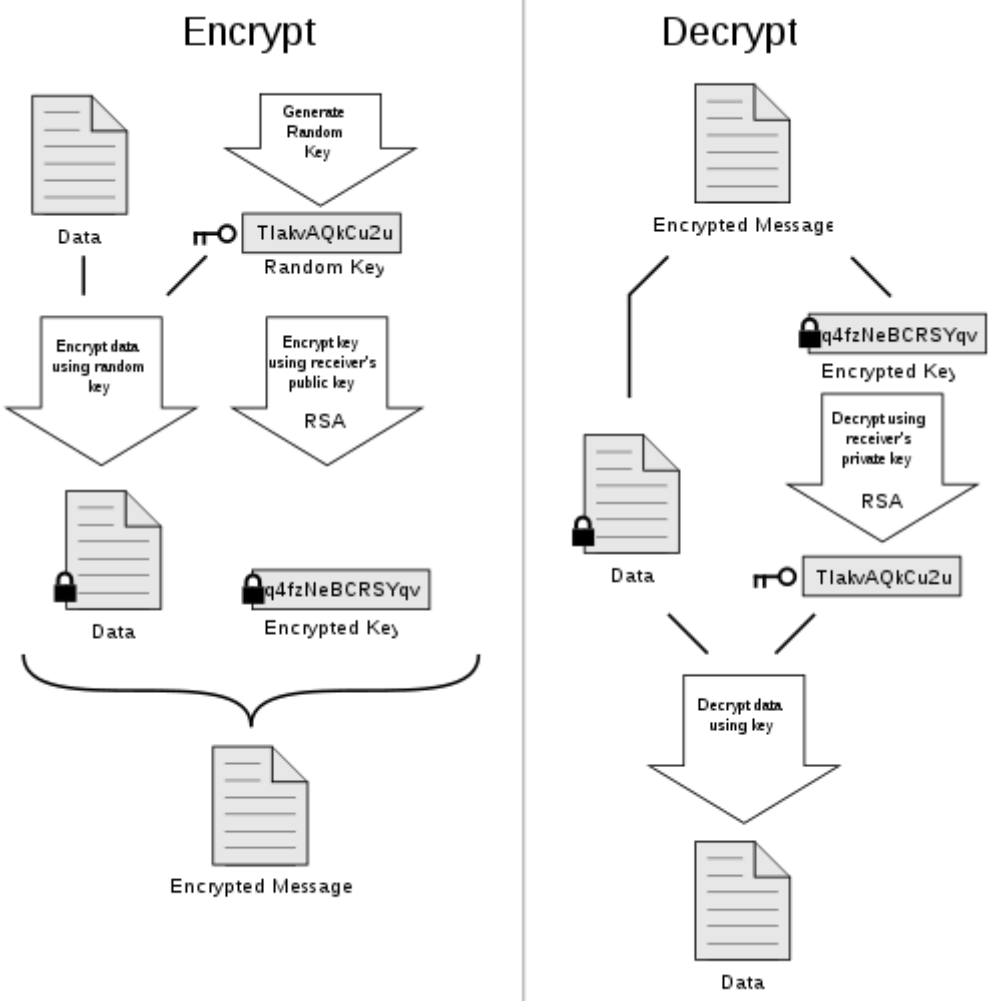
	RFC 3156 MIME Security with OpenPGP
Type	Encryption software
License	Commercial proprietary software

PGP fingerprint

A public key fingerprint is a shorter version of a public key. From a fingerprint, someone can validate the correct corresponding public key. A fingerprint like C3A6 5E46 7B54 77DF 3C4C 9790 4D22 B3CA 5B32 FF66 can be printed on a business card.[4][5]

Compatibility

As PGP evolves, versions that support newer features and algorithms can create encrypted messages that older PGP systems cannot decrypt, even with a valid private key. Therefore, it is essential that partners in PGP communication understand each other's capabilities or at least agree on PGP settings.[6]



How PGP encryption works visually

Confidentiality

PGP can be used to send messages confidentially. For this, PGP uses a hybrid cryptosystem by combining symmetric-key encryption and public-key encryption. The message is encrypted using symmetric encryption algorithm, which requires a symmetric key generated by the sender. The symmetric key is used only once and is also called a session key. The message and its session key are sent to the receiver. The session key must be

sent to the receiver so they know how to decrypt the message, but to protect it during transmission it is encrypted with the receiver's public key. Only the private key belonging to the receiver can decrypt the session key, and use it to symmetrically decrypt the message.

Digital signatures

PGP supports message authentication and integrity checking. The latter is used to detect whether a message has been altered since it was completed (the *message integrity* property) and the former, to determine whether it was actually sent by the person or entity claimed to be the sender (a *digital signature*). Because the content is encrypted, any changes in the message will fail the decryption with the appropriate key. The sender uses PGP to create a digital signature for the message with either the RSA or DSA algorithms. To do so, PGP computes a hash (also called a message digest) from the plaintext and then creates the digital signature from that hash using the sender's private key.

Web of trust

Both when encrypting messages and when verifying signatures, it is critical that the public key used to send messages to someone or some entity actually does 'belong' to the intended recipient. Simply downloading a public key from somewhere is not a reliable assurance of that association; deliberate (or accidental) impersonation is possible. From its first version, PGP has always included provisions for distributing user's public keys in an 'identity certification', which is also constructed cryptographically so that any tampering (or accidental garble) is readily detectable. However, merely making a certificate that is impossible to modify without being detected is insufficient; this can prevent corruption only after the certificate has been created, not before. Users must also ensure by some means that the public key in a certificate actually does belong to the person or entity claiming it. A given public key (or more specifically, information binding a user name to a key) may be digitally signed by a third-party user to attest to the association between someone (actually a user name) and the key. There are several levels of confidence which can be included in such signatures. Although many programs read and write this information, few (if any) include this level of certification when calculating whether to trust a key.

The web of trust protocol was first described by Phil Zimmermann in 1992, in the manual for PGP version 2.0:

> 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.

The web of trust mechanism has advantages over a centrally managed public key infrastructure scheme such as that used by S/MIME but has not been universally used. Users have to be willing to accept certificates and check their validity manually or have to simply accept them. No satisfactory solution has been found for the underlying problem.

Certificates

In the (more recent) OpenPGP specification, *trust signatures* can be used to support creation of certificate authorities. A trust signature indicates both that the key belongs to its claimed owner and that the owner of the key is trustworthy to sign other keys at one level below their own. A level 0 signature is comparable to a web of trust signature since only the validity of the key is certified. A level 1 signature is similar to the trust one has in a certificate authority because a key signed to level 1 is able to issue an unlimited number of level 0 signatures. A level 2 signature is highly analogous to the trust assumption users must rely on whenever they use the default certificate authority list (like those included in web browsers); it allows the owner of the key to make other keys certificate authorities.

PGP versions have always included a way to cancel ('revoke') identity certificates. A lost or compromised private key will require this if communication security is to be retained by that user. This is, more or less, equivalent to the certificate revocation lists of centralised PKI schemes. Recent PGP versions have also supported certificate expiration dates.

The problem of correctly identifying a public key as belonging to a particular user is not unique to PGP. All public key/private key cryptosystems have the same problem, even if in slightly different guises, and no fully satisfactory solution is known. PGP's original scheme at least leaves the decision as to whether or not to use its endorsement/vetting system to the user, while most other PKI schemes do not, requiring instead that every certificate attested to by a central certificate authority be accepted as correct.

Security quality

To the best of publicly available information, there is no known method which will allow a person or group to break PGP encryption by cryptographic, or computational means. Indeed, in 1995, cryptographer Bruce Schneier characterized an early version as being "the closest you're likely to get to military-grade encryption."[7] Early versions of PGP have been found to have theoretical vulnerabilities and so current versions are recommended.[8] In addition to protecting data in transit over a network, PGP encryption can also be used to protect data in long-term data storage such as disk files. These long-term storage options are also known as data at rest, i.e. data stored, not in transit.

The cryptographic security of PGP encryption depends on the assumption that the algorithms used are unbreakable by direct cryptanalysis with current equipment and techniques.

In the original version, the RSA algorithm was used to encrypt session keys. RSA's security depends upon the one-way function nature of mathematical integer factoring.[9] Similarly, the symmetric key algorithm used in PGP version 2 was IDEA, which might at some point in the future be found to have previously undetected cryptanalytic flaws. Specific instances of current PGP or IDEA insecurities (if they exist) are not publicly known. As current versions of PGP have added additional encryption algorithms, their cryptographic vulnerability varies with the algorithm used. However, none of the algorithms in current use are publicly known to have cryptanalytic weaknesses.

New versions of PGP are released periodically and vulnerabilities fixed by developers as they come to light. Any agency wanting to read PGP messages would probably use easier means than standard cryptanalysis, e.g. rubber-hose cryptanalysis or black-bag cryptanalysis (e.g. installing some form of trojan horse or keystroke logging software/hardware on the target computer to capture encrypted keyrings and their passwords). The FBI has already used this attack against PGP[10][11] in its investigations. However, any such vulnerabilities apply not just to PGP but to any conventional encryption software.

In 2003, an incident involving seized Psion PDAs belonging to members of the Red Brigade indicated that neither the Italian police nor the FBI were able to decrypt PGP-encrypted files stored on them.[12]

A second incident in December 2006, (see *In re Boucher*), involving US customs agents who seized a laptop PC that allegedly contained child pornography, indicates that US government agencies find it "nearly impossible" to access PGP-encrypted files. Additionally, a magistrate judge ruling on the case in November 2007 has stated that forcing the suspect to reveal his PGP passphrase would violate his Fifth Amendment rights i.e. a suspect's constitutional right not to incriminate himself.[13][14] The Fifth Amendment issue was opened again as the government appealed the case, after which a federal district judge ordered the defendant to provide the key.[15]

Evidence suggests that as of 2007, British police investigators are unable to break PGP,[16] so instead have resorted to using RIPA legislation to demand the passwords/keys. In November 2009 a British citizen was convicted under RIPA legislation and jailed for nine months for refusing to provide police investigators with encryption keys to PGP-encrypted files.[17]

PGP as a cryptosystem has been criticized for complexity of the standard, implementation and very low usability of the user interface[18] including by recognized figures in cryptography research.[19][20] It uses an ineffective serialization format for storage of both keys and encrypted data, which resulted in signature-spamming attacks on public keys of prominent developers of GNU Privacy Guard. Backwards compatibility of the OpenPGP standard results in usage of relatively weak default choices of cryptographic primitives (CAST5 cipher, CFB mode, S2K password hashing).[21] The standard has been also criticized for leaking metadata, usage of long-term keys and lack of forward secrecy. Popular end-user implementations have suffered from various signature-striping, cipher downgrade and metadata leakage vulnerabilities which have been attributed to the complexity of the standard.[22]

History

Early history

Phil Zimmermann created the first version of PGP encryption in 1991. The name, "Pretty Good Privacy" was inspired by the name of a grocery store, "Ralph's Pretty Good Grocery", featured in radio host Garrison Keillor's fictional town, Lake Wobegon.[23] This first version included a symmetric-key algorithm that Zimmermann had designed himself, named BassOmatic after a *Saturday Night Live* sketch. Zimmermann had been a long-time anti-nuclear activist, and created PGP encryption so that similarly inclined people might securely use BBSs and securely store messages and files. No license fee was required for its non-commercial use, and the complete source code was included with all copies.

In a posting of June 5, 2001, entitled "PGP Marks 10th Anniversary",[24] Zimmermann describes the circumstances surrounding his release of PGP:

> It was on this day in 1991 that I sent the first release of PGP to a couple of my friends for uploading to the Internet. First, I sent it to Allan Hoeltje, who posted it to Peacenet, an ISP that specialized in grassroots political organizations, mainly in the peace movement. Peacenet was accessible to political activists all over the world. Then, I uploaded it to Kelly Goen, who proceeded to upload it to a Usenet newsgroup that specialized in distributing source code. At my request, he marked the Usenet posting as "US only". Kelly also uploaded it to many BBS systems around the country. I don't recall if the postings to the Internet began on June 5th or 6th. It may be surprising to some that back in 1991, I did not yet know enough about Usenet newsgroups to realize that a "US only" tag was merely an advisory tag that had little real effect on how Usenet propagated newsgroup postings. I thought it actually controlled how Usenet routed the posting. But back then, I had no clue how to post anything on a newsgroup, and didn't even have a clear idea what a newsgroup was.

PGP found its way onto the Internet and rapidly acquired a considerable following around the world. Users and supporters included dissidents in totalitarian countries (some affecting letters to Zimmermann have been published, some of which have been included in testimony before the US Congress), civil libertarians in other parts of the world (see Zimmermann's published testimony in various hearings), and the 'free communications' activists who called themselves cypherpunks (who provided both publicity and distribution); decades later, CryptoParty activists did much the same via Twitter.

Criminal investigation

Shortly after its release, PGP encryption found its way outside the United States, and in February 1993 Zimmermann became the formal target of a criminal investigation by the US Government for "munitions export without a license". At the time, cryptosystems using keys larger than 40 bits were considered munitions within the definition of the US export regulations; PGP has never used keys smaller than 128 bits, so it qualified at that time. Penalties for violation, if found guilty, were substantial. After several years, the investigation of Zimmermann was closed without filing criminal charges against him or anyone else.

Zimmermann challenged these regulations in an imaginative way. He published the entire source code of PGP in a hardback book,[25] via MIT Press, which was distributed and sold widely. Anybody wishing to build their own copy of PGP could cut off the covers, separate the pages, and scan them using an OCR program (or conceivably enter it as a type-in program if OCR software was not available), creating a set of source code text files. One could then build the application using the freely available GNU Compiler Collection. PGP would thus be available anywhere in the world. The claimed principle was simple: export of *munitions*—guns, bombs, planes, and software—was (and remains) restricted; but the export of *books* is protected by the First Amendment. The question was never tested in court with respect to PGP. In cases addressing other encryption software, however, two federal appeals courts have established the rule that cryptographic software source code is speech protected by the First Amendment (the Ninth Circuit Court of Appeals in the Bernstein case and the Sixth Circuit Court of Appeals in the Junger case).

US export regulations regarding cryptography remain in force, but were liberalized substantially throughout the late 1990s. Since 2000, compliance with the regulations is also much easier. PGP encryption no longer meets the definition of a non-exportable weapon, and can be exported internationally except to seven specific countries and a list of named groups and individuals[26] (with whom substantially all US trade is prohibited under various US export controls).

PGP 3 and founding of PGP Inc.

During this turmoil, Zimmermann's team worked on a new version of PGP encryption called PGP 3. This new version was to have considerable security improvements, including a new certificate structure which fixed small security flaws in the PGP 2.x certificates as well as permitting a certificate to include separate keys for signing and encryption. Furthermore, the experience with patent and export problems led them to eschew patents entirely. PGP 3 introduced use of the CAST-128 (a.k.a. CAST5) symmetric key algorithm, and the DSA and ElGamal asymmetric key algorithms, all of which were unencumbered by patents.

After the Federal criminal investigation ended in 1996, Zimmermann and his team started a company to produce new versions of PGP encryption. They merged with Viacrypt (to whom Zimmermann had sold commercial rights and who had licensed RSA directly from RSADSI), which then changed its name to PGP Incorporated. The newly combined Viacrypt/PGP team started work on new versions of PGP encryption based on the PGP 3 system. Unlike PGP 2, which was an exclusively command line program, PGP 3 was designed from the start as a software library allowing users to work from a command line or inside a GUI environment. The original agreement between Viacrypt and the Zimmermann team had been that Viacrypt

would have even-numbered versions and Zimmermann odd-numbered versions. Viacrypt, thus, created a new version (based on PGP 2) that they called PGP 4. To remove confusion about how it could be that PGP 3 was the successor to PGP 4, PGP 3 was renamed and released as PGP 5 in May 1997.

Network Associates acquisition

In December 1997, PGP Inc. was acquired by Network Associates, Inc. ("NAI"). Zimmermann and the PGP team became NAI employees. NAI was the first company to have a legal export strategy by publishing source code. Under NAI, the PGP team added disk encryption, desktop firewalls, intrusion detection, and IPsec VPNs to the PGP family. After the export regulation liberalizations of 2000 which no longer required publishing of source, NAI stopped releasing source code.[27]

In early 2001, Zimmermann left NAI. He served as Chief Cryptographer for Hush Communications, who provide an OpenPGP-based e-mail service, Hushmail. He has also worked with Veridis and other companies. In October 2001, NAI announced that its PGP assets were for sale and that it was suspending further development of PGP encryption. The only remaining asset kept was the PGP E-Business Server (the original PGP Commandline version). In February 2002, NAI canceled all support for PGP products, with the exception of the renamed commandline product. NAI (formerly McAfee, then Intel Security, and now McAfee again) continued to sell and support the product under the name McAfee E-Business Server until 2013.[28][29][30]

PGP Corporation and Symantec

In August 2002, several ex-PGP team members formed a new company, PGP Corporation, and bought the PGP assets (except for the command line version) from NAI. The new company was funded by Rob Theis of Doll Capital Management (DCM) and Terry Garnett of Venrock Associates. PGP Corporation supported existing PGP users and honored NAI's support contracts. Zimmermann served as a special advisor and consultant to PGP Corporation while continuing to run his own consulting company. In 2003, PGP Corporation created a new server-based product called PGP Universal. In mid-2004, PGP Corporation shipped its own command line version called PGP Command Line, which integrated with the other PGP Encryption Platform applications. In 2005, PGP Corporation made its first acquisition: the German software company Glück & Kanja Technology AG,[31] which became PGP Deutschland AG.[32] In 2010, PGP Corporation acquired Hamburg-based certificate authority TC TrustCenter and its parent company, ChosenSecurity, to form its PGP TrustCenter[33] division.[34]

After the 2002 purchase of NAI's PGP assets, PGP Corporation offered worldwide PGP technical support from its offices in Draper, Utah; Offenbach, Germany; and Tokyo, Japan.

On April 29, 2010, Symantec Corp. announced that it would acquire PGP for $300 million with the intent of integrating it into its Enterprise Security Group.[35] This acquisition was finalized and announced to the public on June 7, 2010. The source code of PGP Desktop 10 is available for peer review.[36]

Also in 2010, Intel Corporation acquired McAfee. In 2013, the McAfee E-Business Server was transferred to Software Diversified Services, which now sells, supports, and develops it under the name SDS E-Business Server.[28][29]

For the enterprise, Townsend Security currently offers a commercial version of PGP for the IBM i and IBM z mainframe platforms. Townsend Security partnered with Network Associates in 2000 to create a compatible version of PGP for the IBM i platform. Townsend Security again ported PGP in 2008, this time to the IBM z

mainframe. This version of PGP relies on free z/OS encryption facility, which utilizes hardware acceleration. Software Diversified Services also offers a commercial version of PGP (SDS E-Business Server) for the IBM z mainframe.

In May 2018, a bug named EFAIL was discovered in certain implementations of PGP which from 2003 could reveal the plaintext contents of emails encrypted with it.[37][38]

PGP Corporation encryption applications

> *This section describes commercial programs available from PGP Corporation. For information on other programs compatible with the OpenPGP specification, see External links below.*

While originally used primarily for encrypting the contents of e-mail messages and attachments from a desktop client, PGP products have been diversified since 2002 into a set of encryption applications which can be managed by an optional central policy server. PGP encryption applications include e-mails and attachments, digital signatures, laptop full disk encryption, file and folder security, protection for IM sessions, batch file transfer encryption, and protection for files and folders stored on network servers and, more recently, encrypted or signed HTTP request/responses by means of a client-side (Enigform) and a server-side (mod openpgp) module. There is also a Wordpress plugin available, called wp-enigform-authentication, that takes advantage of the session management features of Enigform with mod_openpgp.

The PGP Desktop 9.x family includes PGP Desktop Email, PGP Whole Disk Encryption, and PGP NetShare. Additionally, a number of Desktop bundles are also available. Depending on application, the products feature desktop e-mail, digital signatures, IM security, whole disk encryption, file and folder security, encrypted self-extracting archives, and secure shredding of deleted files. Capabilities are licensed in different ways depending on features required.

The PGP Universal Server 2.x management console handles centralized deployment, security policy, policy enforcement, key management, and reporting. It is used for automated e-mail encryption in the gateway and manages PGP Desktop 9.x clients. In addition to its local keyserver, PGP Universal Server works with the PGP public keyserver—called the PGP Global Directory—to find recipient keys. It has the capability of delivering e-mail securely when no recipient key is found via a secure HTTPS browser session.

With PGP Desktop 9.x managed by PGP Universal Server 2.x, first released in 2005, all PGP encryption applications are based on a new proxy-based architecture. These newer versions of PGP software eliminate the use of e-mail plug-ins and insulate the user from changes to other desktop applications. All desktop and server operations are now based on security policies and operate in an automated fashion. The PGP Universal server automates the creation, management, and expiration of keys, sharing these keys among all PGP encryption applications.

The Symantec PGP platform has now undergone a rename. PGP Desktop is now known as Symantec Encryption Desktop (SED), and the PGP Universal Server is now known as Symantec Encryption Management Server (SEMS). The current shipping versions are Symantec Encryption Desktop 10.3.0 (Windows and macOS platforms) and Symantec Encryption Server 3.3.2.

Also available are PGP Command Line, which enables command line-based encryption and signing of information for storage, transfer, and backup, as well as the PGP Support Package for BlackBerry which enables RIM BlackBerry devices to enjoy sender-to-recipient messaging encryption.

New versions of PGP applications use both OpenPGP and the S/MIME, allowing communications with any user of a NIST specified standard.

OpenPGP

Inside PGP Inc., there was still concern about patent issues. RSADSI was challenging the continuation of the Viacrypt RSA license to the newly merged firm. The company adopted an informal internal standard they called "Unencumbered PGP" which would "use no algorithm with licensing difficulties". Because of PGP encryption's importance worldwide, many wanted to write their own software that would interoperate with PGP 5. Zimmermann became convinced that an open standard for PGP encryption was critical for them and for the cryptographic community as a whole. In July 1997, PGP Inc. proposed to the IETF that there be a standard called OpenPGP. They gave the IETF permission to use the name OpenPGP to describe this new standard as well as any program that supported the standard. The IETF accepted the proposal and started the OpenPGP Working Group.

OpenPGP is on the Internet Standards Track and is under active development. Many e-mail clients provide OpenPGP-compliant email security as described in RFC 3156. The current specification is RFC 4880 (November 2007), the successor to RFC 2440. RFC 4880 specifies a suite of required algorithms consisting of ElGamal encryption, DSA, Triple DES and SHA-1. In addition to these algorithms, the standard recommends RSA as described in PKCS #1 v1.5 for encryption and signing, as well as AES-128, CAST-128 and IDEA. Beyond these, many other algorithms are supported. The standard was extended to support Camellia cipher by RFC 5581 in 2009, and signing and key exchange based on Elliptic Curve Cryptography (ECC) (i.e. ECDSA and ECDH) by RFC 6637 in 2012. Support for ECC encryption was added by the proposed RFC 4880bis (https://tools.ietf.org/html/draft-ietf-openpgp-rfc4880bis) in 2014.

The Free Software Foundation has developed its own OpenPGP-compliant program called GNU Privacy Guard (abbreviated GnuPG or GPG). GnuPG is freely available together with all source code under the GNU General Public License (GPL) and is maintained separately from several graphical user interfaces (GUIs) that interact with the GnuPG library for encryption, decryption and signing functions (see KGPG, Seahorse, MacGPG). Several other vendors have also developed OpenPGP-compliant software.

The development of an open source OpenPGP-compliant library, OpenPGP.js,[39] written in JavaScript, has allowed web-based applications to use PGP encryption in the web browser.

- PGP
  - RFC 1991 PGP Message Exchange Formats (obsolete)[40]
- OpenPGP
  - RFC 2440 OpenPGP Message Format (obsolete)[40]
  - RFC 4880 OpenPGP Message Format
  - RFC 5581 The Camellia Cipher in OpenPGP
  - RFC 6637 Elliptic Curve Cryptography (ECC) in OpenPGP
  - RFC 4880bis (draft) (https://tools.ietf.org/html/draft-ietf-openpgp-rfc4880bis) OpenPGP Message Format
- PGP/MIME
  - RFC 2015 MIME Security with Pretty Good Privacy (PGP)
  - RFC 3156 MIME Security with OpenPGP

OpenPGP's encryption can ensure secure delivery of files and messages, as well as provide verification of who created or sent the message using a process called digital signing. The open source office suite LibreOffice implemented document signing with OpenPGP as of version 5.4.0 on Linux.[41] Using OpenPGP for communication requires participation by both the sender and recipient. OpenPGP can also be used to secure sensitive files when they're stored in vulnerable places like mobile devices or in the cloud.[42]

Limitations

With the advancement of cryptography, parts of PGP have been criticized for being dated:

- The long length of PGP public keys[43]
- Difficulty for the users to comprehend and poor usability[20]
- Lack of ubiquity[20]
- Lack of forward secrecy[43]

In October 2017, the ROCA vulnerability was announced, which affects RSA keys generated by buggy Infineon firmware used on Yubikey 4 tokens, often used with PGP. Many published PGP keys were found to be susceptible.[44] Yubico offers free replacement of affected tokens.[45]

See also

- *Bernstein v. United States*
- Electronic envelope
- Email encryption
- Email privacy
- GNU Privacy Guard
- Key server (cryptographic)
- PGP word list
- PGPDisk
- Pretty Easy privacy
- Privacy
- Public-key cryptography
- S/MIME
- X.509
- ZRTP

References

1. "Where to Get PGP" (https://philzimmermann.com/EN/findpgp/). *philzimmermann.com*. Phil Zimmermann & Associates LLC. February 28, 2006.
2. (in English) « Symantec Endpoint Encryption 11.2 now available » (https://support.symantec.com/en_US/article.ALERT2587.html), sur *Symantec Enterprise Technical Support*, avril 2018 (consulté le 18 septembre 2018).
3. Zimmermann, Philip R. (1999). "Why I Wrote PGP" (https://www.philzimmermann.com/EN/essays/WhyIWrotePGP.html). *Essays on PGP*. Phil Zimmermann & Associates LLC.
4. Furley, Paul M. "PGP public key example" (https://web.archive.org/web/20181221182643/https://www.paulfurley.com/pgp-public-key-example/). There are shorter ways of referring to PGP keys. Archived from the original (https://www.paulfurley.com/pgp-public-key-example/) on December 21, 2018. "can print it on my business card instead of trying to print my whole public key"
5. Marcia Hofmann [@marciahofmann] (January 20, 2015). "my new business card (with image)" (https://twitter.com/marciahofmann/status/557692432494915584) (Tweet). Retrieved July 30, 2020 – via Twitter.

6. "PGP User's Guide, Volume II: Special Topics" (https://web.pa.msu.edu/reference/pgpdoc2.html). *web.pa.msu.edu*. Retrieved November 1, 2020.
7. Schneier, Bruce (October 9, 1995). *Applied Cryptography*. New York: Wiley. p. 587. ISBN 0-471-11709-9.
8. Messmer, Ellen (August 28, 2000). "Security flaw found in Network Associates' PGP" (https://books.google.com/books?id=JxkEAAAAMBAJ&pg=PA81). *Network World*. Vol. 17 no. 35. Southbourough, Massachusetts: IDG. p. 81 – via Google Books.
9. Nichols, Randall (1999). *ICSA Guide to Cryptography*. McGrawHill. p. 267. ISBN 0-07-913759-8.
10. "United States v. Scarfo (Key-Logger Case)" (http://www.epic.org/crypto/scarfo.html). Epic.org. Retrieved February 8, 2010.
11. McCullagh, Declan (July 10, 2007). "Feds use keylogger to thwart PGP, Hushmail | Tech news blog - CNET News.com" (https://web.archive.org/web/20170324015726/https://www.cnet.com/news/feds-use-keylogger-to-thwart-pgp-hushmail/). News.com. Archived from the original (https://www.cnet.com/news/feds-use-keylogger-to-thwart-pgp-hushmail/) on March 24, 2017. Retrieved February 8, 2010.
12. Grigg, Ian (2003). "PGP Encryption Proves Powerful" (http://www.metzdowd.com/pipermail/cryptography/2003-May/004808.html).
13. McCullagh, Declan (December 14, 2007). "Judge: Man can't be forced to divulge encryption passphrase | The Iconoclast - politics, law, and technology - CNET News.com" (http://www.news.com/8301-13578_3-9834495-38.html?tag=nefd.blgs). News.com. Retrieved February 8, 2010.
14. McCullagh, Declan (January 18, 2008). "Feds appeal loss in PGP compelled-passphrase case | The Iconoclast - politics, law, and technology - CNET News.com" (http://www.news.com/8301-13578_3-9854034-38.html). News.com. Retrieved February 8, 2010.
15. McCullagh, Declan (February 26, 2009). "Judge orders defendant to decrypt PGP-protected laptop" (http://news.cnet.com/8301-13578_3-10172866-38.html). CNET news. Retrieved April 22, 2009.
16. John Leyden (November 14, 2007). "Animal rights activist hit with RIPA key decrypt demand" (https://www.theregister.co.uk/2007/11/14/ripa_encryption_key_notice). *The Register*.
17. Chris Williams (November 24, 2009). "UK jails schizophrenic for refusal to decrypt files" (https://www.theregister.co.uk/2009/11/24/ripa_jfl/page2.html). *The Register*. p. 2.
18. Staff, Ars (December 10, 2016). "Op-ed: I'm throwing in the towel on PGP, and I work in security" (https://arstechnica.com/information-technology/2016/12/op-ed-im-giving-up-on-pgp/). *Ars Technica*. Retrieved July 17, 2019.
19. "What's the matter with PGP?" (https://blog.cryptographyengineering.com/2014/08/13/whats-matter-with-pgp/). *A Few Thoughts on Cryptographic Engineering*. August 13, 2014. Retrieved July 17, 2019.
20. Marlinspike, Moxie (February 24, 2015). "GPG And Me" (https://moxie.org/2015/02/24/gpg-and-me.html). Retrieved June 21, 2020.
21. "Latacora - The PGP Problem" (https://latacora.micro.blog/2019/07/16/the-pgp-problem.html). *latacora.micro.blog*. Retrieved July 17, 2019.
22. "Efail: Breaking S/MIME and OpenPGP Email Encryption using Exfiltration Channels" (https://www.usenix.org/system/files/conference/usenixsecurity18/sec18-poddebniak.pdf) (PDF).
23. Holtsnider, Bill; Jaffe, Brian D. (2006). *IT manager's handbook: getting your new job done* (https://books.google.com/books?id=OeQD_QPOYY4C&pg=PA373) (2nd ed.). Morgan Kaufmann. p. 373. ISBN 978-0-08-046574-6.
24. "PGP Marks 10th Anniversary" (http://www.philzimmermann.com/EN/news/PGP_10thAnniversary.html). Phil Zimmermann. Retrieved August 23, 2010.
25. Zimmermann, Philip (1995). *PGP Source Code and Internals*. MIT Press. ISBN 0-262-24039-4.

26. "Lists to Check" (https://web.archive.org/web/20100112230807/https://www.bis.doc.gov//complianceandenforcement/liststocheck.htm). *US Department of Commerce, Bureau of Industry and Security*. Archived from the original (http://www.bis.doc.gov/complianceandenforcement/liststocheck.htm) on January 12, 2010. Retrieved December 4, 2011.
27. "Important Information About PGP & Encryption" (http://www.proliberty.com/references/pgp/). proliberty.com. Retrieved March 24, 2015.
28. "McAfee partners with Software Diversified Services to deliver E-Business Server sales and support." (https://kc.mcafee.com/corporate/index?page=content&id=KB79203) 2014-01-17. Retrieved 2015-06-30.
29. "Long Live E-Business Server for Enterprise-Scale Encryption." (http://www.sdsusa.com/newsdocs/130811.sds.ebs.pdf) Software Diversified Services. 2013-08-11. Retrieved 2015-06-30.
30. "Intel Security is McAfee again." (https://techcrunch.com/2017/04/03/intel-security-is-mcafee-again/) 2017-04-03. Retrieved 2018-01-08.
31. "glueckkanja.com" (http://glueckkanja.com). glueckkanja.com. Retrieved August 6, 2013.
32. "pgp.de" (http://pgp.de). pgp.de. Retrieved August 6, 2013.
33. "pgptrustcenter.com" (https://web.archive.org/web/20140109130044/http://www.pgptrustcenter.com/). pgptrustcenter.com. January 26, 2010. Archived from the original (http://www.pgptrustcenter.com) on January 9, 2014. Retrieved August 6, 2013.
34. "News Room – Symantec Corp" (http://www.pgp.com/insight/newsroom/press_releases/pgp_corporation_acquires_chosensecurity.html). Pgp.com. Retrieved March 23, 2012.
35. "Symantec buys encryption specialist PGP for $300M" (http://www.computerworld.com/s/article/9176121/Symantec_buys_encryption_specialist_PGP_for_300M). Computerworld. April 29, 2010. Retrieved April 29, 2010.
36. "Symantec PGP Desktop Peer Review Source Code" (http://www.symantec.com/connect/downloads/symantec-pgp-desktop-peer-review-source-code). Symantec.com. September 23, 2012. Retrieved August 6, 2013.
37. "Critical PGP and S/MIME bugs can reveal encrypted emails—uninstall now [Updated]" (https://arstechnica.com/information-technology/2018/05/critical-pgp-and-smime-bugs-can-reveal-encrypted-e-mails-uninstall-now/). *arstechnica.com*. May 14, 2018.
38. "EFAIL" (https://efail.de/). *efail.de*. Retrieved May 18, 2018.
39. OpenPGPjs-Team. "OpenPGPjs" (https://openpgpjs.org/).
40. David, Shaw; Lutz, Donnerhacke; Rodney, Thayer; Hal, Finney; Jon, Callas. "OpenPGP Message Format" (https://tools.ietf.org/html/rfc4880). *tools.ietf.org*.
41. "OpenPGP signature support in LibreOffice" (https://blog.thebehrens.net/2017/07/28/openpgp-signature-support-in-libreoffice/). *Thorsten's Weblog*. July 28, 2017. Retrieved December 10, 2017.
42. By Eric Geier, PCWorld. "How to use OpenPGP to encrypt your email messages and files in the cloud (http://www.pcworld.com/article/2472771/how-to-use-openpgp-to-encrypt-your-email-messages-and-files-in-the-cloud.html)." August 22, 2014. September 3, 2014.
43. Green, Matthew (August 13, 2014). "What's the matter with PGP?" (https://blog.cryptographyengineering.com/2014/08/13/whats-matter-with-pgp/). A Few Thoughts on Cryptographic Engineering. Retrieved December 19, 2016.
44. The Return of Coppersmith's Attack: Practical Factorization of Widely Used RSA Moduli (https://crocs.fi.muni.cz/_media/public/papers/nemec_roca_ccs17_preprint.pdf), Matus Nemec, Marek Sys, Petr Svenda, Dusan Klinec, Vashek Matyas, November 2017
45. "Yubico Replacement Program" (https://web.archive.org/web/20181222101837/https://www.yubico.com/keycheck/verify_otp). Archived from the original (https://www.yubico.com/keycheck/verify_otp) on December 22, 2018. Retrieved June 13, 2018.

Further reading

- Garfinkel, Simson (1995). *PGP: Pretty Good Privacy*. O'Reilly & Associates. ISBN 1-56592-098-8.
- Levy, Steven (January 8, 2001). *Crypto: How the Code Rebels Beat the Government—Saving Privacy in the Digital Age*. Penguin Books. ISBN 0140244328.
- Lucas, Michael W. (April 1, 2006). *PGP & GPG Email for the Practical Paranoid*. No Starch Press. ISBN 978-1-59327-071-1.
- Zimmermann, Phil (June 1991). "Why I Wrote PGP" (http://www.philzimmermann.com/EN/essays/WhyIWrotePGP.html). Retrieved March 3, 2008.

External links

- OpenPGP::SDK (https://github.com/public/OpenPGP-SDK)
- MIT Public Key Directory for Registration and Search (https://pgp.mit.edu/)
- List of public keyservers (https://www.rossde.com/PGP/pgp_keyserv.html#pubserv)
- IETF OpenPGP working group (https://datatracker.ietf.org/wg/openpgp/charter/)
- OpenPGP Alliance (https://www.openpgp.org/)