Addendum to My Presentation – FTC P2P Conference, 12/15/04

I am writing to supplement my presentation of December 15th in the panel titled "Technological Responses to Protect Consumers Using P2P File-Sharing Programs." Some of the staff at the FTC requested I do so, given my understanding of certain issues not discussed in the conference.

I wish to address three topics that are critically important with regards to Peer-to-Peer technology and software development. These three issues are (1) the use of encryption, (2) the goal of anonymity, and (3) the "open-source" and international manufacturing of P2P software. Any "solution" to the P2P problem will need to attend to these three issues.

Encryption involves the scrambling and unscrambling of data being downloaded via P2P over the Internet. By definition, anybody intercepting encrypted data will see just random streams of information. Unless the person intercepting the download possesses the enormous computational power to decrypt instantaneously, they are effectively unable to determine the data content in time to prevent the download or act on it. As a result, encryption is being enthusiastically implemented in numerous P2P programs as a solution to avoid music or movie filters that search for recognizable and copyrighted streams of data.

Thus, a Madonna song might be sent past a filter residing on an Internet server. The filter might search for songs that pass through it that have specific characteristics, or "fingerprints." If such a song is found, the file transfer is stopped. However, if one encrypts the Madonna song, it is no longer recognizable as such and the song is downloaded without difficulty. Encryption is also heralded as a method to prevent various authorities from viewing P2P transfers from a trusted source. A porn file-sharing group might use encryption to hide the content of their transfers from others, outside of the group, that might "intercept" the data. Encryption does not, however, hide the identities of the person sending and receiving the P2P file. Thus, encryption alone does not really protect P2P users from lawsuits or legal sanctions. As a result, the P2P file sharing community is generating several programs that claim to be both encrypted and anonymous.

Anonymity is a more difficult problem for P2P manufacturers. The first-generation P2P programs, like Napster, had a central database that recorded all the IP addresses that were offering up a particular piece of data and matched them to those people requesting it. Then, the computers of the two people communicated directly and automatically negotiated the file transfer. This was very efficient, requiring just 3 "transactions" to effect a transfer.

As a result of lawsuits, this approach to P2P was largely eliminated[1]. Instead, programmers created the second-generation P2P programs that are in widespread use today. With these programs, requests for specific data are broadcast to an expanding number of "node" computers until a match is found. The computer with the requested file then communicates directly with the requesting computer and transfers the file. With regards to the use of system resources, this is less efficient but still quite tolerable. For example, assume someone is looking for a somewhat uncommon file. Their request might be sent from their local node to 8 other nodes. Each of these nodes may then forward it on to another 8 nodes. Assume this happens 4 times in total before a match is found. Such a search would now require about 4,098 transactions, rather than the 3 described above. Still, however, the file transfer is not anonymous – the person sending the file knows what IP address is getting it and the person receiving it also knows who sent it.

Again, as a result of recent lawsuits and legal issues, P2P manufacturers have modified their technology in an attempt to offer their users more anonymity. Several third-generation P2P applications are now being made that come closer to such a goal. Two such programs are "Ants" and "Mute." These programs operate in a complex fashion, best exemplified by the process by which a typical ant trail is created.

These programs send out a request for a particular file or song. This request is passed from computer to computer in a somewhat random fashion. When, finally, the request hits a computer that has the desired file, the file is sent back in a similar fashion. It is not sent directly to the requesting computer but, instead, is sent back semi-randomly, from computer to computer. It is as if the foraging ant has just found food and now must find his way back to the hive. The ant does not throw the food back to the hive but, instead, carries it and tries to retrace its footsteps. Thus, the donating computer tries to send the file back the same way it came. But, considerable randomness is added to the file exchanges; the file may end up traveling back a different route. Once one or more routes to (and from) the donating computer are found, the file transfer paths are cemented down – a functional ant trial is created.

This approach allows for near-complete anonymity as no computer knows, exactly, who sent the file or who requested it. Even the donor does not know who they are sending to. In addition, both Ants and Mute encrypt the files; nobody but the source and receiver know what is being sent.

The drawback to such an approach is that it is hugely inefficient and wasteful of computational effort. Due to the complex and semi-random communication between

[1] With the notable exception of extremely popular P2P programs like BitTorrent. Such programs create a highly efficient system where a central list of IP addresses of all active participants *trading a particular file* is maintained. The BitTorrent system is unique in that (1) everybody who uses BitTorrent is both a downloader and a contributor and (2) the central list of active participants is maintained by various websites and not the software manufacturer. These websites are usually overseas. Thus, you typically go to a website with a particular theme (e.g. "action films") and join the "torrent" for a particular download that might interest you (e.g. "Terminator 3")

computers, it is hard to describe this technology in actual transactional numbers. However, as per the example given above, a four-tier search might take about 4,096 transactions to find a match. For the donor computer to then find the computer requesting the file might take another 4,096 transactions. To then cement down that particular pathway for communication might take another several thousand transactions. Let's round to 10,000 transactions, so far. This waste is small, though, relative to the next burden such programs place on networks. When the actual P2P file transfer occurs, it happens piecemeal. Every computer that is part of the "ant trial" is involved. The file is not sent directly from the donating computer to the requesting computer but, rather, is sent in chunks to each adjacent computer along the pathway. Thus, each of the pathway computers will, essentially, have created and deleted a near-duplicate of the requested file.

For a typical P2P file transfers, one might hope the associated inefficiencies of such programs might make them unpopular within the general public. Indeed, one might expect enormous amounts of bandwidth to be chewed up if such programs ever became popular. Regardless, such inefficiencies are less prominent on smaller networks and we might expect to see such near-anonymous and encrypted P2P programs on many small, private networks in the near-future.

Such an event is worrisome as it effectively creates an ideal secure environment for those with more malicious or nefarious intent. The pressure exerted on P2P manufacturers by criminal and civil lawsuits may inadvertently have caused software designers to develop more dangerous and destructive software. The third-generation products, as described above, have significant national security and criminal issues associated with them.

In addition to the creation of encrypted and near-anonymous P2P software, we see one other disconcerting trend. Considerable legal pressure has been generated to impair the functioning of public and private firms engaged in the business of P2P software design and sales. Such pressure has been effective, driving companies like Napster out of business. This has not, however, really harmed the P2P marketplace. Instead, the old consumers of Napster turned their attention to other software developers. Indeed, one way of viewing what happened was that a P2P industry that could be regulated is now decentralized and much harder to control. Moreover, it is alarming that we see the same trend occurring today – we see firms like Sharman Networks being pushed out of profitability rather than regulated. If one is to pursue such a policy, one should also take a hard look at the consequences.

Today, some of the most popular P2P applications are no longer made by formal "firms" but, rather, open-source developers. The motivation for their work is not direct profit but is, instead, something rather more complex. Most open source developers are motivated by some mix of artistic, rebellious, idealistic, and anarchist beliefs. If we outlaw businesses that manufacture P2P software, we should expect that the P2P software development and distribution will nonetheless continue. However, the software will no longer be developed by firms with a profit motive but, instead, by loose groups of

anonymous developers. Their sites will be placed overseas and any attempt to regulate them will give rise to enormous international legal issues.

For current examples of such open-source software developments, one need just consider the enormous popularity of programs like "eMule," "DC++," and "BitTorrent." More worrisome is if we were to push consumers to demand anonymity, such that programs like "Ants" or "Mute" became valid alternatives. The potential for such programs to cause significant bandwidth flooding, degradation in the Internet, and nefarious communication is concerning. Alternatively, such legislation might simply push downloading underground, where it cannot be monitored and, again, becomes venue for more serious illegal activities, such as pedophile movie sharing. The recent popularity of private network P2P programs (e.g. TrustyFiles) is evidence that such a trend is quite likely.

In summary, I wish to urge the FTC and Congress caution before recommending additional legislation. Any such legislation is likely to have unintended results, especially given the typical age of P2P downloaders. New laws would likely aim to curtail an enormously popular activity among adolescents and young adults. Speaking now as a psychiatrist, I would point out that, developmentally, these are rebellious and anti-authoritative ages. Many youth will install the more dangerous technologies rather than give up their illegal downloading of music and films.

Jerald J. Block, MD
Founder, SMARTGuard Software

January 4, 2005