Arcadia: A Blueprint for Censorship Resistance on the Internet

by

Brandon Keith Wiley, B.A., B.S.

Thesis

Presented to the Faculty of the Graduate School
of The University of Texas at Austin
in Partial Fulfillment
of the Requirements
for the Degree of

Master of Arts

The University of Texas at Austin
December 2005

Arcadia: A Blueprint for Censorship Resistance on the Internet

APPROVED BY

SUPERVISING COMMITTEE:

_____

_____

Arcadia: A Blueprint for Censorship Resistance on the Internet

by

Brandon Keith Wiley, M.A.

The University of Texas at Austin, 2005

SUPERVISOR: Sandy Stone

The majority of academic and open source research into censorship resistance on the Internet has been focused on the very specific problem of giving users plausible deniability against a global eavesdropping attack, mainly through the use of mixnets. Arcadia is a next generation approach to censorship resistance which reevaluates the fundamental assumptions of the field. The Arcadia system presents a new attack model for Internet censorship in the real world and presents a blueprint for designing a system which resists these attacks in the form of a number of design characteristics, the Arcadia Conditions. Possible implementation suggestions which fulfill the Arcadia Conditions are also presented. The result is a fully designed system, which anyone could implement using existing protocols and libraries. Arcadia is a full blueprint for how to design and implement networks to protect websites from censorship.

TABLE OF CONTENTS

1 INTRODUCTION

> *In an ocean of ashes, islands of order. Patterns making themselves out of nothing.*
>
> Tom Stoppard, *Arcadia*

Before a censorship-resistant publication system can be designed, we must clarify what censorship resistance means. What is being censored, who is censoring it, and how is this censorship resisted against in the system? We will confine the scope of the project to censorship of websites. There are many other types of communication: printed publication, person-to-person conversations, online chats, mailing lists, newgroups. However, in order to determine the tolerances of the system in aspects such as latency and throughput, it is useful to limit the scope to a single network protocol and publication model. In order to define censorship and censorship resistance, we will have to explore attackers.

Once it is determined what the attack model is, the general properties of a system which can defend against that attack model can be described. These will be known as the Arcadia Conditions. Any system which fulfills this system can be classified as an Arcadia system. Finally, a specific system, which is called Arcadia, can be put forth which fulfills these conditions, one possible solution to the problem of censorship.

1.1 Related Works

1.1.1 Mixnets

The majority of academic anonymity research is dedicated to the development of increasingly complex mixnets. [1] The goal of a mixnet is to defeat traffic analysis. The attack model here is that the attacker is intercepting traffic between a number of users and a number of servers and they want to know which users are connecting to which servers. Basic mixnet techniques such as winnowing, chaffing, mix cascades, and minimum length paths, are all designed to further protect against this particular attack model. However, this is not an attack which is actually relevant to the real world attackers which Arcadia is designed to defend against. This unfortunately means that the majority of research in anonymity systems is irrelevant to these real world attacks.

The most popular of the current generation of mixnet implementations is Tor, a network which provides onion routing of TCP connections through a set of dedicated volunteer servers. [2] To access a website through Tor, the user makes an encrypted connection to one of the public Tor servers. It then tunnels another encrypted connection through the first one to another server. It repeats this process one more time, connecting to a third server through the first two. It then asks the third server to act as a simple proxy for it, connecting to the actual server. It then makes a normal HTTP request to the webserver. Tor is not web specific, but rather proxies any TCP traffic. Since HTTP is carried on top of TCP, it needs no modification to work through Tor. The web browser connects

to Tor via a normal SOCKS proxy. Other traffic, such as IRC and e-mail, can be tunneled on top of Tor using the same SOCKS proxy interface. Tor also has provisions for hidden servers in the form of rendezvous points. A server makes an outgoing Tor connection to the rendezvous point. Then, the client makes an outgoing connection to the same rendezvous point. The rendezvous point matches up the connections so that the client and the server can establish an encryption tunnel over which to communicate. After that, the client can make a normal HTTP request to the server, or initiate communication in any other protocol that works on top of TCP.

1.1.2 Plausible Deniability

The majority of open source software designed for anonymity is designed to provide a property called “plausible deniability”. This property means that if a user is accused of connecting to a particular server or downloading, uploading or storing a particular file, he can make the claim that no one can prove that this is actually true. Unfortunately, this property is not useful against any of the attackers which Arcadia is designed to defend against. Plausible deniability is a legal defense, but none of the actual attackers take users to court. While they do use legalistic methods such as legal threats, these methods don’t result in going to court and so legal defenses used in court never come into play. In fact, mixnets are indeed a way to provide plausible deniability. Unfortunately, their lack of relevance means that entirely new methods will need to be developed in order to provide useful anonymity.

The most popular tool for plausible deniability is Freenet. [3] There have been many versions of Freenet and at least three different architectures. The first one, based on the routing algorithm described in the original Freenet paper, was the most popular, at least in terms of press attention. The original Freenet routing algorithm was very similar to modern DHT algorithms such as Chord, [4] but with nondeterministic behavior. The idea behind Freenet was that the nodes form a data storage network. Users publish web-site content into the network. Other users can then use a Freenet HTTP proxy to browse the content in the network with their browser. This design has censorship resistance qualities by letting the network host the content rather than the person who originally found or created the content. Additionally, it moves the content semi-randomly around in the network so that it is difficult to find and eradicate all copies, and encrypts locally stored data so that a node does not know what content it is storing.

Surprisingly, despite the lack of relevance to real world attacks that the academic and open source designs have, the systems which implement them are in fact useful to those seeking anonymity. Both Tor and Freenet have been used successfully by people in China to access websites in the United States which are normally blocked in China. This is, however, the result not of their different designs, but because of a common quality which both systems share: that they allow users to contact the websites through the nodes of the system instead of directly. In effect, they are both useful only in that they both act as proxies. A network of unencrypted single-hop proxies would work just as well as the complicated designs which Tor and Freenet implement. It has yet to be demonstrated that more than one hop is necessary or useful against real attackers. However, in order to pro-

vide a sense of comfort and security to users, additional mixing could be made available in order to increase the popularity of the system. In particular, hidden servers may desire more than one hop.

1.1.3 Darknets

A popular trend in recent peer-to-peer development has been “darknets”, a term which has recently been redefined to refer to peer-to-peer networks in which all of the links between nodes follow links in a social network. In other words, you only let your node connect to the nodes of your “friends”. There are both commercial ventures in this space, such as Grouper and FolderShare, and open source projects, such as WASTE and the most recent incarnation of Freenet. Following small world graph theory, the number of hops to get from one point in the network to another should be similar to that of a distributed hashtable (DHT). However, such systems have limited application to censorship resistance. In a situation such as China, where access to certain information is forbidden by the government, the last thing you’d want is to implicate all of your friends in your dissident browsing. Similarly, for those running hidden servers with highly controversial content, you may want to remain anonymous from your friends, who may not approve of your secret political affiliations. In situations such as this, the greatest protection is not from your friends, but from strangers. The assumption is that a random set of strangers will not care very much about any particular person, website, or content. The separate motivations of the individuals will balance out into an even tolerance, except for things

which are considered universally unacceptable, which will be rejected by the consensus of random strangers.

1.2 The Economics of Censorship

One of the most damaging attitudes in the anonymity, censorship resistance, and peer-to-peer communities is a black and white attitude towards resistance to attacks. The feeling is that if an attack can be imagined which might cripple or corrupt a design, then the design is without value. Unfortunately, this attitude does not result in perfect designs. Instead, it results in designs which are either sufficiently complicated, or sufficiently surrounded by complex mathematical explanations that are hard for anyone to understand, and thus hard for anyone to convincingly describe an attack. Being unable to imagine an attack is used as a replacement for proving that it is unattackable. This level of complexity does not produce better systems, just systems whose value is difficult to determine. An unfortunate side effect of these complex systems is that they are hard to implement and it is difficult to determine if an implementation of the design should be considered a proper or degenerate implementation. So the empirical results found in running the implementation can be attributed as much to the implementation as to the design.

The goal of Arcadia is to reverse this trend in a number of ways. First of all, attacks should not be considered an all-or-nothing affair. The various possible attacks must be arranged into tiers of priority. First-tier attacks should have precedence in the design over second-tier attacks and so on. Additionally, the goal is not to make the listed attacks impossible, which is an impossible goal, but to raise the cost of the attacks above what it

currently costs to attack existing systems. By raising the cost of censorship, the amount of censorship can be reduced. By raising the cost primarily of the attacks which are most common, the effort expended on the design can have the maximum effect on reducing censorship in the real world. In addition to developing a more useful attack model, the goal of Arcadia is to provide design constraints. This will reduce the complexity of both the design and implementation, and simplify the analysis of the results. Additions which will imbue the design with unknown characteristics and complexify analysis of its properties, such as novel encryption algorithms or steganographic encoding, will be eschewed in favor of a design which can be easily understood and implemented. One of the fundamental necessities of any anonymity system is having a sufficiently large base of adoption, so this is considered to be a higher priority than addressing every possible criticism of the design with increasingly complex workarounds.

1.2.1 The Economics of Identity

At the core of the economics of censorship is the economics of identity. As will be discussed later in the section of choosing identifiers, one of the most economical ways of succeeding in censorship, once normal censorship methods have become expensive due to the popularity of censorship resistance networks, is to undermine the censorship resistance network. It is necessary, in order to infiltrate and destroy censorship resistance networks, to be able to either choose identities at will, or be able to generate a large number of possible identities, as choosing from the pool of already generated identities is sufficiently similar to being able to choose an identity at will.

In order for networks to be able to withstand attacks, the economics of identity need to be understood. Identities need to be expensive to obtain, such that the number of identities that an attacker can own is far less than the number of real users. When the number of attacker identities outnumbers the number of real users, the attacker has control of the network. When the number of attacker identities is small, however, the damage that the attacker can do to a well-designed network should be inconsequential. While there are a number of other possible attacks that are not linked to identity, identity generation in most systems is the most inexpensive route. So, in order for a network to survive long enough to be able to experience the other possible attacks and have a chance to defend against them, the cost of generating new identities must be sufficiently high.

1.2.2 The Economics of Attention

As is discussed later, making the cost of obtaining a new identity sufficiently high is not an easy task. The Sybil attack [5] invalidates many of the proposed methods for buying identifiers. The most popular types of identifiers used today are IP addresses and e-mail addresses, both of which are extremely inexpensive to the various attackers that we include in our attack model.

What has been emerging lately, which is of particular interest, is economic systems based on attention. Reputation systems and social networks are examples of this sort of economics. Continuous active work on the part of the user is required in order to maintain a position, although there are often ways to radically reduce the cost of real human attention through automated and streamlining the interface. Wikipedia is a good example

of attention being the currency with which control can be bought. Wikipedia allows (for the most part) anyone to edit any page. Therefore, you can spend many hours finely crafting an entry and a minute later an anonymous vandal could come along and change it to something unrelated and offensive. You are, however, free to change it back, and the vandal is then free to vandalize it again. The result is that people who care about the contents of certain pages monitor them for changes and fix any defacement or mistakes, often within minutes of the offending change. Quality in the entries is maintained entirely by obsessive people that refuse to give up. They continuously spend attention in order to exert control. This usually works because vandals generally have less attention to spend on any one thing than people who really care about the subject. Cases do arise occasionally where fanatical fans of different points of view battle over a page at great personal cost (in terms of attention) until the Wikipedia administrators eventually lock it from further changes. However, for the most part, the economics of attention provide a robust means of keeping Wikipedia in reasonable order without centralization.

2 DESIGN

> *Nobody would kill a man and then pan his book. I mean, not in that order. So he must have borrowed the book, written the review, posted it, seduced Mrs Chater, fought a duel and departed, all in the space of two or three days. Who would do that?*
>
> Tom Stoppard, *Arcadia*

2.1 Designing an Attack Model

The first step in designing a censorship-resistant publication system is to determine which attackers the system is designed to protect against and the relative priority of these attackers. This is an often overlooked step, as systems are designed to provide "perfect" security against "all" attackers. This is equivalent to prioritizing all attackers equally or randomly. Since not all attacks are equally likely or costly in the real world, this leads to a suboptimal design. The best possible design for solving real censorship issues can only be achieved by first examining which attacks on censorship are currently occurring in the world, their relative cost, and the relative severity of the results of the attacks.

2.2 Attack Model

A necessary prelude to choosing attackers is to first determine which attacks have actually occurred. As there are no clear statistics on censorship incidents, the attackers and attacks here have been taken from stories about online censorship in the Your Rights Online section of the popular news site slashdot.org. From these examples, we can esti-

mate the popularity and severity of attacks, as well as the resources of the attackers. Furthermore, from this information we can determine a general set of vulnerabilities which open web sites to attack, and determine what characteristics a censorship-resistant system needs to possess in order to be resistant to real attacks.

These real attacks against websites form the first tier of our attack model, the attacks which our system must defend against if it is to be useful to the websites which are currently under threat of censorship, or other websites which may be faced with similar attacks in the future. The second tier of our attack model is attacks which are currently used to attack censorship resistance and anonymity services. These should be dealt with to some extent if the system is to survive, but not at the expense of responding properly to the first tier. The third tier attacks are attacks which could, hypothetically, be used against censorship resistance and anonymity services, but which are not currently in common use in the real world. Some of these will be discussed briefly, but are not of actual consequence to a real censorship resistance system. The main goal of discussing these attacks is to show that, even though they were not specifically included in the design criteria for Arcadia, some measure of defense against them is still provided.

2.2.1 First Tier

2.2.1.1 RIAA

The RIAA's purported goal is to crack down on sharing of music not authorized by the copyright holder. However, in this course they have also curtailed much author-

ized music sharing. The RIAA has, in the past, shut down sites aimed at authorized music distribution under the pretense of it being possible that they were also distributing unauthorized music. Since there is no method to technically determine whether a music file contains content under the jurisdiction of the RIAA, there is no technical means to prove that no RIAA music is being shared. The only mechanism to do this is through the legal system, by going to trial against the RIAA lawyers, a costly venture with unknowable results. Thus, the RIAA can attack whomever they want. Their means of attack is legalistic. They send threatening letters to websites, companies that make peer-to-peer file sharing software, and users of peer-to-peer file sharing software. Other similar associations, such as the Harry Fox Agency, have used the same technique to shut down websites with material like guitar tablature and television show transcripts. It is important to note that none of these sites have been found to actually be doing anything illegal. No court case has set a precedent on the legality of hosting such content. The RIAA and similar organizations need not prove that their claims of illegality are valid, as a threatening letter alleging illegal activity is sufficient to shut down most websites. If the website maintainer does not comply, then a similar letter to the ISP will usually be effective. The RIAA has increased the severity of its attacks by including a statement in its letter to Kazaa users that in order to avoid litigation, they must send the RIAA money, ranging from $3,000 to $7,000. While sending this money to the RIAA provides no actual legal protection for the recipient of the letter, it has caused to RIAA to receive a significant number checks for the simple act of sending a letter.

2.2.1.2 Popularity

Popularity is a strange form of self-censorship on the Internet. Many websites are hosted on servers which either do not have the capacity to server a large number of simultaneous users, or are hosted under a pricing plan which is a flat rate up to a certain level of popularity and then becomes drastically more expensive as the popularity rises, or else automatically shuts down the website once a certain threshold of popularity is reached. This is sometimes called the Slashdot Effect because the popular news site Slashdot.org often links to such small websites without the proper provisions for popularity. The rapid rise in popularity from being mentioned in an article on Slashdot.org is often enough to take that same website down.

2.2.1.3 The Church of Scientology

Many claims have been made about the Church of Scientology and its attacks against those who publish negative material about it. Primarily, it is alleged that they have harassed website maintainers with threats of violence. While hypothetical situations involving political dissidents being shot for using circumvention software in totalitarian regimes are often discussed, here is an example in the United States where running a website might result in physical violence to your person. While not all of these claims can be verified as structured efforts by the organization, such alleged attacks are important, at the very least because they are a perceived threat which publishers will worry about when

publishing information which they think might draw the guile of such an organization. Therefore, a censorship-resistant publication system should protect against such attacks.

2.2.1.4 The Government of China

The Chinese government has a unique and very interesting method of censoring websites. All Internet traffic between China and the rest of the world passes through government owned computers. These computers scan Internet connections to see if they are connecting from inside China to an outside computer on a blacklist of known banned sites. If so, the connection is rejected. This stops people inside China from connecting to news sites outside of China that carry new stories critical of the Chinese government.

2.2.1.5 Summary of Attacks

The attacks of our various attackers can be broken down into a few fundamental attack strategies. Both the RIAA and the Church of Scientology rely on the identity of the website maintainer to be discovered. This is usually done by obtaining the IP address or domain name of the website, and then contacting various entities, such as the maintainer's Internet Service Provider and domain name registrar, in order to link the IP address or domain name with their maintainer's real address. The government of China blocks connections leaving China to any IP address which is on a known blacklist of websites which should be blocked. Censorship by popularity occurs by either overtaxing the server, or using more than the maintainer's allowed amount of bandwidth trying to serve the website to a large number of people in a small amount of time. These basic

types of attacks lead us to the properties which our system must have to be useful. The system must hide servers behind the network of nodes and provide a cache between the client and the server so that not all requests destined for popular servers need to be fulfilled by the actual server. These properties will be discussed more later in the section on the Arcadia Conditions.

2.2.2 Second Tier

The second tier of our attack model consists of attacks which have been used against existing anonymity systems. The systems included will be limited to the most popular systems which are in the same space as Arcadia, those which allow browsing of websites which are in danger of censorship.

2.2.2.1 Freenet vs. The Media

Despite it's popularity and the number of attacks suggested for Freenet, few were actually implemented. This is most likely because Freenet was sufficiently unstable that it wasn't enough of a threat that any of the first tier attackers bothered to interfere with it. The greatest threats to Freenet were its own design and implementation. The indeterministic nature of its routing alogorithm made it difficult to debug the network. The use of an original wire protocol orthogonal to, but incompatible with, HTTP and original cryptographic protocols made independent implementations almost impossible. The amount of original code necessary to implement all of these original protocols made it difficult to maintain and debug the one working implementation. The implementation was used as a

de facto standard on the design and protocols, which further complicated making independent implementations.

The main external attack on Freenet came from the media. Despite Freenet's altruistic goals of fighting for freedom of speech on the Internet, it was decried by the sensationalist media as a sanctuary for every type of unsavory activity imaginable. The specific focus of many stories was the secret cost of freedom of speech, the propagation of the most hated types of files that the media could think of: child pornography, coordination of terrorist activities, instructions for making explosives, and copyright infringement. This coverage had no basis in reality and was purely speculative, but was presented as fact, and resulted in much hateful e-mail being sent to the Freenet developers. However, this negative press coverage actually increased the adoption of the system. Intelligent users were able to read between the lines of the media coverage and saw a tool that could be used to help solve their censorship problems. Users in China found that it allowed them to bypass the firewall if someone outside of China periodically inserted copies of the desired website into the network. Copies of anti-Scientology websites and other censored material began to appear on the network.

2.2.2.2 Tor vs. Wikipedia

Tor has been much more successful than Freenet in terms of actually working, and so has gathered even more users, particularly those in China. Because it can be used to browse all websites and not just those that are inserted into the network, it has proven in-

stantly more useful than Freenet for this demographic. Due to its popularity, it has come under attack from a number of sources.

The most interesting attack has been from Wikipedia, with similar attacks from a number of IRC networks. Wikipedia and the IRC networks have apparently decided that Tor users are disreputable people who should be banned as a whole just for using Tor. The maintainer of Wikipedia has said that anonymous users have no business editing Wikipedia entries. Normally Wikipedia allows anonymous editing since users are not required to log in and are identified only by their IP address. Wikipedia normally bans IP addresses that repeatedly deface entries, although defacement is itself difficult to define in an anonymous collaborative editing project. However, despite normally allowing anonymous editing, using Tor is considered indicative of malign intentions, and so Tor users are universally banned. This is easy for Wikipedia to implement because Tor publishes a global list of all of their proxy servers. So in order to block Tor users, an attacker needs only to download this list and add it to their local blacklist.

2.2.2.3 Tor vs. BitTorrent

The other major attack that Tor has faced is due to its sudden increase in popularity when it was mentioned on Slashdot.org. Users of the popular BitTorrent file-sharing software read about it and, because any TCP-based protocol can be tunneled over Tor, they decided to tunnel BitTorrent over Tor. Since BitTorrent is designed to maximize the use of the bandwidth available to it, and since the Tor network proxies all traffic through a small set of dedicated servers, all of the bandwidth of the network was quickly ex-

hausted, bringing the network to a crawl. Tor was unable to regulate the bandwidth used because it has no knowledge of the content it is relaying except at the exit nodes, which actually connect to the destination server. Within the network, there are only encrypted connections between nodes. Therefore, a single client can make a connection through every Tor node, using a great deal of bandwidth per client. Normally there is no benefit from doing this except to attack the network, but BitTorrent automates this process and uses all of the available bandwidth to increase download speed. The Tor developers worked around this issue by blocking the default BitTorrent port on the exit nodes. This doesn't actually prevent the use of BitTorrent over Tor, but raises the attention cost such that the amount of BitTorrent traffic over Tor dropped to a manageable level.

2.2.1 Third Tier

There are a number of properties which other censorship-resistant system designs have attempted to attain, but which are not included here because there is insufficient evidence that they are useful. High among their ranks is "plausible deniability", which is the property that a user of the system can make the claim that files which they've downloaded were not downloaded intentionally and were placed there by the system. This property is not considered useful as it is a legal protection and we include no legal attackers in our attack model. Plausible deniability does not stop an RIAA or Church of Scientology attack, but hiding your IP address will.

Another major property which is not included in our system but which is a major aspect of much research in this area is the unlinkability of the client and the server. A

number of sophisticated techniques, such as mixing and winnowing and chafing, are unnecessary if this is not a desirable property. [6] We ignore this property because it assumes an attacker who is monitoring both ends of the connection and attempting to determine which clients are talking to which servers. We have no attacker who has this capability in our attack model. This simplifies our problem somewhat, but unfortunately this also makes a large body of work in this area irrelevant.

2.3 The Arcadia Conditions

The basic design of the Arcadia system is a set of nodes which organize themselves into a distributed hashtable. A node in the network keeps track of its neighbors just like in any other DHT. Nodes also provide two additional services: acting as HTTP proxies for other nodes and local web browsers; and, providing for any other node a list of neighbors for that node, so that new nodes have a way to join the network and find their neighbors starting with any node.

The implementation of a system which addresses the attack model detailed here has much room for variation. There are a number of design choices which are arguable, and particular implementations of features may have as yet unknown flaws. Thus, the implementation described here may not be the only or final possible implementation of the ideas presented here. However, any implementation must follow certain design criteria to be considered a proper implementation of these design principles. These criteria will be referred to as the Arcadia Conditions.

2.3.1 Implementation Constraints

In order to succeed, an Arcadia system must achieve significant adoption so that the size of the crowd is large enough to provide sufficient anonymity. In order to achieve such a level of penetration, some conditions must be imposed on the implementation, which are separate from the rest of the design:

(1.1) No cryptographic protocols should be implemented.

(1.2) No wire-level protocols should be implemented.

The majority of time spent on other similar projects, such as Freenet and Tor is spent implementing new and untested cryptographic protocols with unknown security issues and new wire-level protocols which must be understood in order to make compatible implementations. In order to avoid this pitfall, any Arcadia system must use existing protocols and cryptography libraries. Standards such as HTTP and SSL should be all that is necessary to build such a system.

2.3.2 Defenses Against First Tier Attacks

Any Arcadia implementation must defend against the first tier attackers. In order to accomplish this, it must fulfill some conditions:

(2.1) Allow proxying through the network to a website specified by domain

(2.2) Allow proxying through the network to another node specified by node identifier

(2.3) All website requests must be cached by the network

2.3.3 Distributed Hashtable Characteristics

In order to implement condition 2.2, a distributed hashtable (DHT) is necessary to find a node by its identifier. So that the DHT can defend against attacks, some general design guidelines should be followed.

(3.1) Make IP fishing difficult

(3.2) Allow Byzantine checking of node connections

(3.3) Keep the number of connections per node to an optimal crowd size

There are many ways to fulfill the Arcadia Conditions. They can be considered guidelines for developing solution to the problem of censorship on the Internet. In the following section, specific ways to fulfill these characteristics will be discussed.

2 IMPLEMENTATION

> *Anonymous networks have plenty of good uses, however editing wikipedia is not one of them, at least not until either the anonymous network or wikipedia has set up some sort of trust mechanism based on something other than the scarcity of accessible IPs.*
>
> Roger Dingeldine

While there are many ways to fulfill the Arcadia Conditions, there are certain characteristics of a system which, while not entirely necessary, are nevertheless desirable. This section will address suggestions various implementation details, which, if followed, will result in a system with a number of desirable qualities, and which is resistant to a number of second and third tier attacks.

3.1 Security

Since Arcadia provides anonymity to its users, and anonymity can be used to commit crimes and other undesirable acts, limiting hacking through the anonymizing network is a concern that must be addressed even though it is not really an attack against the network. The negative associations which could be associated with a system to its actual or perceived potential use as a platform for hacking, spamming, and trolling could possibly reduce the number of users. On the other hand, infamy does not seem to have reduced the adoption of Freenet or Tor. However, it is still desirable to limit the harm that

can be done through the anonymizer as long as this can be done without impeding the positive uses of the system.

3.1.1 Limitations

The fundamental philosophical tenet of Arcadia is that everyone should have the opportunity to voice his or her opinion, but that does not necessarily mean that anyone should be forced to listen. Thus, while the system should make it possible for anyone to run a website without fear of it being shut down, it is not necessarily the case that anyone might be interested in reading their website. There are a number of restrictions imposed on the system in order to ensure that it is used to transmit information only to those who want to read it, which also reduce other unsavory uses of the network.

First, the network only caches information: it does not store it permanently. In order to publish, you must find a place to host your website. In the case of material which no one anywhere will agree to host, it is impossible to publish it through the network. Second, the network only supports HTTP, which means that it cannot be used for other protocols such as SMTP for sending spam. While it is possible to use tricks, such as making what looks like an HTTP request to a non-HTTP server in order to establish a connection with it and hack it directly, this is limited in Arcadia, as a direct connection to the server is never available. All HTTP requests are fully processed and cached before the results are sent back to the originator. Invalid HTTP responses will result in a termination of the connection.

Third, only cachable HTTP responses are allowed. This has a two-fold purpose. It assures that the network is being used for publication and not for one-on-one correspondence. Something like a TCP over HTTP tunnel is impossible to establish if all of the responses are cached. Additionally, since HTTP POST requests are not cachable, POST is not allowed. This will eliminate posting to most web forums such as Wikipedia. While some anonymity proponents will certainly claim that anonymous posting to Wikipedia has value, and while this is probably true, it is not within the scope of the design. The goal is to give people a way to publish information, not a way to enforce their views on others. It is entirely possible for someone to have an anonymous website which a non-anonymous Wikipedia author uses as a source for an article. Thus, the ability to support the dispersal of important information anonymously is still maintained without opening up the possibilities for abuse that anonymous posting to public forums entails. Most public forums are simply not structured to support anonymous posting socially, and if they are then they will incorporate their own anonymous posting feature.

3.1.2 Consequences

There is still the possibility of hacking websites with only HTTP GET. To some extent, the responsibility of maintaining security of webservers must fall to the website maintainers. However, by limiting the communication which Arcadia will relay to the subset most useful to censorship-resistant publishing, the opportunities for hacking and other such abuses are significantly reduced. The system administrators will have to fix the remaining security vulnerabilities in their websites. If their websites follow the HTTP

protocol specification, which specifies that GET operations should have no side effects, they should be entirely unhackable via Arcadia. In fact, they are more protected from Arcadia users than normal Internet users, as attempts to flood their server with requests will be blocked by the network's caching and the attacker will succeed only in flooding the nodes that it connects to and not bothering the webserver at all.

The question is raised, however, of how, if the system integrates directly with the web browser via a proxy, but there are a number of POST-based websites which are not available through the system, how the browser integration will funtion properly. There are a number of ways to address this. One way is to simply refuse to connect to sites which cannot be connected to anonymously. The user can switch into and out of anonymous mode manually by changing their proxy settings. The proxy could also support this behavior internally, connecting directly to non-anonymous sites. However, there would need to be some indication to the users when they were leaving an anonymous site to enter a non-anonymous one so that they could choose if they wanted to do this. Alternatively, a secondary anonymity system such as Tor could be used to connect to sites which Arcadia doesn't support. This last suggestion somewhat begs the question, as systems such as Tor lack the same scalability as Arcadia, and are already blocked by some sites such as Wikipedia. Even if a secondary system was used, it may still be impossible to connect to the site except for directly. The most probable solution, then, is some sort of browser extension which will alert the user of non-anonymous sites and let them choose to enter or not, similar to the warnings which are currently found in browsers when moving between encrypted and unencrypted web pages.

3.2 Authentication of Content

An issue, which comes to light as a result of caching all web pages which go through the system, is determining if the copy of the website that you receive is actually the website that you requested and has not been corrupted. This is normally done using SSL encryption end-to-end between the browser and the website. This is impossible in Arcadia, however, because there is a cache between the browser and the website. Choosing a random exit node to connect to the webserver provides some security against tampering of specific content, assuming that the majority of nodes are not “evil” (owned by the attacker). An evil node cannot choose which websites it is asked to connect to. However, if an evil node wants to tamper with all web content then it can do so. In a caching chain, this problem is worse since an evil node can tamper with content if it is anywhere in the chain, not just if it is the exit node. A Byzantine solution is possible. Byzantine systems works by choosing three random nodes and asking them the same question. If two out of three of the nodes give the same answer, the third node is considered to be incorrect. Using a Byzantine solution to this problem would require requesting the file from three different nodes. This increases the bandwidth cost significantly, as three entire copies must be downloaded of every file. However, it is not an unreasonable approach. The only other methods are to connect directly to the web server to directly download and verify random chunks, which would entirely defeat the anonymity, or to use a byzantine caching, system but only for the file's hash tree. This would work best if web servers provided hash trees in the headers, as then only one copy of the file would need to be

downloaded, but it would still be beneficial even if three copies had to be downloaded, because only one would need to actually be stored.

Other systems do not really deal with this problem. Tor provides an end-to-end connection between the browser and the web server and does no caching. Freenet provides hash-based keys, which could be used to verify the file, but there is an important distinction to be made here because these hashes do not come from the original website. Rather, in Freenet, someone must always upload a copy of the website into the network. It is the uploader that computes the hashkey. There is no guarantee, other than the word of the uploader, whom is anonymous, that the file has any relation to the website of which it purports to be a copy. In reality, it is the person who links to the Freenet copy of the website that makes this claim, as the hashkey is embedded in the hyperlink. In a caching system such as Arcadia, embedding hashkeys in hyperlinks can still be done. This will allow readers to check whether the content has been modified since the time that the link was generated.

Including the hash in the URL is optional and only makes sense on anonymous websites where it is known that the user is using an Arcadia proxy, as the hash needs to be stripped out of the URL. Another option, which was used in Freenet, is to include a signature in the URL. However, in the case of hidden servers in Arcadia, the websites are reached by using the public key in the URL instead of a normal domain name. Therefore, the necessary information to verify a signature is already in the URL. The signature itself can be included by the anonymous webserver in the HTTP headers and will then be

cached along with the file. This doesn't allow for signatures of non-anonymous websites, but having signatures for non-hidden servers doesn’t make sense in a caching-only network such as Arcadia. Using signatures is a valid alternative to validating anonymous websites other than byzantine fault tolerance, but requires additional cryptography, and so comes temptingly close to violating condition 1.1.

3.3 Bidirectional Communication

The limitations imposed here for the sake of security may seem extreme, particularly the exclusion of POST, as it greatly limits the number of websites which can be viewed anonymously. However, these measures are a technical necessity and the security benefits are just useful side effects. In order to protect the network against flooding, all requests must be cached. Thus, uncachable requests must be denied. This does raise the question, however, of how certain functionality fundamental to communication can be regained. For instance, if someone were to run an anonymous weblog, how would people submit news stories? They cannot post their stories on the website because POST requests are blocked, and they cannot contact the author because the author is anonymous. One solution is to use external methods, such as an anonymous remailer system, to contact side authors. This, however, begs the question, much as using Tor to browse Wikipedia does. Anonymous remailers are blocked from China, for instance, so Chinese users could not submit news stories to their own anonymous website.

A system could conceivably be developed which allows for posting of items to a website without violating the fundamental tenets of the design. The key is to allow bidi-

rectional communication without allowing for private communication or non-cachable traffic. If we start with the idea that everyone using Arcadia is capable of running an anonymous website, the initial idea is clear. If you have suggestions for news stories, post them on your website. Then, if the maintainer of the news website is interested in what you have to say, they can take your story and post it on their own website. This is a reasonable plan, but the question remains as to how the maintainer will know that you posted a story for them to read. Normally, bloggers find out about each other via referrer headers in requests. However, this won't work in Arcadia, as the majority of requests never make it to the server, instead being responded to by one of the nodes with a cached copy. Therefore, there may be lots of anonymous blogs all over the network that would never find out about each other. The only method to publicize your website would be to tell someone directly, and thus lose your anonymity, or to tell someone via a secondary method which might not work. Being able to index and find anonymous blogs is a great asset when the purpose of the system is to allow people to find and read anonymous websites. This can be easily achieved by crawling the known anonymous websites, just like on the normal web, as long as there is some way for existing anonymous websites to find out about new ones.

The solution to this issue is to use the DHT itself as a very limited publication tool. There are a couple of ways to approach this. The most minimal way is to create a way to scan the entire network for all hidden servers. This would be very useful if you were running an anonymous search engine, as you would want to index every site. However, if you are a news site looking for story submissions, this is a very inefficient

method. In order to save every site that wants to take submissions from scanning every hidden server periodically, a second service could be provided which would allow a website to register with the DHT when it would like to be checked for new content. For instance, if you had a news story to submit to a Chinese website, you would tell the DHT to add your URL to list of URLs for that website. The website would then periodically ask the DHT for its list of URLs, and check them for new submissions. Websites which are uninterested in submissions need not ever check the DHT for submissions, and so incur no additional burden. The only trick is assuring that the DHT is not overloaded by this traffic. Allowing users to store an unlimited number of pieces of arbitrary data in the DHT opens up all sorts of abuses, so this feature must be added very carefully.

3.4 DHT Design

There is more than one possible DHT implementation which could fulfill the Arcadia conditions. However, a sample implementation must be proposed in order to give a starting place for designing or implementing Arcadia systems. In order to fulfill condition 3.3, a DHT where the number of connections per node does not grow with the number of nodes in the network, such as Koorde, [7] is desirable. However, to fulfill condition 3.1, we must implement a system similar to Achord, [8] where nodes don't communicate with any other nodes except the ones in their own table. Thus, it is useful to have bidirectional connections, such as in Kademlia. Additionally, fulfilling condition 3.2 requires multiple redundant paths so that the node connections can be double-checked, a property not generally ascribed to existing DHT systems.

In order to fulfill all three conditions, a new DHT is suggested which can be referred to as Byzantine Symmetric Koorde, or simply the Arcadia DHT. It is based on Koorde fundamentally, in that it uses a similar method to Koorde for making connections: in order to determine the identifiers of the nodes you should connect to, take the current node's identifier, bitshift it one space to the *left* and replace the rightmost bit with a 0 to get the first identifier, and a 1 to get the second identifier, resulting in two node identifiers to which the current node will connect. In addition to the normal Koorde connection criteria, the Arcadia DHT also connects to nodes which are generated by bitshifting one space *right* and then replacing the leftmost bit with a 0 and a 1, resulting in two *more* connections. Additionally, Arcadia also connects to the nodes which can be reached with a bitshift two places either left or right. This creates a total of 16 connections per node. Since there is overlap between a node's neighbors and a node's neighbor's neighbors, a node can ask its neighbors to double-check the connections. Also, since Arcadia makes connections both by right and left bitshifting, there are two paths to any destination, the left path and the right path. This allows double-checking of downloads. Also, since there are eight connections per direction instead of just two total connections, there are multiple starting nodes for any path in case one of the nodes in the path is suspected of being evil. The existence of 16 neighbors is also useful in providing a sufficiently large set of nodes for an anonymizing mixnet.

3.4.1 Proxy Service

The most important of all of the Arcadia services is the proxy service. The basic functionality is that a client connects to the node and specifies the address of a server to connect to, either a domain name or a node identifier. The node then chooses one of its existing peers, connects to it, and passes on the request. When a response comes in, it caches the result and then returns it to the client. In order to integrate as easily as possible with existing web browsers, the proxy should be implemented as an HTTP proxy. Caching HTTP proxy implementations already exist which could be adapted to chain to other proxies based on URL or node identifier.

3.4.2 Joining and Parting

Many DHT systems have complicated join and part schemes so that they can repartition replicated data and reassign connected nodes to new peers. Luckily, since Arcadia is only a caching network and not a data storage network, any repartitioning of cached data is at most a bandwidth optimization. It may, in fact, not even be necessary at all since the chain of caches ensures that popular data is already replicated in multiple places. The only additional bandwidth used is when a new node with an empty cache joins and is then sent queries that it cannot answer, resulting in it connecting to the server to obtain a copy of the file it needs to cache. Since popular files will already be cached elsewhere, these queries will only be for a small set of file requests which cannot be answered elsewhere. The node would have to spend the bandwidth to obtain a cached copy,

whether it was from the server or another node, so the only additional cost is to the server. Since this will only occur for files which are not particularly popular, the additional overhead that occurs when nodes churn in the network will probably be insignificant to the server. Therefore, the only join and part procedure that is necessary is to update the connections between nodes. This is important in Arcadia because nodes only talk to their neighbors. The only conversation they will have with non-neighbors is to direct that questioning node to its own neighbors. Due to this property, it is important to keep the connections up to date. Luckily, since there is redundancy in the node connections in the Arcadia DHT, and the set of neighbors is small, a node only needs to update those of its neighbors that will want to connect to the new node whenever a new node joins. Updating all interested neighbors ensures that the update does not get lost while being forwarded. While this does create redundant traffic, since the number of neighbors is constant despite the size of the network, this is probably a manageable amount of traffic, and the algorithm is certainly simple enough to implement, which will aid in adoption. The other important element of the join protocol is to verify the identifier of the joining node. In order for the DHT to protect against IP fishing, it is important that node cannot forge new identifiers easily. Therefore, identifiers must always be checked before communication is allowed.

3.4.3 Sparsely Populated Networks

One problem with the Arcadia DHT, which will be obvious to anyone familiar with its predecessor, Koorde, is that it has not yet been specified how it will deal with

sparsely populated networks. The properties attributed to the Arcadia DHT so far require a network which is fully populated with a node for every possible node identifier. This is clearly not the case in the majority of real world situations. Even if the length of node identifiers is adjusted based on the number of actual nodes in the network, the identifiers can only possibly be full when the number of nodes is an even power of two. This is a known issue which is dealt with in different ways in the other two existing DHTs based on De Bruijn graphs, Koorde and Broose. [9] Unfortunately, the solution to this issue is outside of the scope of this work due its very technical nature and the length of the explanation. This work is intended to address the high level issues involved in stopping censorship on the Internet. The Arcadia DHT is just one of the many possible DHT designs which can fulfill the Arcadia Conditions for DHTs. Therefore, the initial description of the design will be sufficient for its inclusion in the overall Arcadia design. The details of the particular DHT design and how it works on a sparsely populated network will be dealt with in a future paper dedicated entirely to the design of the Arcadia DHT.

3.5 Obtaining Identifiers

Choosing a good node identifier scheme is the key to creating a DHT which is resistant to IP fishing. The criteria for a good node identifier were previously laid out in the Achord paper, and the criteria will be summarized here, while skipping the supporting evidence. An identifier must be difficult to forge, easy to verify, and it should be difficult to obtain a large number of identifiers. The particular identifier generation scheme should depend on the attack model. For instance, IP addresses are often used as unique identifi-

ers in many security applications. Their use is questionable, however, due to systems like Tor, which allow a user to cycle through a set of IP addresses which are available as proxies. This is the primary reason that Tor users have been banned from Wikipedia, because Wikipedia's security from abuse is implemented via IP blacklists and Tor circumvents this. In fact, one of the goals of Arcadia is to circumvent IP blacklists, so using IP addresses as identifiers would be unwise. Additionally, the Sybil Attack paper rules out identifiers which are verified via a challenge, which requires the expenditure of computational resources. Since challenges do not happen in parallel, the computational resources can be reused over time, allowing identifier generation to be much more efficient than it should be. A similar attack can be used for identifiers which utilize human resources in the identifier challenge, such as those based on image recognition (captchas) or puzzle solving (thinkcash). A dedicated worker with an automated system to collect and present the challenges can simulate a huge number of users, as the per-user effort must be kept down to a relatively small amount in order for the system to have any substantial amount of adoption.

The solution to this dilemma comes in the realization that the important property of an identifier is not that it is expensive to verify, but that it is expensive to obtain. The verification of the identifier should be simple and automatic. However, obtaining an identifier should require a substantial amount of work. Although this was not the conclusion of the Sybil Attack paper, the method of identifier generation which it proposes has this property. The suggested method is to have a trusted authority (Microsoft, in this case, since it was a Microsoft paper) assigns random numbers and provides a digital signature

which proves that the random number was assigned by Microsoft. Microsoft requires some unspecified process (probably a substantial payment) in order to assign such a number. This method has some ideal characteristics. It is easy to verify: just check the digital signature. The method for obtaining an identifier can be made arbitrarily expensive. However, even if it takes substantial effort, it is effort which must only be expended one time for most casual users, whereas it must be expended over and over again, constantly, for attackers. At some point a line must be drawn, as a truly resourceful and determined attacker could always undertake the expensive process of running enough actual nodes to dominate the network. The goal is that by the time the size of the network is large enough for the attacker to take notice, the expense to subvert a network of that size will be too great to bear. The attacker will then try the less expensive method of running a few nodes with quickly changing identifiers. For the system to survive this attack, the cost of generating a sufficient number of identifiers to subvert a network of noticeable size should also be unmanageably expensive.

This system may sound unworkable due to its centralized nature. The identifier generator service must scale to the rate of influx of new nodes. However, this sort of system is not at all unprecedented. The centralized identifier generator is a design used throughout the Internet. Services such as Hotmail, AIM, and Friendster each have their own global namespaces. On each service, a user must go through a process to obtain an identifier from the central service. Even decentralized networks such as Kazaa and Skype still use a centralized identifier service so that they can use global namespaces. However, it is also possible to partition the identifier space into multiple parts and then assign dif-

ferent providers to assign identifiers in that space, as is done with IP addresses and domain names. The problem with allowing multiple providers is that if any provider decides to collude with the attacker (or is in fact an attacker), then it can start assigning non-random addresses to the attacker. Once the attacker can obtain non-random addresses, it functionally has the power to generate arbitrary identifiers by choosing an identifier from the available set that is sufficiently close to its intended target to intercept enough traffic to do damage. The solution, then, is to either have a single generation service which is trustworthy, or to not allow providers to give out non-random addresses. One possible approach to this is to not use the assigned number as the identifier, but rather the hash of the number. Since the identifier is a hash, it is randomly distributed through the identifier space. In order to choose a particular identifier, the attacker would have to find a value within its allocated space which hashes to the identifier that the attacker desires. This is equivalent to breaking the hash function, which is a known difficult problem, and, therefore, can be considered to have a high cost.

The question remains of what the process should be for obtaining an identifier. In order to achieve maximum effectiveness against the given attackers, one possible method of allocation of identifiers is to partition the address space and delegate assignments of identifiers in these partitions to organizations which are publicly known for opposing the agents of censorships. For instance, one partition could be given to an organization that promotes democracy in China. Another could be given to an organization which opposes the RIAA's control of music distribution and supports independent music labels. This system reduces the burden on the central authority, as it only needs to identify trustwor-

thy organizations which oppose censorship. This will mostly likely scale up much better than having a central authority assign all identifiers.

The possible attack on this system is that if the attacker can manage to have an evil organization get a partition allocated to it, it can use a brute force attack of hashing all of the identifiers in the partition in order to produce a dictionary of what real identifiers it can generate. Once this dictionary is computed, it is inexpensive to find the identifier in the dictionary which is closest to the one that the attacker desires. One solution to this problem is to make the identifier space so large that computing a dictionary would take a prohibitively long time. However, in a distributed hashtable, the number of bits which are actually used for routing is proportional to the number of nodes in the network. So there is no point in arbitrarily raising the size of the identifier space past the number of bits which will actually be used. For instance, if only 10 bits are going to be used, then the attacker only needs to continue to hash identifiers until it has a dictionary which contains hashes with every permutation of the first 10 bits. The values of the remaining bits are irrelevant, so the additional bits in the identifier may as well all be equal. Therefore, the only real solution to the problem of dictionary-based attacks is to make the number of actual nodes in the network sufficiently large.

3.6 Generating Identifiers

The node identifier has three necessary properties in order to be secure:

- It must come from an authority and this must be verifiable.

- Other nodes must not be able to copy a node’s identifier.

- The authority must not be able to pick arbitrary identifiers.

The following is one way to generate identifiers which fulfills these conditions. To generate an identifier, the authority generates a random public and private key pair. It then signs the public key and presents the signature to the user along with the key pair. In order to establish communication, a node, Alice, contacts another node, Bob, and sends its public key and the signature from the authority. Bob checks the signature and then sends a random number to Alice. Alice signs it with the private key and then returns the result. Bob checks the signature. Bob then hashes the public key, and the result is Alice's identifier. The first condition is fulfilled by checking the signature from the authority. The second condition is fulfilled by checking the signature on Bob's random number. The third condition is fulfilled by using the hash of the public key instead of the key itself.

3.7 Caching Policies

As addressed in “The Economics of Censorship Resistance”, [10] it is important in the design and adoption of a censorship resistance system to take into account the motivations of people who run nodes in the network. While some of them participate in order to gain anonymity for themselves in their own web browsing activities, a good many of them, as was shown by the contents of the Freenet mailing list, participate in order to support a general idea of the protection of the freedom of speech. However, despite a general agreement that speech should be “free”, there is dissent among users about which

speech should be protected and which should be prohibited. Of particular interest is the case of where participants are in different legal jurisdictions, such that some content might be legal for some to store and illegal for others. Even users who are fervent proponents of freedom of speech are uncomfortable at the idea that they may be storing material which is considered universally evil. The fear of accidentally storing such content will keep many people from running nodes.

The problem of universally evil content is already addressed by making the network only cache information. This way, someone somewhere must originally agree to host the content. If it is truly universally evil, they will be unable to obtain hosting anywhere in the world. However, there is still the issue of legal jurisdictions, such as certain sorts of speech which are illegal in Germany are legal in the United States, which means that it is possible to obtain hosting for content in one country for content which is illegal in another country. Clearly, users should not be allowed to censor such information. Otherwise, our goal of making democratic speech available in China would be invalid. However, we should not force users into storing content which is illegal in their jurisdiction. As the economics of censorship papers points out, one of the motivations of participating in a censorship resistance network is not a universal respect for all speech, but a desire to further your own viewpoints, which are censored otherwise. Such a desire goes hand in hand with not wanting to participate in the distribution of ideas contrary to your own. Thus, in order to increase the motivation of users to actively participate in the resistance of censorship, they should be able to make decisions on how much they want to help in the distribution of specific content. Specifically, they should be able to opt out of storing

certain pieces of information, and, similarly, act as permanent caches for information which they want to guarantee will always be available. Luckily, the results of the research on this matter show that networks which allow nodes to show discretion in what information they store are actually more resistant to censorship than purely randomly allocated storage.

There are some subtle advantages and disadvantages to using a caching proxy system for censorship resistance purposes, addressed in “Using Caching for Browsing Anonymity”. [11] The main issues with using caching is that the owners of websites have issues with their ownership of the website content and what people should be allowed to do with it. The issues they have with caching are issues of control: users can still view content which the website owner may have intentionally deleted; users may be able to view restricted content if it was previously viewed and cached by someone with access (for instance sites which require registration or perhaps pay subscriptions); and the website records fewer hits since many of the hits on the website will be serviced by the proxies instead of the webserver. This last issue is of particular concern when per-hit banner ads are used. Serving the ads out of the cache will cause the number of ad hits reported to be less than the number of actual ad hits, reducing the amount of money that changes hands based on ad hits. Google, who maintains a massive cache of most of the Internet, suggests that websites which don’t want to be cached can simply include the appropriate meta tags in their web pages to tell Google to not cache their content. This is even easier in Arcadia because Arcadia follows normal HTTP caching conventions. Therefore, simply making the website use POST instead of GET or including HTTP headers which in-

dicate that the page should not be cached will be sufficient to keep the content from being accessed anonymously and from cache.

The hidden advantage of using a caching network is that it actually provides considerable protection against intersection attacks. Since the HTTP request is often going to be answered by a random proxy in the chain instead of the originating webserver, there will be no correlation between the requesting user and the website which stores the content, neatly defeating intersection attacks. While intersection attacks were specifically not included in the design criteria for Arcadia because none of the attackers in the attack model use this sort of attack, the design nevertheless inherently provides some protection.

3.8 Proxy Discovery

The key to success of a censorship resistance proxy network, as addressed in "Thwarting Web Censorship with Untrusted Messenger Discovery", [12] is the proxy discovery problem. The clients must keep an updated list of proxies as the proxies are continually added, removed, and blocked by the attackers. However, the attacker must be unable to discover all of the proxies, either by in-band or out-of-band means. The following characteristics are necessary for a solution to the proxy discovery problem if it is to be resistant to attack: the system should have a large number of proxies, clients must discover proxies independently of each other, the client must incur some cost to discover a proxy, and brute-force scanning techniques must not expose proxies. The method proposed by Feamster is called keyspace hopping, and is similar in a number of ways to the Arcadia technique. The main design criteria that should be taken from the keyspace hop-

ping approach is that out-of-band discovery is a logical next phase of attack if in-band attack proves to be too expensive. Therefore, out-of-band attacks should be limited as well. Limiting out-of-band attacks in Arcadia is mostly already taken care of, as nodes will not communicate with nodes who are not their neighbors. Therefore, in order to check to see if a given server is an Arcadia node, you must be their neighbor in the network, in which case it is actually an in-band attack. The one exception is bootstrapping. A node must always be able to ask another node for its set of neighbors or else new nodes would be unable to join the network unless they already knew who their neighbors were, which is impossible if they are assigned a random identifier. However, bootstrapping is not entirely out-of-band the way that brute force scanning of IP addresses is. It is assumed that new nodes get the address of a bootstrap node along with their invitation to the network. Otherwise, they would be unable to find a node with which to bootstrap into the network. So, when a node is invited to the network is should be given both the address of the node and its identifier. If a connecting node does not already know the identifier of the node that it is connecting to, then communication should be refused. This will make out-of-band scanning impossible, as the attacker will need to know the identifiers of all of the nodes it is attempting to contact.

4 CONCLUSION AND FUTURE WORK

> *We will hold these discourses in productive tension without allowing them to collapse into univocal accounts.*
>
> Sandy Stone, *Teaching the Unnamable Discourse*

Arcadia is a set of blueprints for building a network which can circumvent existing censorship imposed on websites and defend against the attacks that have been used against existing censorship circumvention systems. Additionally, it provides some protection from a number of theoretical attacks which are often discussed in the literature. Besides giving general guidelines for designing such a system, a full system is described so that implementation can begin immediately. Using these blueprints, a number of independent systems can be implemented to circumvent censorship. With this information readily available to the public, the current methods of censorship cannot continue to function. New methods will need to be developed to impose censorship, as well as new methods to defend against these attacks. In the meantime, the principles of the Arcadia system need to be implemented and deployed to help real users circumvent existing censorship. This initial implementation will serve as a fertile ground to research dissident communities and how communication works when barriers to free speech are removed.

A very interesting area of research is the way that media is changing as the barriers to communication are removed. The Internet provides a more decentralized means of distribution than traditional means such as television and print. With the rise of blogging

and podcasting, traditional media's power is being diluted by the new decentralized Internet media. Censorship of the Internet can be viewed as the old media attempting to assert control over this new medium. With the fear of censorship removed, the new media is free to fully embrace its decentralized nature and reach its full potential. Coupled with the increasing ubiquity of Internet connectivity and mobile computing, a new media revolution is brewing.

BIBLIOGRAPHY

[1] Danezis, George. Better Anonymous Communications [dissertation]. Cambridge (England): University of Cambridge; 2003 Dec. 215 p. Available From: http://citeseer.csail.mit.edu/danezis04better.html

[2] Dingledine R, Mathewson N, Syverson P. Tor: The Second-Generation Onion Router. Proceedings of the thirteenth USENIX Security Symposium [Internet]; 2004 Aug; [cited Dec 8]. p. 17. Available From: http://citeseer.csail.mit.edu/dingledine04tor.html

[3] Wiley B, Sandberg O, Hong TW, Clarke I. Freenet: A Distributed Anonymous Information Storage and Retrieval System. In: Fedarrath, Hannes, editor. Proceddings of the ICSI workshop on Design Issues in Anonymity and Unobservability [Internet]; 2000 Jul 25-26; Berkeley, CA. Berkley (CA): International Computer Science Institute; [cited 2005 Dec 8]. p. 21. Available From: http://citeseer.csail.mit.edu/clarke00freenet.html

[4] Balakrishnan H, Stoica I, Morris R, Karger D, Kaashoek MF. Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications [Internet]. Cambridge (MA): Massachusetts Institute of Technology, 2001 March [cited 2005 Dec 8]. 11 p. Available From: http://citeseer.csail.mit.edu/658598.html

[5] Douceur, John. The Sybil Attack. In: Druschel, Kaashoek, Rowstron, editors. Proceedings of the first annual international workshop on Peer-to-Peer Systems [Internet]; 2002 Mar; Cambridge, MA. Cambridge (MA): MIT Faculty Club; [cited 2005 Dec 8]. p. 6. Available From: http://citeseer.ist.psu.edu/512516.html

[6] Serjantov, A. Anonymizing Censorship Resistant Systems. In: Druschel, Kaashoek, Rowstron, editors. Proceedings of the first annual international workshop on Peer-to-Peer Systems [Internet]; 2002 Mar; Cambridge, MA. Cambridge (MA): MIT Faculty Club; [cited 2005 Dec 8]. p. 6. Available From: http://citeseer.csail.mit.edu/serjantov02anonymizing.html

[7] Kaashoek MF, Karger DR. Koorde: A Simple Degree-Optimal Distributed Hashtable. Proceedings of the second International Peer-to-Peer Symposium [Internet]; 2003 Feb; Berkeley, CA.; [cited 2005 Dec 8]. p. 6. Available From: http://citeseer.csail.mit.edu/kaashoek03koorde.html

[8] Wiley, B and Hazel, S. Achord: A Variant of the Chord Lookup Service for Use in Censorship-Resistant Peer-to-Peer Publishing Systems. In: Druschel, Kaashoek, Rowstron, editors. Proceedings of the first annual international workshop on Peer-to-Peer Systems [Internet]; 2002 Mar; Cambridge, MA. Cambridge (MA): MIT Faculty Club; [cited 2005 Dec 8]. p. 5. Available From: http://citeseer.csail.mit.edu/hazel02achord.html

[9] Gai, A.T. and Viennot, L. Broose: A Practical Distributed Hashtable Based on the De-Bruijn Topology. France: INRIA; 2004 June [cited 2005 Dec 8]. 8 p. Available From: http://citeseer.csail.mit.edu/gai04broose.html

[10] Danezis G, Anderson R. The Economics of Censorship Resistance [Internet]. Cambridge (UK): University of Cambridge; 2004 [cited 2005 Dec 8]. 12 p. Available From: http://citeseer.ist.psu.edu/672740.html

[11] Shubina A, Smith S. Using caching for browsing anonymity. ACM SIGecom Exchanges [Internet]. 2003; [cited 2005 Dec 8]. p. 20. Available From http://citeseer.ist.psu.edu/653646.html

[12] Feamster N, Balazinska M, Wang W, Balakrishnan H, Karger D. Thwarting Web Censorship With Untrusted Messenger Discovery. Proceedings of the third workshop on Privacy Enhancing Technologies [Internet]; 2003 Mar; [cited 2005 Dec 8]. p. 16. Available From: http://citeseer.csail.mit.edu/feamster03thwarting.html

[13] Diaz C, Seys S, Claessens J, Preneel B. Towards Measuring Anonymity. Proceedings of the workshop on Privacy Enhancing Technologies [Internet]; 2002; [cited 2005 Dec 8]. p. 15. Available From: http://citeseer.csail.mit.edu/diaz02towards.html

VITA

Brandon Keith Wiley was born in Austin, Texas on October 26, 1978, the son of Mary Naomi Bashara and Norman Keith Wiley. After completing his work at The Texas Academy of Leadership in the Humanities, Beaumont, Texas, in 1997, he entered The University of Texas at Austin. He received a degree of Bachelor of Arts in Plan II and Bachelor of Science in Radio-Television-Film in May, 2001. During the following years he was employed as a software engineer. In September, 2003, he entered the Media Studies Master's program at The University of Texas at Austin.

Permanent Address: 15904 Arla Cove

Austin, TX 78717

This thesis was typed by the author.