Invisible Internet Project (I2P)

Project Overview – August 28, 2003

http://www.invisiblenet.net / info@invisiblenet.net

Project Overview

InvisibleNet has formed the Invisible Internet Project (I2P) to support the efforts of those trying to build a more free society by offering them an uncensorable, anonymous, and secure communication system. I2P is a development effort producing a variable latency[1], fully distributed[2], autonomous[3], scalable[4], anonymous[5], resilient[6], and secure[7] network. The goal is to be able to operate successfully in arbitrarily hostile environments – even when an organization with unlimited financial and political resources attacks it. All aspects of the network are open source and available without cost, as this should both assure the people using it that the software does what InvisibleNet says it does, as well as enable others to contribute and improve upon it to make use of newer and better ideas to defeat more aggressive attempts to stifle free speech.

I2P is designed to allow people to communicate with each other anonymously – both sender and recipient are unidentifiable to each other as well as third parties. While some application developers will surely implement a proxying system allowing users to browse the normal web anonymously, it is quite likely that the outbound proxies from the anonymous I2P network to the normal Internet will be monitored, disabled, or even taken over for more malicious attacks when that proxy software comes into widespread use. It is for that reason that developers may want to look towards both ends of the network – for example, building both an HTTP/I2P proxy to allow normal web browsers to be used and as well as building support for I2P into various web servers.

Technical Overview[8]

From a technical perspective, I2P is building a peer to peer network that takes advantage of the anonymity and security of mixnets – both free route and mix cascades – the performance, scalability, and resilience of distributed hash tables, and the global interoperability of the Internet. Communication between individuals does not need to expose the location or identity of those communicating – to each other or to a third party attempting to monitor their activity, even if the third party has unlimited resources dedicated to doing so. Supporting those for whom even the use of this software is deemed illegal is an essential goal which will be achieved through allowing trusted peers, strong cryptography, rotating and expiring physical transport addresses, arbitrarily long sender selected delays, non traditional communication

1 Variable latency: while every participant has the ability to adjust the response time of their system to meet their anonymity requirements, current models show that I2P will support strong anonymity with up to sub-second latency in a 5+ million node network.
2 Distributed: no centralized point of failure, control, or monitoring
3 Autonomous: everyone is empowered to operate on the network, and even run parallel networks
4 Scalable: bandwidth, throughput, and latency do not significantly suffer as the network grows
5 Anonymous: individuals control the disclosure of information about themselves
6 Resilient: the network can operate and evolve in the face of attacks and failures
7 Secure: sufficient levels of information integrity and confidentiality are available
8 This is neither a technical spec nor a design document (both of which are located elsewhere). However, a brief technical overview is provided to assist in understanding how I2P works

protocols, and stegonographic techniques.

I2P itself isn't an application that people will use in the traditional sense. To understand where I2P fits into the realm of software systems, think of it as a replacement for the Internet Protocol (IP), not as a specific file sharing, instant messaging, publishing, or other communication application that runs on top of it. Rather than sending information across a socket or out on a datagram from one fixed location (IP address) to another, you simply send information out on the network to a public key, and the I2P software "routers" securely and scalably use whatever means are available to deliver the message, taking into account appropriate anonymity, performance, bandwidth, and reliability constraints, trust metrics, and economic models . An application doesn't need to know where the destination of that message is located, or even how it gets there (and in fact, it can't know either). Inversely, when receiving a message sent over I2P, an application doesn't know where it came from, or even who sent it, unless the sender includes that information. More than that, the machines that route the message from its origin to its final destination have no idea who sent the message or where it is destined, if the message wraps multiple messages in a "garlic", or even if there's a real message involved at all instead of just test data.

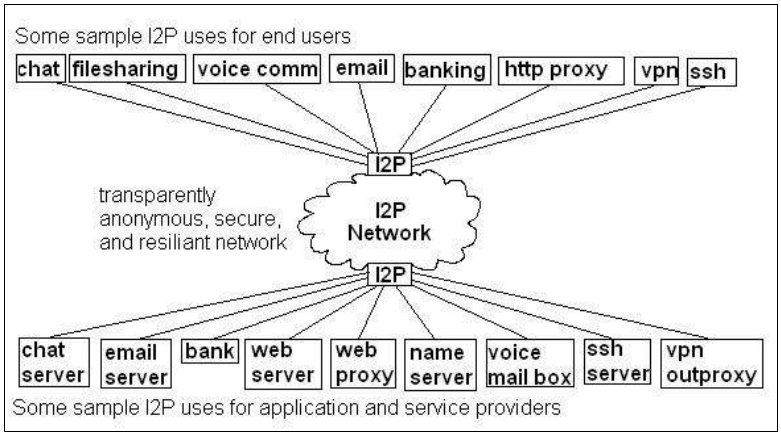

Figure 1: An illustration showing how I2P can fit into traditional application architectures

It is helpful for developers writing applications using I2P understand the concept of location independence, as it allows for faster development cycles and more efficient use of the network. Location independence means that when a message is sent to a destination, there is no difference to the application from when that destination is located half way around the world, if it is just down the hall, or even if it is on the same computer as the sender. This is exactly how TCP/IP sockets work: you open up a socket to some IP address and send data, rather than determine whether that IP address is on the local network, if it needs to go out through a few switches, bridges, and wide area link, or the like. Exploiting this property, I2P includes not only the network software, but the Software Development Kit (I2P SDK) that has APIs in various languages as well as implementations of routers that only support communication with local destinations. With location independence, an application developed using the client API and tested against this local only router will require no changes to be deployed across the fully operational I2P network, in the same way that normal TCP/IP applications can be developed and tested running against the local network.

An important thing to note about the SDK is that even as the network itself evolves, design and implementation of client applications can go on. Regardless of how different routers talk to each other, how messages are mixed between them, how resources are shared, or what programming languages or operating systems are in use, the Invisible Internet Client Protocol (I2CP) will be supported, along with the abstractions outlined in this document. I2P does not exist in a bubble, and through the peer review process, enhancements will certainly be made and criticisms will surely be addressed. Even so, I2P is convinced that the model that the client APIs expose to application developers will meet their needs, even if minor adjustments need to be made along the way, as the APIs are based on the lessons of IP networking and inspired by the resilience of message oriented middleware.

Call for help

I2P is a massive undertaking requiring tremendous efforts from people with many different skills. As the I2P network spec goes forward, during peer review, and beyond we will need the eyes, minds, and voices of many to improve upon it. As router implementation proceeds, we will need developers, cryptographers, testers, and documenters to help build the system. With the SDK available, we need creative and capable developers proficient in various programming languages and operating systems to design and implement applications and proxies to run over I2P. As the network moves towards production, it will be imperative that not only the software works, but that the goals and use of the software be communicated clearly to the end users. And, of course, I2P will need many people running the software to both allow the developers to improve upon it and to add content, purpose, and resources to the applications running over it.

Ideas for developers who want to help abound. Infrastructure such as streaming APIs built on top of the standard message oriented APIs, as well as standards for repliable messages would help application developers contribute more easily. Client applications such as an SSH tunnel, an HTTP proxy, an HTTP server, and a group chat would help the development team proceed using only I2P – running CVS through SSH, publishing and communicating through wikis, web pages, and message boards via the HTTP proxy and server, and discussing the development and support for the network could occur over the group chat. Beyond those ideas are innumerable other applications and services that would help make I2P useful for end users who require its anonymity, security, and resilience.

The I2P development team at InvisibleNet currently has both full time and part time personnel dedicated to the project with skills varied from network design and implementation, cryptography, software engineering, project management, web site design, and documentation. We are always open to new ideas, harsh criticism, and eager volunteers, and we encourage those interested in getting involved to join the mailing list (http://www.invisiblenet.net/iip/devMailinglist.php) and jump into varied discussions on IIP's #iip-dev (http://www.invisiblenet.net/iip/) or join in during the development meetings there on Tuesdays at 21:00GMT.