

PGP Identity Management

Secure Authentication and Authorization over the Internet

Vinnie Moscaritolo
PGP Corporation, Palo Alto CA
vinnie@pgp.com

July 12, 2004

Abstract

Access of computer services have conventionally been managed by means of secret passwords and centralized authentication databases, this method dates back to early timeshare systems. Now that applications have shifted to the Internet it has become conspicuously evident that the use of passwords is not scaleable or secure enough for this medium. As an alternative, this paper discusses ways to implement *federated identity management* using strong-cryptography and the same PGP key infrastructure that is widely deployed on the Internet today.

Beyond Passwords

The inherent security weakness and management complexities of password-based authentication and centralized authorization databases often prove inadequate for the real-world needs of todays public networks. However by applying the same proven cryptographic technology used today for securing email we can construct a robust authentication system with the following goals in mind:

- Provide a single sign-on experience in which a user needs to only remember one password, yet at the same time make it less vulnerable to cracking attempts.
- Employ strong user authentication, extendable to multi-factor methods such as tokens or smartcards. The only copy of the authenticating (secret) material is in the possession of the user.
- Such a system should not depend on any trusted servers. Nor should the compromise of any server effect the security of other servers or users.
- We should build on existing and well-accepted infrastructures that scale to fit a very large base of users and servers.
- Ultimately users should be able to securely sign on to the networks of more than one enterprise to conduct transactions.

Authentication with Cryptographic Signatures

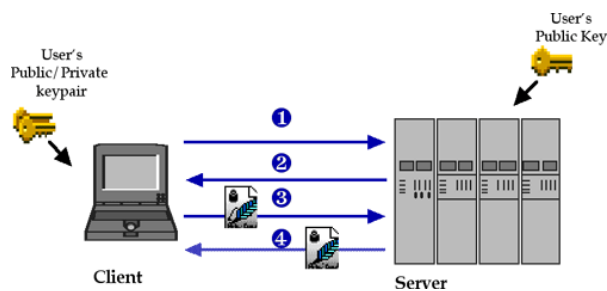
E-mail communication on the Internet faces a security challenge similar to network user authentication. Messages travel through public networks, and can be eavesdropped or counterfeited without large effort. Yet we have been able to successfully mitigate these risks with public key cryptography to digitally sign and authenticate e-mail messages.

With public key cryptography, each user or *principle* generates a pair of mathematically related cryptographic keys. These keys are created in such a way that it is computationally infeasible to derive one key from the other. One of the keys is made publicly available to anyone who wishes to communicate with that user. The other key is kept private and never revealed to anyone else. This private key can be further secured by either placing it in a hardware token or encrypting it to a passphrase, sometimes both. The principal uses the private key to digitally sign data. This digital signature can later be checked with the matching public key to ensure that the data has not been tampered with and that it originated from the holder of the private key.

Since the holder of the private key is the *only* entity that can create a digital signature that verifies with the corresponding public key, there is a strong association between a users identity and the ability to sign with that private key. Thusly a digital signature is strong testimony to the authenticity of the sender.

Cryptographic Challenge-Response

Since the *the public key functions as a principles identity in cyberspace*, we can apply digital signatures to strongly authenticate users of network services. One way to do this is to challenge the user to sign a randomly generated message from the server. The server then verifies the identity of the user with the public key. This process is illustrated below.



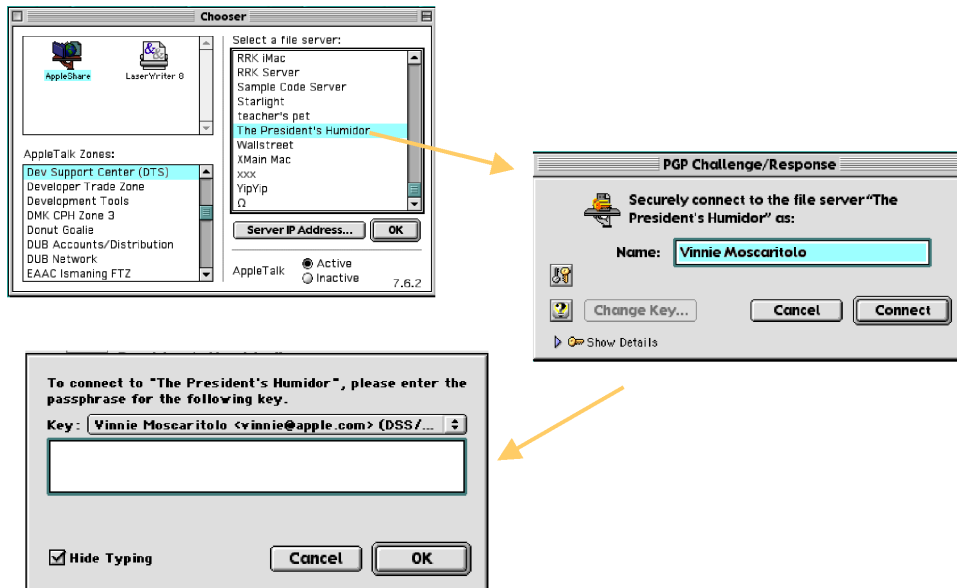
1. The user initiates network service access.
2. The server looks up the users public key in its authentication database. The server then generates a random challenge string and sends the challenge to the client.
3. The client digitally signs the challenge string and returns the cryptographic signature to the server. The client also sends a counter challenge string, which is used to verify the servers authenticity.
4. The server then checks the clients signature, and if successful grants access. It also signs and returns the clients counter-challenge.

The use of such cryptographic user authentication offers a number of advantages over password based systems. For example, if we employ the same key as to sign e-mail, user authentication

becomes as strong as the applied cryptographic digital signature algorithm, thus reducing the need to periodically change the password. Yet the user only would only need to remember one passphrase for all servers using this system. In addition since the user maintains the only secret material in the system, compromising a servers user database results in limited damage. All this is done without the risks associated with passphrase caches or key chains.

PGPuam - a proof of concept

A public key login system was originally prototyped by the author as *PGPuam* and later distributed as sample code by Apple Computer in 1998 [PGPUAM]. Consisting of an *AppleShare-IP* client and server plug-in, it enabled a user to perform two-way strongly authenticated logins to an AppleShare-IP server from a *Mac OS* client. The cryptographic routines were provided by the PGP SDK shipped with PGP 6.0. The user interface was an extension of the existing AppleShare login and is illustrated below:



Although entirely functional, the PGPuam sample was never intended as a shipping product, rather as a practical demonstration of why public key cryptography should be treated as a core operating system component. Unfortunately in the late 90s cryptography was mired in both commercial and political constraints and widespread public key infrastructure was slow to solidify. Nevertheless PGPuam was successful in that it demonstrated that cryptography could be used for more than encrypting email. Note that AppleShare-IP was just a convenient test platform, and this concept is portable to file servers which supports plug-in authentication modules such as Apache modules or Windows GINA Authentication DLL.

Authentication vs. Authorization

Although the PGPuam authentication effectively addresses most password management and single sign-on issues, it does nothing for user authorization. File servers still have to maintain some form of user-file access control database. Managing and maintaining these user authorization databases securely quickly become cumbersome for server administrators when more than a handful of servers are involved.

Consider for example what happens when a new user wishes to gain access to a server. The system administrator must create an account and add their name and access information to the appropriate server database. If the user wishes to access a number of servers, this process must be duplicated and kept synchronized on each server. This is further exacerbated when the servers are owned by different organizations. Conversely when a user departs from an organization each of the servers must then updated to reflect this change. Often removed users are overlooked and left active on servers managed by different departments thus creating a security risk.

Although have been a number of attempts to create automated systems to replicate or centralize the authorization databases, Kerberos for example, yet they all seem to share the following drawbacks:

- The authorization server itself must be physically secure and is a critical link in the security chain.
- Each server must be in communication with the authorization server to verify user identity and authorizations. This could be an unreasonable requirement for remote sites or devices. For example a door badge reader.
- The authorization server is an ideal target for denial of service attacks, since they affect all the servers managed by it.

PGPticket - Secure Federated User Authorization

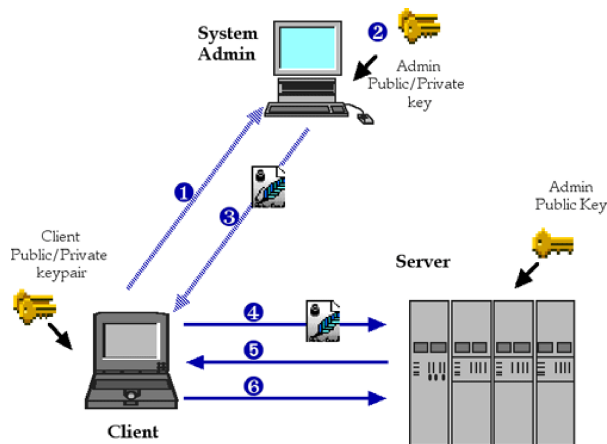
A number of papers have been written describing ingenious alternatives for distributing network service authorization [BFL] [SPKI]. In particular the Simple Public Key Infrastructure (SPKI) model introduces a change in how authorization is performed for network services. Instead of maintaining a per-server database of users names, passwords and their corresponding access rights, we can apply digital-signature technology to create a *authorization certificate* . Think of it as a digital permission slip, valid only for a specific user's key over a certain period of time. This authorization certificate is signed by the organization or a proxy who owns the server and presented by the user upon accessing a restricted service.

One way to encapsulate these certificates is in the form of an OpenPGP standalone signature packet [OPENPGP]. These packets knows as PGPtickets form the basis of a lightweight but very secure federated authorization protocol [PGPTICKET]. Each PGPticket contains the following fields:

- The ISSUER who generates and signs the certificate, represented by a PGP Key-ID.
- The SUBJECT, the principal or set of principals to which the certificate grants its authorization. A combination of KeyID, Algorithm Id and Key fingerprint is used to represent the subject.
- VALIDITY is some combination of dates or on-line tests specifying the validity period/conditions of the certificate. Typically a creation and expiration date. This might be useful for a school that would allow access to facilities for some limited period such as term.
- AUTHORIZATION is a structured field expressing the authorization that this certificate grants to the subject. This data could be represented as SAML or some form of XML.
- DELEGATION is a flag that indicates if the subject is allowed to delegate the specified authorization further.

PGPtickets can be issued in or out of band and are uniquely identified by the hash of the ticket packet, known as it's Ticket-ID. The issuer verifies the subject's key information through standard practice, such as key fingerprint. Unless there is a specific requirement to encrypt the signed tickets they can be returned via clear-text e-mail or even placed on a public website and accessed by the Ticket-ID. The subject can even store the ticket in a database, smart-card or token.

The following illustrates the process of accessing a service with a PGPticket.:



1. The user requests server access from the system admin (could be a verbal request). The user provides either a copy of his public key or makes the key available on a key-server. The issuer verifies the validity of this key.
2. The administrator generates the PGPticket
3. The user retrieves the PGPticket and stores it in a local ticket database.
4. When the user attempts to access a network service, the client searches its ticket database for appropriate ticket, and sends a copy of the it along with the access request.
5. The server checks validity of ticket by verifying the admin signature and expiration date of ticket. The server then generates a random challenge string and sends the challenge to the client requesting that the key specified in the ticket sign it.
6. The client signs and returns the challenge which is checked by the server and if successful access is granted with the authorizations specified in the ticket

The server only requires a copy of the root issuer's public key. It doesn't need to store copies of the subjects public keys since the key-fingerprint is signed into the ticket body. The subject public key can be provided by by request from the client and cache it for later use. The same is true for delegated tickets. There is no specific requirement for a certifying authority although its use is certainly not precluded. This makes it quite usable for small sites as well as enterprises.

One of the more interesting features of this design is that it enables the servers to function without access to a key-server, independent of outside influences and resilient to denial of service attack. This allows the use of PGPtickets for standalone devices where no network connection is practical. Imagine the possibilities. PGPtickets can be transported in a token or Bluetooth device, and not only used for such things as web service or VPN access but also for restricted door access. Extending the the usefulness of the PGP PKI to the physical world.

PGPcoupon - Building on XML Web Services

Some other interesting potentials occur when you consider that PGPticket piggybacks on the flexibility of the PGP key infrastructure. Consider a system that produced tickets automatically through some pay per use service and combine that with anonymous keys. Or a using the key splitting features to create a ticket that needs a certain amount of shares for service access.

Another possibility is to mix the technologies of PGPticket and XML object web services. A client could post a web request for a proposals whereby the vendors could reply with a PGP signed coupon that would be honored by various distributors for a given period of time.

For example, imagine a school wants to order a number of books; various vendors compete and send replies. In the replies is 20% off PGPcoupon that say Amazon or BN.com would honor. The client could present this on purchase and have the transaction processed automatically.

Conclusion

I have outlined a number of alternate uses for PGP technology that goes far past email encryption. Most of these designs have been around for a number of years but were untapped because of political or commercial restraints and at times lack of vision. Times have changed and cryptographic technology can be used to solve a number of real-world identity management problems today.

References

- [OPENPGP] Callas, J., Donnerhackle, L., Finney, H., Thayer, R. *OpenPGP Message Format*. RFC 2440, November 1998, <http://www.ietf.org/rfc/rfc2440.txt>
- [SPKI] Ellison, C., *SPKI Requirements*. RFC 2692, September 1999, <http://www.ietf.org/rfc/rfc2692.txt>
- Ellison, C., Frantz, B., Lampson, B., Rivest, R., *SPKI Certificate Theory*. RFC 2693, September 1999, <http://www.ietf.org/rfc/rfc2693.txt>
- [BFL] Matt Blaze, Joan Feigenbaum and Jack Lacy, *Distributed Trust Management* , Proceedings 1996 IEEE Symposium on Security and Privacy.
- [PGPTICKET] Moscaritolo, V, *PGPticket- A Secure Authorization Protocol* , October 1998, Mac-Crypto Workshop, <http://www.vmeng.com/mc/cfp1998.html>
- Moscaritolo, V. Mione, A, *draft-ietf-pgpticket-moscaritolo-mione-02.txt*
- [PGPUAM] Moscaritolo, V, *PGPuam - Public Key Authentication forAppleShare-IP* , October 1998, Mac-Crypto Workshop, <http://www.vmeng.com/mc/cfp1998.html>