Peer-to-Peer Security and Intel's Peer-to-Peer Trusted Library
Chris McKean
August 20, 2001

SANS Security Essentials
GSEC Practical Assignment
Version 1.2e (amended May 22, 2001)

Introduction

The peer-to-peer computing model is not new, but has recently become a commonplace among the average computer user due to the introduction of applications like Napster and Gnutella. Along with the freedom of connecting peer-to-peer come security risks. In response to some of these security risks, Intel has released a code library that software developers can use to strengthen the security of, and add "trust" to, new peer-to-peer applications.

1. 0 Overview of Peer-to-Peer Computing

P2P computing can be defined as the sharing of computer resources and services by direct exchange. P2P computing provides an alternative to the traditional client/server architecture, while employing the existing infrastructure of networks, servers, and clients. The two models can coexist and complement each other.

In a client/server model, the client makes requests of the server. The server responds to the requests and acts on them. With P2P computing, each participating computer (peer) functions as a client with a layer of server functionality. This allows the peer to act both as a client and as a server. P2P applications can handle such functions as storage, computation, messaging, and file distribution.

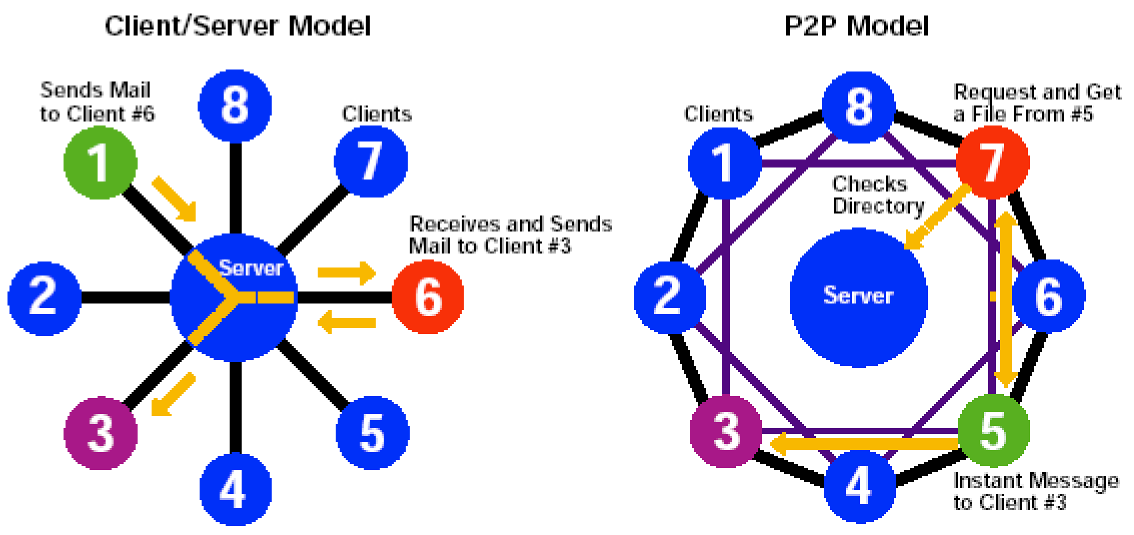

Figure 1. Client Server and P2P Models

Figure 1 (Barkai, p. 4) illustrates the difference between the client/server and the P2P models. It also illustrates how the two models can coexist with each other. In the client/server model every exchange and communication goes through a central server. In the P2P computing model, the peer systems communicate and exchange data directly. Some P2P applications may, at times, also use servers.

Some people distinguish between “pure P2P computing” and “hybrid P2P computing.” “Pure P2P computing” refers to a model where all participating computers are peers and no central server is used to control, coordinate, or manage the exchanges among the peers. In “hybrid P2P computing,” the application relies on a central server to perform some of the functions.

2.0 Peer-to-Peer Security Concerns

P2P has been described as “an anarchistic threat to the current Internet” (David Streitfeld, *The Washington Post*, July 18, 2000), and Marc Andreesen has called P2P software a “benevolent virus.”

The potential security concerns for P2P software can be categorized as follows:

- Reputation - copyright infringement
- Denial of Service - bandwidth and storage consumption
- Security Holes
- Confidentiality - file sharing
- Malware - trojan horse and virus distribution
- Information Gathering - disclosure of IP and MAC addresses, connection speed

2.1 Reputation

Although most consumers have shown that they are not concerned with copyright violations, companies can potentially be held responsible and, therefore liable, for the copyright violations that the users of P2P within the company commit.

2.2 Denial of Service

Every user of a P2P program is soaking up network bandwidth. Software like Napster, Gnutella and Scour are generally used to download relatively large files such as MP3s, AVIs, MPGs, JPGs, and GIFs. If enough users are downloading these types of files, this can cause network resources to be tied up, resulting in a denial of service. This can occur inside a company or even among cable modem users who share the same network resources. In addition to network bandwidth, full hard disks can also result in a denial of service. If you can’t save the file that you are working on, you will stop working to avoid the risk of losing the work. For example, in some Universities students have downloaded so many MP3 files that the Universities have banned the use of Napster on campus.

2.3 Security Holes

Most P2P software can be manipulated to create and slip through holes in the security architecture of a network. For example, AOL Instant Messenger can allow you to “sniff” for open ports on a peer machine and Gnutella has instructions on their Web site that will allow a user to bypass the port rules on a firewall.

2.4 Malware
Just as the average user can freely distribute any file s/he chooses, malicious users can freely distribute trojan horse applications and viruses. To make matters worse, if the malicious user is using a P2P program such as FreeNet, there is little fear of getting caught since FreeNet does not use a central server and IP addresses are not tracked. As if it couldn't get worse with a model like FreeNet, when a file is requested, it is copied locally from participating client to participating client until it gets to the requesting client. It would be hard to design a better mechanism to spread malware faster. The possibility of introducing and spreading a trojan horse or virus becomes as easy as sending an e-mail.

As has been done with viruses in the past, if you throw in some social engineering, the infection rate can be sped up. Using P2P software, the Zeropaid group found that files that were given fake names referring to pornography were downloaded at an alarming rate (more information can be found at www.zeropaid.com/busted). One can imagine the rate that a new virus could spread through the P2P community if a user gave it a provocative name and description.

Even commercial code can be modified and spread as malware. Programs like AOL Instant Messenger, or any other P2P software, can be reverse engineered and released as Open Source software. This gives malicious hackers the ability to change the software code so that it can be used for other purposes. What better place to put a back door than within a P2P application?

2.5 Information Gathering
The Internet still provides most networks with enough anonymity to feel a certain amount of security through obscurity. P2P software can take away that anonymity. The addresses of routers and gateways are exposed, and if the user is connected directly to the Internet, their IP address and MAC addresses are also exposed. For example, Gnutella users are given the IP address from which they are downloading by widening the column immediately to the right of the "Status" column in the "Downloads" window. This is a good starting place for a hacker with simple scanning software.

Users of some P2P programs like Napster have access to information concerning connection speed. Because most 56k connections and below are dial-up accounts, and probably are not 'always on' and do not have static IP addresses, hackers may not waste their time on these systems. But, connection speeds of 144k and higher can indicate DSL or cable modems that are 'always on' with static IP addresses, a more desirable target.

2.6 Confidentiality
Napster and Gnutella give all clients direct access to files that are stored on a user's hard drive. The files are stored in folders that are not shared by default, but there is the

possibility that other folders can be added and shared. A hacker could figure out what operating system the peer computer has and could connect to folders that are hidden shares, thus gaining access to folders and information that was not meant to be accessed.

3.0 Intel's Peer-to-Peer Trusted Library

Along comes Intel's Peer-to-Peer Trusted Library. Where did it come from, what does it offer, and can it help alleviate some of the security issues surrounding P2P?

3.1 Motivation

Intel has stated that it's intention with this project is to spur open innovation in the peer-to-peer security space and provide a basis for an open-source development project focusing on the security aspects of peer-to-peer. Intel has an unabashed, vested interest in the success of peer-to-peer. "Anything that causes people to use their computer more makes them buy up," said S. Kea Grilley, Intel's director of platform marketing. "We think P2P itself will cause people to use their computer for more and different stuff."

In order to for peer-to-peer to become widespread, Intel understands that better security and standardized methods are a must. At the company's Peer-to-Peer Working Group meeting, Bob Knighten, Intel peer-to-peer evangelist, said, "Security is one of the primary issues that the whole peer-to-peer thing will have to address. Most companies [developing peer-to-peer applications] have thought about security, but they all do it differently."

3.2 Overview

The Peer-to-Peer Trusted Library (PtPTL) allows software developers to add the element of "trust" to their peer-to-peer applications. It provides support for digital certificates, peer authentication, secure storage, public key encryption, digital signatures, digital envelopes, and symmetric key encryption. The library also provides simple support for networking and some operating system primitives, such as threads and locks. Applications built with the library are portable to both Win32 and Linux.

"This is one example, a starting point, that people can use," said Bob Knighten. "Getting peer-to-peer into the hands of end users is the effort. People can build from this [library] secure peer-to-peer clients to avoid some of the security problems encountered in peer-to-peer."

The latest version of the library (currently version 0.2, released April 5, 2001) can be downloaded from Source Forge at http://sourceforge.net/projects/ptptl. It's developed in C++ with support for the Win32 and Linux development environments, and uses the BSD license model. As is the case with most open source projects, contribution to the PtPTL project is highly encouraged. The library is built on top of the open-source OpenSSL Toolkit (http://www.openssl.org/). OpenSSL provides all the low level certificate and cryptographic support and the PtPTL provides high-level interfaces to OpenSSL.

Nelson Minar, CTO for Popular Power, at a recent conference presentation said that the areas of intrusion detection systems, SSL, and Intel's PtPTL will be highly desirable resources for security professionals to master.

3.3 Documentation
HTML API documentation for the PtPTL is available and was built with an internal documentation system, similar to that used by the Gnome Project and the Linux kernel. Basically, a tool is used to extract API information from the source code and build HTML documentation.

3.4 Cryptographic and Network Standards
PtPTL conforms to, and includes support for, the following standards:

* X.509 digital signatures
* PKCS#1 (RSA cryptography)
* PKCS#5 (password-based cryptography)
* PKCS#7 (digital envelopes)
* PKCS#12 (personal information exchange)
* RFC 1421 (privacy enhanced mail format)
* Various standard symmetric encryption algorithms
* HTTP

3.5 Installation
In the MS Windows environment, the Visual C++ compiler and build environment must be installed to build the library and sample applications. In the Linux environment, the GNU g++ compiler and GNU make must be installed to build the library and sample applications.

3.6 Example Applications
The PtPTL comes with three example applications, each complete with source code and make files.

3.6.1 Trutella
Trutella is a simple secure file sharing application. It uses the Gnutella network protocol and is a complete Gnutella client. Trutella supports the notion of secure groups and private file sharing groups for which members can choose who to let in and who to keep out. Secure group requests are encapsulated inside Gnutella requests, so they are passed freely around the GnutellaNet.

3.6.2 Secure File Sharing Application
Secure File Sharing is another simple secure file sharing application. It exchanges digital certificates, authenticates peers, searches, and transfers files over HTTP. Secure File Sharing is considerably more versatile than the Trutella application, especially with regard to certificate distribution.

3.6.3 Cert

Cert is a simple digital certificate manager. It can be used to create, import, export, and store digital certificates in PKCS#12 format. Cert also creates new (randomly-generated) RSA public/private key pairs from which it creates new certificates.

4.0 Conclusion

P2P has made a big splash into the mainstream, and has proven that it adds important new features and benefits to the existing mainstream computing models.

Just as there are security issues with every other computing model, P2P adds it's own take on security issues – specifically reputation (with copyright infringement), denial of service (with bandwidth and storage consumption), creating new security holes, confidentiality (through unauthorized file sharing), malware (through the spread of trojan horses and viruses), and information gathering (through the disclosure of IP and MAC addresses, and connection speed).

The Intel PtPTL is not designed to solve all of the security issues surrounding P2P applications but it is a good building block that directly addresses some of them and helps alleviate others.

For example, by providing the primitives necessary to create services like private file sharing groups and peer authentication, the issue of introducing and spreading malware can be minimized by reducing the likelihood of a malicious user being in the mix. Likewise, confidentiality can be addressed by utilizing the features of secure storage and encryption, thwarting an attacker even if s/he is able to access unauthorized information.

Similarly, security holes and information gathering are still possible, but can be minimized by more secure application design and the use of secure groups and peer authentication to reduce the exposure of the client to malicious users.

The PtPTL doesn't address the issues of copyright infringement or bandwidth and storage consumption, but this is not the goal of a P2P application security API. These issues are not isolated to P2P and the solution lies within a combination of other technologies and the proper enforcement of security and usage policies.

References:

"SourceForge: Project Info – The Peer-to-Peer Trusted Library." Version 0.2. April 5, 2001. URL: http://sourceforge.net/projects/ptptl (3 July 2001).

"The Peer-to-Peer Trusted Library (Release 0.2) README." Version 0.2. April 5, 2001. URL: http://sourceforge.net/docman/display_doc.php?docid=3851&group_id=19950 (3 July 2001).

Barkai, David. "An Introduction to Peer-to-Peer Computing." Intel Developer Update

Magazine. February 2000. URL: http://developer.intel.com/update/departments/initech/it02012.pdf (13 Aug. 2001).

Petruzzi, Mike, et al. "Security Concerns for Peer-to-Peer Software." July 18, 2000. URL: http://www.ktsi.net/pdf_files/Security_Concerns_Peer-to-Peer_KTSI.pdf (13 Aug. 2001).

Moore, Cathleen. "Intel Unveils Peer-to-Peer Security Building Blocks." InfoWorld. February 8, 2001. URL: http://www.nwfusion.com/news/2001/0208intelpeer.html (3 July 2001).

Barnes, Cecily. "Intel Shores Up Peer-to-Peer Security." ZDNet eWeek. February 8, 2001. URL: http://www.zdnet.com/eweek/stories/general/0,11011,2683691,00.html (3 July, 2001).

Solomon, Karen. "Long-term Forecast for Building P2P." c/net Builder.com. March 5, 2001. URL: http://builder.cnet.com/webbuilding/0-7535-8-4950892-4.html (3 July, 2001).

Kabay, M. E. "Peer-to-Peer Software and Security." Network World Security Newsletter. August 28, 2000. URL: http://www.nwfusion.com/newsletters/sec/2000/0828sec1.html?nf (3 July, 2001).