Censorship Resistance Revisited*

Ginger Perng, Michael K. Reiter, and Chenxi Wang

Carnegie Mellon University, Pittsburgh PA, USA

Abstract. "Censorship resistant" systems attempt to prevent censors from imposing a particular distribution of content across a system. In this paper, we introduce a variation of censorship resistance (CR) that is resistant to selective filtering even by a censor who is able to inspect (but not alter) the internal contents and computations of each data server, excluding only the server's private signature key. This models a service provided by operators who do not hide their identities from censors. Even with such a strong adversarial model, our definition states that CR is only achieved if the censor must disable the entire system to filter selected content. We show that existing censorship resistant systems fail to meet this definition; that Private Information Retrieval (PIR) is necessary, though not sufficient, to achieve our definition of CR; and that CR is achieved through a modification of PIR for which known implementations exist.

1 Introduction

Digital censorship resistance, as defined by Danezis and Anderson [11], is the ability to prevent a third-party from imposing a particular distribution of documents across a system. Following the original work of Eternity service [2], a number of proposals have appeared in the literature to implement censorship resistant information services, including Tangler [27], Freenet [9], and Freehaven [12]. However, the term "censorship resistance" has never been formally defined. The most common definition is a variation of "Our system should make it extremely difficult for a third party to make changes to or force the deletion of published materials" [28]. What is meant by "extremely difficult" is subject to interpretation. As a result, it is challenging to precisely evaluate and compare the effectiveness of the various proposals.

In this paper, we present one possible formal definition of *censorship susceptibility*, the dual of censorship resistance. Informally, we define censorship susceptibility as the likelihood a third-party can restrict a targeted document while allowing at least one other document to be retrieved. Both our definition, and the threat model that it permits, refine various prior discussions of censorship resistance. First, while some prior works have included availability of the service as a necessary condition for censorship resistance (e.g., [2]), our definition decouples these notions (as does, e.g., Dagster [24]). As defined here, censorship susceptibility measures the extent to which an adversary can prevent *selected* content from being distributed. A service with low censorship susceptibility thus leaves the adversary only the option of completely shutting down the

* This research was supported in part by National Science Foundation Grant No. 0208853 and the NSF Graduate Research Fellowship.

M. Barni et al. (Eds.): IH 2005, LNCS 3727, pp. 62–76, 2005.

service. While high availability, in a technical sense, is one approach to preventing this, it is not the only one; others include defeating efforts to shut down the service in court[1] and extreme social opposition[2]. As such, we find it useful to decouple the notions of censorship resistance and high availability.

A second distinction of our work is that the threat model we adopt grants the adversary more capabilities than prior works, and in this sense is conservative. First, we permit the adversary to identify the server(s); unlike many prior works in censorship resistance, we do not employ anonymization mechanisms, or, if they are employed, we presume that the adversary can break them (e.g., [17,25,29]) and discover the servers. Second, we permit the adversary to inspect every request to and response from the server, and even a transcript of the server's processing steps on the request, in order to reach a decision as to whether to filter (drop) the response. The *only* secret that the server is permitted to keep from the adversary is a digital signing key, which we argue is a necessity to keep the adversary from simply impersonating and effectively replacing the service to all clients. The only other limitation is that the adversary is not permitted to modify the server's contents. We are primarily interested in censorship resistance for documents which are legal to possess (and so it is not lawful for the adversary to remove them), but the distribution of which may be restricted; Church of Scientology scriptures and other lawfully purchased but copyrighted works are but two examples.

In this context, we conduct a foundational study of censorship susceptibility in an effort to relate it to known cryptographic primitives. Our primary results are as follows:

- Censorship Resistance (CR) implies Private Information Retrieval (PIR) [10,8,6]. That is, in order to implement CR (or more specifically, low censorship susceptibility), it is necessary to implement PIR.
- PIR does not imply CR. That is, not any implementation of PIR satisfies our definition of CR.
- CR can be achieved through a simple modification of PIR using digital signatures.

PIR is a cryptographic primitive that allows a user to query a database without revealing the index of the queried item. Our study shows that censorship resistance, as defined here, cannot be achieved with primitives weaker than PIR. Additionally, we show that PIR achieves a set of properties that are not sufficient to implement CR services, but that are sufficient when combined with digital signatures.

The rest of the paper is structured as follows: in Section 2, we discuss related work, in Section 3 we describe the system model and give a formal definition of censorship susceptibility. We analyze existing CR approaches using the definition in Section 4. We show a reduction from general CR to PIR in Section 5 and prove that PIR is not sufficient for CR in Section 6. In Section 7 we describe a system that implements CR. We conclude in Section 8.

[1] An example is the April 2003 U.S. federal court decision that file-sharing services Grokster and StreamCast networks were not culpable for illegal file trading over their networks, citing substantial uses in addition to copyright-infringing ones.

[2] The Chinese media control department, in 2001, reversed its prior decision to censor a Taiwanese TV series due to extreme popular demand.

2 Related Work

Our model of censorship resistance is similar to the all-or-nothing integrity (AONI) model proposed by Aspnes *et al* [3]. AONI is the notion of *document dependency* where the corruption of one document leads to the corruption of other documentsin the system. Their definition leads to an analysis of classes of adversaries in which AONI is not possible. Ours is a different formulation that more readily allows a reduction of CR to other cryptographic primitives (e.g., PIR). This allows us to make precise statements about the hardness of the problem and delineate the necessary cryptographic primitives to build a system that is indeed CR.

Danezis and Anderson proposed an economic model to evaluate censorship resistance of peer-to-peer systems [11]. They analyzed the effect of resource distribution schemes (random vs. discretionary) on the system's ability to resist censorship. Their model focuses on the economic values of censoring and anti-censoring. While the findings in [11] are interesting and can help shape future designs of censorship resistant systems, the goal of our study is somewhat different. We strive to understand the fundamental relationships between censorship resistance (as per the assumptions of our model) and other known cryptographic primitives as to permit the derivation of a formal security argument.

There exist numerous implementations of censorship resistant services [2,4,5,9,12,26,24,28,27]. We defer our discussion of those schemes to Section 4.

3 System Model and Definitions

In this section, we describe a formal definition of censorship susceptibility. We begin by describing the model of an information system to frame our discussion.

3.1 System Components and Interface

An information system, sys, includes the following components:

1. **Server:** A server is an entity that stores and responds to requests for (perhaps in conjunction with other servers) data. The server uses a signing oracle that encapsulates it private signature key.
2. **Document store:** A document store ds is a collection of servers that jointly provides information services. ds has a public key pk_{ds} such that pk_{ds} is the union of all servers' public keys.
3. **Client:** A client c is a process that queries the document store for documents.

Note that the private signature key is used solely for authentication. We discuss the ramifications of the private signature key in Section 3.2.

We now describe the system interface for information retrieval. We assume that there is a finite and static set of documents, DOC, in the system and that each document, $doc \in DOC$, is associated with some descriptive $name$ that uniquely identifies doc. The format of $name$ and the mapping between $name$ and doc is left intentionally

undefined. We further assume a client, c executes an information retrieval function get such that

$$doc \leftarrow \mathsf{get}^{ds}(pk_{ds}, name)$$

to retrieve the document doc identified by $name$ from ds. We assume that ds signs all responses to queries, and as part of the get function, $\mathsf{get}(\cdot)$ verifies the response from ds with public key pk_{ds}.

3.2 Adversarial Model

A censorship resistance adversary A_{CR} wishes to remove a targeted document. A_{CR} is composed of two algorithms:

1. **Generator:** A generator G is an algorithm that outputs the $name$ of a targeted document and some state s to send to the filter.
2. **Filter:** A filter $f(\cdot)$ is an algorithm imposed upon a document store ds. $f(\cdot)$ takes in state s from G as input. $f(\cdot)$ may intercept, modify, or drop queries for and responses from ds. $f(\cdot)$ is allowed to examine the internal transcripts of the servers in the document store, but it cannot modify the state of the servers or the client side algorithm.

Note that $f(\cdot)$ models our adversary of interest. We assume that $f(\cdot)$ runs on each information server and the servers readily subject themselves to the filter's inspection. We assume that filters have full access to the server's communication and work logs. Modeling the server and the adversary in such a manner allows the conceptual formulation of a law-compliant server that willingly turns over all internal states to a third party censor (a filter). The only exception is the server's private signing key; the signing operation is modeled by a signature oracle whose internal state is hidden from outside view. We note that allowing the server to keep its signature key private is a reasonable assumption as it is used solely as an authentication mechanism. Furthermore, the disclosure of the signature key would allow the adversary to impersonate the server to clients using the service, thus defeating any other mechanisms to resist censorship.

We now discuss the changes in the $\mathsf{get}(\cdot)$ protocol when an adversarial filter $f(\cdot)$ is placed upon a document store ds. We define ds' as ds installed with the filter, $f(\cdot)$. More specifically, $ds' = f(\cdot)^{ds}$ where $f(\cdot)$ uses ds as an oracle to answer queries. The $\mathsf{get}(\cdot)$ function between client c and ds' is modified as follows:

1. $\mathsf{get}^{ds'}(pk_{ds}, name)$ constructs a query, $\mathsf{q}(name)$ and sends the query to ds'.
2. ds' receives the query from c. If the query is passed onto ds, ds performs some computations in response to the query (e.g, searching for the document, signing the response using its oracle, etc.), all of which are logged and viewable by $f(\cdot)$. ds' may or may not respond to $\mathsf{get}(\cdot)$.
3. If $\mathsf{get}(\cdot)$ receives an answer, it authenticates the response using pk_{ds} and performs the necessary client-side computation before outputting the document to c.

3.3 Definition

Informally, we define censorship susceptibility as the probability that an adversary can block a targeted document while allowing at least one other document to be retrieved. A system has low censorship susceptibility if the maximum *advantage* of any censorship resistance (CR) adversary is small. More formally:

Definition 1. *Let $A_{\mathrm{CR}} = \langle G, f(\cdot)\rangle$ be a CR adversary. Let $Name$ be the set of descriptive names that retrieve documents in ds, and $\perp$ be defined as an undefined or incorrect result. $Name$ and pk_{ds} are public knowledge. A_{CR}'s advantage is:*

$$\mathbf{Adv}^{\mathrm{CR}}_{sys}(A_{\mathrm{CR}}) = \Pr[\perp \leftarrow \mathsf{get}^{ds'}(pk_{ds}, name) \mid (name, s) \leftarrow G^{ds};\ ds' \leftarrow f(s)^{ds}] \ - \\ \min_{name' \in Name}[\Pr[\perp \leftarrow \mathsf{get}^{ds'}(pk_{ds}, name') \mid (name, s) \leftarrow G^{ds};\ ds' \leftarrow f(s)^{ds}]]$$

The censorship susceptibility of a system is thus defined as:

$$\mathbf{Adv}^{\mathrm{CR}}_{sys}(t, q) \stackrel{\mathrm{def}}{=} \max_{A_{\mathrm{CR}}}\{\mathbf{Adv}^{\mathrm{CR}}_{sys}(A_{\mathrm{CR}})\}$$

where the maximum is taken over all adversaries that run in time t and make at most q queries to its oracle, ds.

This definition states that for a system to be CR, there cannot exist an adversary who can block a certain document while allowing another to be retrieved. More specifically, generator G outputs the $name$ of the doc it wishes to censor and passes s to $f(\cdot)$. Given a system sys with its document store installed with filter $f(s)$, sys's censorship susceptibility is based on the probability that the filter successfully blocks $name$ while not blocking some $name'$. If sys has low censorship susceptibility, then the adversary's only viable option is to institute a complete shut down of the service.

For the remainder of this paper, we omit specifications of t and q, as these values will be clear from context.

4 Analysis of Current CR Schemes

In this section we analyze current implementations of CR schemes based on the model described in Section 3. We briefly describe each mechanism first and then analyze its capability to resist censorship. Some of the interface descriptions are abbreviated as we only describe the parts that are integral to the discussion. In the discussions that follow, we loosely categorize the different CR proposals into four categories: data replication, anonymous communication, server deniability, and data entanglement.

4.1 Data Replication

Eternity Service: Eternity Service [2] provides censorship resistance through anonymous communication and data replication across multiple jurisdictions. The underlying premise is that a universal injunction across all jurisdictions is unlikely.

Freenet: Freenet [9] consists of volunteer servers that provide a document store. A document in Freenet is encrypted with a descriptive name as its key and requested using

Freenet's get($name$) interface where $name$ is its content key. Queries are forwarded to servers hosting names which offer the closest match to $name$. If the document is found, the server reverse-routes doc back to the client, and each server on the return route caches a copy of the document.

Gnutella: Gnutella [26] also uses data replication to resist censorship. Data replication in Gnutella is achieved primarily in a discretionary manner—data is replicated to users who request the data.

These schemes rely on replicating data either arbitrarily across the system or in some structured way (as in Freenet) to thwart a censor authority in its effort to locate and destroy all copies. While data replication will likely increase the cost of censorship [11], it is insufficient against the attacker model we described in Section 3.

Recall that we permit an adversarial filter to observe and inspect communications and internal processings of each information server. In effect, one can view the filters as the result of a universal injunction; thus, systems that rely solely on replication are not sufficient. In the case of Freenet, documents are referenced by a publicly known $name$. The filter can simply block any query with a particular $name$, thereby achieving censorship. We can calculate the censorship susceptibility of Freenet using the definition in Section 3.3; as each $name$ is uniquely mapped to a single document, the censorship susceptibility of Freenet by our definition is equal to 1. The censorship susceptibility of Gnutella and Eternity Service is similar.

In addition, censorship against both Freenet and Gnutella can be achieved without significant effort on the part of the censor. In Freenet, documents are only replicated along the retrieval route; if a document is not particularly popular, then its number of replicas will be limited. In peer-to-peer systems such as Gnutella that rely solely on discretionary replication, it has been shown that the system degenerates into a client-server model where most documents are replicated only on a subset of the peers [1]. As such, the cost of censorship can be significantly less than what is required to procure a universal injunction.

4.2 Anonymous Communication Systems

Many CR implementations rely on anonymous communication channels [7,13,20,21]. We discuss two such systems here.

Free Haven: Free Haven [12] is a peer-to-peer network that provides anonymity and document persistence. In Free Haven, each stored document is divided into shares that are signed with the document's private key. The shares are stored on a server along with the hash of the corresponding public key. Clients retrieve documents with a get($name$) interface where $name$ is the hash of the document's public key. A request is broadcast to the entire network; servers holding the matching key hash respond with the stored shares. The client recreates the file upon receiving a sufficient number of shares.

Anonymizing Censorship Resistant Systems: Serjantov [22] describes a peer-to-peer system that provides censorship resistance using Onion Routing [14]. Each peer in the system can act as a server, forwarder, or decrypter. Each stored document is divided into encrypted blocks and placed on multiple servers. A forwarder acts as an intermediary between the servers and a client; only a forwarder knows the mapping between the

data blocks and the servers that store them. A decrypter is responsible for decrypting data blocks but does not have knowledge of the data-server mapping. In this system, $name$ in the get($name$) function encodes the set of forwarders and the labels of the encrypted blocks. All communications in this system are carried out on top of anonymous channels.

Both Free Haven and Serjantov's system rely on anonymous communication channels (e.g, [14], [18]) to resist censorship. Free Haven additionally migrates document shares periodically among servers to offer another level of protection [12]. Serjantov's system also uses data encryption to protect servers; the intuition is that if servers cannot decrypt the blocks they store, they cannot be prosecuted for storing certain data.

Unfortunately, in Free Haven one can easily identify the target document in a query using well known public keys. In Serjantov's system, the identities of the forwarders are in the public $name$ of the document. A third party filter can easily deny queries associated with a particular key or in the latter case, simply deny queries to the particular forwarders found in $name$, thus achieving selective filtering. Similar to prior discussions, the censorship susceptibility of both systems is thus equal to 1.

We note that attacks against anonymous communication channels have been demonstrated (e.g., [17,19,25,29]). It is feasible that a censor authority could possess the resources to undermine current anonymous communication technology and discover the locations of the servers in the system.

4.3 Server Deniability

Publius: Publius [28] consists of a static set of servers hosting encrypted documents. The encrypted documents are stored onto multiple servers, each with a share of the document key. Because servers do not store the entire key for a particular document, and documents are stored encrypted, Publius suggests that it achieves *server deniability*, the ability for a server to deny knowledge of the hosted documents' contents. To retrieve doc, a Publius client use the get($name$) function, where $name$ is the Publius URL that encodes the hosting servers and the target document. A subset of those servers respond with the encrypted document and their key shares, the latter of which the client uses to reconstruct the key to decrypt the document.

Publius claims to be censorship resistant because servers cannot determine the content of its storage. However, since the Publius $name$ (e.g., the URL) for a document is well known, a censor authority can easily locate the servers and filter the requests associated with a particular $name$.

4.4 Data Entanglement

Tangler: Tangler [27] is a network of servers that provide data storage. Censorship resistance is provided by "entangling" documents such that the removal of one document will result in the removal of other documents. In Tangler, each document is divided into blocks, each of which is *entangled* (using Shamir secret sharing [23]) with two arbitrary blocks in the system. Each entanglement creates two new blocks in addition to the two existing ones. A threshold number of entangled blocks reconstruct the original block. To retrieve a document, a client uses the get($name$) function where $name$ identifies the

necessary blocks to reconstruct the document. Tangler servers periodically move blocks from server to server using consistent hashing [15].

Dagster: Dagster [24] is a single-server system with a similar goal to Tangler: prevent censorship resistance by "entangling" different documents. In Dagster, data is entangled in such a manner that the removal of a document results in the unavailability of the documents entangled with it.

Both Tangler and Dagster use data entanglement to offer protection against censorship. However, because only a limited number of blocks are used in a particular entanglement, an adversary can institute the removal of a document without visibly affecting the overall availability of the remaining documents. Per our definition in Section 3.3, the system censorship susceptibility is equal to 1.

4.5 Discussion

We note that our threat model grants the adversary more capabilities than prior work in that we allow the adversary power to impose universal policies across the system and capability to inspect internal server states. In situations where these assumptions do not hold, that is, servers do not voluntarily cooperate or jurisdiction boundaries prevent universal censor policies, the mechanisms analyzed in this section would offer certain protection against censorship. In those cases, the economic model [11] can be used to reason about their abilities to resist censorship.

5 CR Implies PIR

In this section, we show that CR implies Private Information Retrieval (PIR). This implication states that CR systems cannot be constructed with primitives weaker than PIR. We first introduce the PIR primitive. We then prove that CR implies PIR.

5.1 Preliminaries

Private Information Retrieval: A private information retrieval (PIR) scheme [6,8,10,16] is an interactive protocol between two entities: a database, DB, and a user, U. The goal of a PIR protocol is to allow U to query DB without revealing to DB the index of the queried item. DB holds a n-bit string $x \in \{1,0\}^n$ which is indexed by $i \in \{1,...,n\}$. U construct queries, $q(i)$, to retrieve the i-th bit, x_i, from DB. At the end of the protocol, two properties must hold:

1. Correctness: U has the correct value for x_i, and
2. User Privacy: DB has no knowledge of the retrieved index, i.

The user privacy of a PIR may be modeled as the advantage a PIR adversary has against the scheme. More formally, the advantage of a PIR adversary is defined as:

Definition 2. *The advantage an adversary, A_{PIR} has against a PIR protocol,* P*, is defined as:*

$$\mathbf{Adv}_{\mathrm{P}}^{\mathrm{PIR}}(A_{\mathrm{PIR}}) = \max_{i,j}[\Pr[A_{\mathrm{PIR}}(q(i)) = 1] - \Pr[A_{\mathrm{PIR}}(q(j)) = 1]]$$

where $i, j \in \{1 \ldots n\}$ are indices into the database. The advantage of the PIR protocol is thus defined as:

$$\mathbf{Adv}_{\mathrm{P}}^{\mathrm{PIR}}(t) \stackrel{\mathrm{def}}{=} \max_{A_{\mathrm{PIR}}}[\mathbf{Adv}_{\mathrm{P}}^{\mathrm{PIR}}(A_{\mathrm{PIR}})]$$

where the maximum is taken over all adversaries that run in time t.

In the remainder of this paper, we omit the specifications of t, as its value will be clear from context.

Informally, the user privacy of a PIR protocol is modeled as the probability that an adversary can distinguish between queries for two different indices.

In the discussion below, we assume a PIR protocol similar to those in [8,10]. These protocols retrieve a block of bits per query, which permits us to draw a parallel between a document from a CR system and a block of bits from a PIR database.

5.2 CR Implies PIR

In this section, we construct a PIR protocol from a generic CR system. We define a PIR protocol built on top of a secure CR system, sys, as PIR-sys. We show that if there exists a PIR adversary against PIR-sys with significant advantage, then there exists a CR adversary against sys with significant advantage. More formally:

Theorem 1. $\forall A_{\mathrm{PIR}}, \exists\, A_{\mathrm{CR}} \;:\; \mathbf{Adv}_{sys}^{\mathrm{CR}}(A_{\mathrm{CR}}) \geq \mathbf{Adv}_{\text{PIR-}sys}^{\mathrm{PIR}}(A_{\mathrm{PIR}})$.

Proof. We prove the above theorem by constructing the PIR protocol, PIR-sys, from a generic CR system, sys.

A PIR protocol has two parties: a user U, and a database DB. To create the PIR protocol, we map the CR document store (ds) to DB, and allow U access to all of the functions available to the CR client. The set of retrievable documents in a CR system can be indexed from 1 to n, where $n = |DOC|$. We view this enumerated set of documents as the PIR data-string held by DB.

Recall from Section 3.1 that documents in the CR system are retrieved using descriptive names. Because a descriptive name uniquely identifies a document, we can map the set of descriptive names associated with a document to an index in the PIR database. A PIR query for index i, $q(i)$, simply becomes a combination of a lookup function to find a descriptive name that corresponds to the document at i and a call to the CR query function with the given descriptive name. The protocol for the PIR protocol is as follows:

1. U wishes to retrieve the document indexed at i from DB. He calls the mapping function, $\mathsf{map}(i)$, to get a descriptive name, $name$, corresponding to the document.
2. U uses $name$ to outputs a query, $\mathsf{q}(name)$, using the query function of the CR system. Specifically, U performs $\mathsf{get}^{ds}(pk_{ds}, name)$.
3. Upon receiving the query, ds performs its normal computations to return a response to U.
4. U performs the CR computations to reconstruct the document from the response.

We denote the described protocol as PIR-sys. We now prove that the two properties of PIR, correctness and user privacy, indeed hold for the resulting PIR protocol.

Correctness: Follows directly from the CR system. The PIR query is essentially a CR query.

User Privacy: To prove that our PIR scheme satisfies the property of user privacy, we show that the existence of a PIR adversary, A_{PIR}, implies the existence of a CR adversary, A_{CR} with at least the same advantage. Let us assume that there indeed exists an A_{PIR} that distinguishes between two indices with advantage, $\mathbf{Adv}^{\mathrm{PIR}}_{\text{PIR-}sys}(A_{\mathrm{PIR}})$. From A_{PIR}, we can build a CR adversary A_{CR}, whose advantage $\mathbf{Adv}^{\mathrm{CR}}_{sys}(A_{\mathrm{CR}})$ is at least that of A_{PIR}. Recall that $A_{\mathrm{CR}} = \langle G, f(\cdot)\rangle$. The CR adversary works as follows:

1. A_{PIR} has two indices, i and j, where the PIR advantage is maximized. Namely,
$$\mathbf{Adv}^{\mathrm{PIR}}_{\text{PIR-}sys}(A_{\mathrm{PIR}}) = \Pr[A_{\mathrm{PIR}}(q(i)) = 1] - \Pr[A_{\mathrm{PIR}}(q(j)) = 1] \quad (1)$$
2. G designates i to be the target document doc and uses $\mathsf{map}(i)$ to find $name$ such that $doc \leftarrow \mathsf{get}^{ds}(pk_{ds}, name)$.
3. G creates some state s to pass to $f(\cdot)$. For every query, $f(s)$ will function as follows:
   (a) $f(s)$ sends the received query q to A_{PIR}.
   (b) If A_{PIR} returns 1, $f(s)$ denies the query. Otherwise, $f(s)$ passes the query to ds.
4. G outputs $(name, s)$.

Recall from Section 3 that the goal of the CR adversary is to block a targeted document while allowing at least one other document to be retrieved. The CR adversary's advantage is calculated as:

$$\mathbf{Adv}^{\mathrm{CR}}_{sys}(A_{\mathrm{CR}}) = \Pr[\perp \leftarrow \mathsf{get}^{ds'}(pk_{ds}, name) \mid (name, s) \leftarrow G^{ds};\ ds' \leftarrow f(s)^{ds}] - \min_{name' \in Name}[\Pr[\perp \leftarrow \mathsf{get}^{ds'}(pk_{ds}, name') \mid (name, s) \leftarrow G^{ds};\ ds' \leftarrow f(s)^{ds}]]$$

We label the two events in this advantage calculation as "A_{CR} block targeted" and "A_{CR} block not targeted" respectively. From our construction of PIR from CR, we see that

$$\Pr[A_{\mathrm{CR}} \text{ block targeted}] = \Pr[A_{\mathrm{PIR}}(q(i)) = 1] \quad (2)$$
$$\min_{name' \in Name}[\Pr[A_{\mathrm{CR}} \text{ block not targeted}]] \leq \Pr[A_{\mathrm{PIR}}(q(j)) = 1] \quad (3)$$

Thus, we have the following reduction:

$$\begin{aligned}\mathbf{Adv}^{\mathrm{CR}}_{sys}(A_{\mathrm{CR}}) &= \Pr[A_{\mathrm{CR}} \text{ block targeted}] - \min_{name' \in Name}[\Pr[A_{\mathrm{CR}} \text{ block not targeted}]] \\ &\geq \Pr[A_{\mathrm{PIR}}(q(i)) = 1] - \Pr[A_{\mathrm{PIR}}(q(j)) = 1] \\ &\geq \mathbf{Adv}^{\mathrm{PIR}}_{\text{PIR-}sys}(A_{\mathrm{PIR}}) \quad (4)\end{aligned}$$

This proof shows that CR implies PIR. Since the CR system sys is secure, and the PIR adversary's advantage is bound from above by the advantage of the CR advantage against sys, the PIR protocol, PIR-sys is secure. Thus, this theorem states that CR implies PIR. Consequently, CR systems cannot be built with primitives weaker than PIR.

6 PIR Does Not Implement CR

In this section, we sketch a proof of why PIR does not implement CR. In this proof, we use a specific implementation of computational PIR based on the Quadratic Residuosity Assumption [16]. We first describe the PIR implementation; we then show that using this PIR implementation trivially as the CR mechanism results in high censorship susceptibility.

6.1 PIR Scheme

The parties in the PIR protocol from [16] are identical to the generic PIR protocol described in Section 5.1. However, in this implementation, we view the database as a $s \times t$ matrix of bits (denoted by M). The target document is thus $M_{(a,b)}$, where a and b are the row and column indices, respectively. The following notation is used in describing the protocol:

- N is a natural number.
- $\mathbb{Z}_N^* \equiv \{x \mid 1 \leq x \leq N, \gcd(x, N) = 1\}$
- $H_k \equiv \{N | N = p_1 \cdot p_2$ where p_1, p_2 are $k/2$-bit primes$\}$
- $Q_N(y)$ denotes the quadratic residuosity predicate such that $Q_N(y) = 0$ if $\exists w \in \mathbb{Z}_N^* : w^2 = y \mod N$ and $Q_N(y) = 1$ otherwise.
- y is a QNR if $Q_N(y) = 1$, otherwise y is a QR.
- $J_N(y)$ denotes the Jacobi symbol of $y \mod N$. Note that if $J_N(y) = -1$, then y is QNR, and if $J_N(y) = 1$, then y can be QNR or QR.
- $\mathbb{Z}_N^1 \equiv \{y \in \mathbb{Z}_N^* | J_N(y) = 1\}$

The PIR protocol for U interested in $M_{(a,b)}$ is as follows:

1. U begins by picking a random k-bit number $N \in H_k$.
2. U selects t random values $y_1, ...y_t \in \mathbb{Z}_N^1$ such that y_b is a QNR and y_j is a QR for $j \neq b$. U sends $(y_1, ..., y_t, N)$ to DB and keeps N's factorization secret.
3. DB computes, for every row r, a number $z_r \in \mathbb{Z}_N^*$, as follows: It first computes $w_{r,j}$ such that $w_{r,j} = y_j^2$ if $M_{r,j} = 0$, and $w_{r,j} = y_j$ otherwise. It then computes $z_r = \prod_{j=1}^{t} w_{r,j}$.
4. DB sends $z_1, ..., z_s$ to U.
5. U only considers z_a. Because $Q_N(xy) = Q_N(x) \oplus Q_N(y)$, z_a is QR iff $M_{(a,b)} = 0$. Since U knows the factorization of N, she can check whether z_a is a QR and thus retrieve bit $M_{(a,b)}$.

This protocol has been shown to be a secure PIR protocol under the Quadratic Residuosity Assumption[16]. In the discussion below, we denote this PIR protocol as QRA.

6.2 PIR Does Not Implement CR

We now show how the PIR scheme, QRA, does not implement CR. Consider a system that is a straightforward implementation of the PIR protocol, QRA. Namely, interpreting the PIR protocol as the CR system: the CR document store consists of one-bit documents, the "descriptive names" of the documents are the indices into the data string, and the CR get$(\cdot)$ function is the PIR query function. Let n be the size of the PIR database. We have the following theorem:

Theorem 2. $\exists\, DB\ :\ \mathbf{Adv}^{\mathrm{CR}}_{\mathrm{QRA}}(A_{\mathrm{CR}}) = 1$

Proof. Assume that the CR adversary wishes to censor the document, $M_{(a,b)}$. As described in Section 3, the filter, $f(\cdot)$, may modify the query, but it may not change the retrieval algorithm of the server. We show an implementation of $f(\cdot)$ that modifies client queries to censor $M_{(a,b)}$. The filter works as follows:

1. $f(\cdot)$ receives query $(y_1, ..., y_t, N)$. $f(\cdot)$ creates a $y' = z^2$ where $z \leftarrow_R \mathbb{Z}^*_N$.
2. $f(\cdot)$ replaces y_b (where b is the column that holds the bit the filter wishes to censor) with y'.
3. $f(\cdot)$ forwards the modified query, $(y_1, ...y', ..., y_t, N)$ to DB.

We now calculate the censorship susceptibility of the system. Let us assume that the PIR database holds the value 0 for all indices other than (a, b). For the bit $M_{(a,b)}$, the value is 1. Recall that an adversary's censorship susceptibility advantage is calculated as follows:

$$\mathbf{Adv}^{\mathrm{CR}}_{sys}(A_{\mathrm{CR}}) = \Pr[A_{\mathrm{CR}}\ \mathsf{block\ targeted}] - \min_{name' \in Name} \Pr[A_{\mathrm{CR}}\ \mathsf{block\ not\ targeted}].$$

where $sys = \mathrm{QRA}$. From the filter we created, the probability of "A_{CR} block targeted" is calculated as:

$$\Pr[A_{\mathrm{CR}}\ \mathsf{block\ targeted}] = \Pr[M_{(a,b)} = 1] \tag{5}$$

The filter algorithm replaces y_b with y' for every query that it sees. Because y' is guaranteed to be a QR, regardless of the value stored on the database, the modified query result always returns a QR, effectively returning a 0. Since our database is such that $M_{(a,b)} = 1$, this modification guarantees that the client is unable to retrieve the correct bit. Therefore, the probability that the filter blocks the targeted bit is equal to 1.

Using a similar argument, we can calculate $\Pr[A_{\mathrm{CR}}\mathsf{block\ not\ targeted}]$. Let $M_{(a',b')}$ be the bit the client wishes to retrieve. If $b' = b$, then the filter's modification of y_b to y' results in $M_{(a',b')}$ being returned as a 0. Since the database values have been chosen such that only $M_{(a,b)}$ equals 1, a modification of y_b to y' does not affect results for $M_{(a',b)}$ since $M_{x,y} = 0\ \forall x \neq a, y \neq b$. In this case, $\Pr[A_{\mathrm{CR}}\ \mathsf{block\ not\ targeted}]$ is equal to 0.

From our calculations, the censorship susceptibility of this CR protocol is 1. Thus, PIR does not trivially implement CR.

Combining the result of this theorem with Theorem 5.2, we have shown that PIR is necessary but not sufficient to implement CR.

7 A CR Implementation

In this section, we show a CR implementation by modifying any generic PIR protocol.

In Section 6, we showed that PIR does not implement CR. A filter can simply modify client queries to censor a targeted document. We show a modification of the PIR protocol such that client queries cannot be altered without detection.

A client, c, can detect if her request has been altered if ds simply includes c's request in its reply to c. ds can digitally sign each message with its response and c's request. Thus, c's $\mathsf{get}(\cdot)$ verifies that the request it generated is unaltered and that the response comes from ds. If the query has been altered, then $\mathsf{get}(\cdot)$ fails to retrieve the document. We denote this new CR system as sys+S.

For the following theorem, we implement the CR system, sys, using a generic PIR protocol, P. Our modification leads us to the following theorem:

Theorem 3. $\forall A_{\mathrm{CR}}, \exists\, A_{\mathrm{PIR}} \;:\; \mathbf{Adv}_{\mathrm{P}}^{\mathrm{PIR}}(A_{\mathrm{PIR}}) \geq \mathbf{Adv}_{\mathrm{P+S}}^{\mathrm{CR}}(A_{\mathrm{CR}})$.

Proof. Recall the document retrieval protocol from Section 3. $f(\cdot)$ may modify the query before it invokes ds for a response, and $f(\cdot)$ may drop the query after ds finishes computing its answer for the query. However, because ds includes c's query in its digitally signed response, $f(\cdot)$ is unable to modify the query without c detecting the modification. Thus, $f(\cdot)$'s only viable option is to drop queries. Given that the CR system, P+S, uses a generic PIR protocol, P, as its underlying retrieval protocol, we prove that if A_{CR} has some advantage, $\mathbf{Adv}_{\mathrm{P+S}}^{\mathrm{CR}}(A_{\mathrm{CR}})$, then there exists a PIR adversary with advantage, $\mathbf{Adv}_{\mathrm{P}}^{\mathrm{PIR}}(A_{\mathrm{PIR}})$ against P.

Similar to Section 6, we assume that documents in the CR system are one-bit documents, and the "descriptive names" of the documents are the indices for a PIR datastring. The PIR adversary works as follows:

1. Given $(name, s) \leftarrow G$, A_{PIR} finds the corresponding index, i, that maps to $name$.
2. For any query q, A_{PIR} passes q to $f(s)$. If $f(s)$ drops the request or response, A_{PIR} outputs 1. Otherwise, A_{PIR} outputs 0.

Recall from Section 5 that the CR adversary's advantage is calculated as:

$$\mathbf{Adv}_{\mathrm{P+S}}^{\mathrm{CR}}(A_{\mathrm{CR}}) = \Pr[A_{\mathrm{CR}} \text{ block targeted}] - \min_{name' \in Name} [\Pr[A_{\mathrm{CR}} \text{ block not targeted}]]\,.$$

Recall that the PIR adversary's advantage is defined as:

$$\mathbf{Adv}_{\mathrm{P}}^{\mathrm{PIR}}(A_{\mathrm{PIR}}) = \max_{i,j} [\Pr[A_{\mathrm{PIR}}(q(i)) = 1] - \Pr[A_{\mathrm{PIR}}(q(j)) = 1]]\,.$$

We can see from our construction of the PIR adversary that:

$$\Pr[A_{\mathrm{PIR}}(q(i)) = 1] = \Pr[A_{\mathrm{CR}} \text{ block targeted}]$$
$$\exists j \;:\; \Pr[A_{\mathrm{PIR}}(q(j)) = 1] = \min_{name' \in Name} \Pr[A_{\mathrm{CR}} \text{ block not targeted}]$$

Thus, we have the following reduction:

$$\begin{aligned}
\mathbf{Adv}_{\mathrm{P}}^{\mathrm{PIR}}(A_{\mathrm{PIR}}) &= \max_{i,j} [\Pr[A_{\mathrm{PIR}}(q(i)) = 1] - \Pr[A_{\mathrm{PIR}}(q(j)) = 1]] \\
&= \Pr[A_{\mathrm{CR}} \text{ block targeted}] - \min_{name' \in Name} \Pr[A_{\mathrm{CR}} \text{ block not targeted}] \\
&= \mathbf{Adv}_{\mathrm{P+S}}^{\mathrm{CR}}(A_{\mathrm{CR}}) \qquad (6)
\end{aligned}$$

8 Conclusion

In this paper, we introduced a formal definition for censorship susceptibility. Intuitively a system with low censorship susceptibility is censorship resistant. Our definition provides a framework with which to evaluate different designs of censorship resistant information services. In this work, we adopt an aggressive threat model, allowing the possibility of a universal injunction and voluntary cooperation of servers with external censor authorities. We show that many current implementations of CR services do not satisfy censorship resistance under this threat model.

Our model allows us to prove intrinsic relationships between the property of censorship resistance and the cryptographic primitive of Private Information Retrieval (PIR). We show that PIR is necessary for censorship resistance, but does not trivially implement CR. We then show an implementation using PIR that does meet our definition of censorship resistance.

References

1. E. Adar and B. A. Huberman. Free Riding on Gnutella. *First Monday*, **5**(10), October 2004.
2. R. J. Anderson. The Eternity Service. *Pragocrypt*, 1996.
3. J. Aspnes, J. Feigenbaum, A. Yampolskiy, and S. Zhong. Towards a Theory of Data Entanglement. *European Symposium on Research in Computer Security*, September 2004.
4. A. Back. The eternity service. *Phrack Magazine. http://www.cypherspace.org/ adam/eternity/phrack.html.*, 7:51, 1997.
5. T. Benes. The strong eternity service. *Information Hiding Workshop*, April 2001.
6. C. Cachin, S. Micali, and M. Stadler. Computationally Private Information Retrieval with Polylogarithmic Communication. *Eurocrypt*, 1999.
7. D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, **24**(2):84-88. ACM Press, 1981.
8. B. Chor and N. Gilboa. Computationally Private Information Retrieval. *Symposium on Theory of Computing*, 1997.
9. I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong. Freenet: A distributed anonymous information storage and retrieval system. *Workshop on Design Issues in Anonymity and Unobservability*, 2000.
10. D. A. Cooper and K. P. Birman. Preserving Privacy in a Network of Mobile Computers. *IEEE Symposium on Security and Privacy*, 1995.
11. G. Danezis and R. Anderson. The Economics of Censorship Resistance. *Workshop on Economics and Information Security*, 2004.
12. R. Dingledine, M. J. Freedman, and D. Molnar. The Free Haven project: distributed anonymous storage service. *Workshop on Design Issues in Anonymity and Unobservability*, 2001.
13. M. J. Freedman and R. Morris. Tarzan: A Peer-to-Peer Anonymizing Network. *ACM Conference on Computer and Communications Security*, 2002.
14. D. Goldschlag, M. Reed, and P. Syverson. Onion Routing for Anonymous and Private Internet Connections. *Communications of the ACM*, **42**(2):39-41. ACM Press, 1999.
15. D. Karger, E. Lehman, T. Leighton, M. Levine, D. Lewin, and R. Panigraphy. Consistent Hashing and Random Trees: Distributed Caching Protocols for Relieving Hot Spots. *ACM Symposium on Theory of Computing*, 1997.
16. E. Kushilevitz and R. Ostrovsky. Replication Is Not Needed: Single Database, Computationally-Private Information Retrieval. *IEEE 38th Annual Symposium on Foundations of Computer Science*, 1997.

17. B. N. Levine, M. K. Reiter, C. Wang, and M. Wright. Timing Attacks in Low-Latency Mix Systems. *Financial Cryptography*, 2004.
18. D. Mazieres and M. F. Kaashoek. The Design and Operation of an E-mail Pseudonym Server. *ACM Conference on Computer and Communications Security*, 1998.
19. J.-F. Raymond. Traffic Analysis: Protocols, Attacks, Design Issues, and Open Problems. *Workshop on Design Issues in Anonymity and Unobservability*, 2000.
20. M. G. Reed, P. F. Syverson, and D. M. GoldSchlag. Proxies for anonymous routing. *12th Annual Computer Security Applications Conference*, 1996.
21. M. K. Reiter and A. D. Rubin. Crowds: Anonymity for Web Transactions. *ACM Transactions on Information and System Security*, **1**(1):66–92.
22. A. Serjantov. Anonymizing censorship resistant systems. *International Workshop on Peer-to-Peer Systems*, 2002.
23. A. Shamir. How to share a secret. *Communications of the ACM*, **22**(11):612-613. ACM Press, 1979.
24. A. Stubblefield and D. S. Wallach. *Dagster: Censorship-Resistant Publishing Without Replication*. Technical Report TR01-380. Rice University, July 2001.
25. P. Syverson, G. Tsudik, M. Reed, and C. Landwehr. Towards an Analysis of Onion Routing Security. *Workshop on Design Issues in Anonymity and Unobservability*, 2000.
26. Gnutella. http://gnutella.wego.com.
27. M. Waldman and D. Mazieres. Tangler: A Censorship Resistant Publishing System Based on Document Entanglement. *ACM Conference on Computer and Communication Security*, 2001.
28. M. Waldman, A. D. Rubin, and L. F. Cranor. Publius: A robust, tamper-evident, censorship-resistant web publishing system. *USENIX Security Symposium*, 2000.
29. M. Wright, M. Adler, B. N. Levine, and C. Shields. An Analysis of the Degradation of Anonymous Protocols. *ISOC Network and Distributed Security Symposium*, 2002.