NEXt generation Techno-social Legal Encryption Access and Privacy nextleap.eu

Grant No. 688722. Project started 2016-01-01. Duration 36 months.

DELIVERABLE D4.5

Privacy-Preserving Analytics for Wisdom of the Crowds Simulation

George Danezis, Marios Isaakidis, and Raphael R. Toledo

University College London

Beneficiaries: **UCL** (lead), IMDEA

Workpackage: **Validation via Formal Modelling and Simulation**

Description: This deliverable demonstrates the simulation-based evaluations of the systems proposed in D2.4. In addition, it presents evidence of online censorship in Cyprus, based on the analysis of the crowd-sourced measurements.

Version: Draft

Nature: Report (R)

Dissemination level: Public (P)

Pages: 35

Date: 2019-1-22

Project co-funded by the European Commission within the Horizon 2020 Programme.

1 Introduction

This deliverable D4.5 is a companion of D2.4 and represents the analysis, and evaluation of the NEXTLEAP systems presented in D2.4, through simulations, as well as mathematical proofs of security, safety and performance. The description of the deliverable within the original project proposal is "*D4.5. Simulation-based evaluation of the scalability of privacy-enhanced crowd-sourcing analytics built on a private information retrieval framework. Depends on D2.4*". Due to the technical nature of the proposed designs, their security arguments, be it mathematical or empirical, are technical and such is the nature of this deliverable.

The structure of this deliverable follows closely the structure of D2.4. For each of the technical designs in D2.4 it provides context, in terms of the related work in the literature, and then a set of full or sketched security arguments, as well as performance measurements.

Specifically, for the novel ε-PIR mechanisms we provide definitions of security, and show that the PIR systems proposed meet those definitions. Furthermore, when composed with an anonymous channel the PIR systems see their privacy enhanced. We also provide an evaluation of performance, and comparison with other PIR systems.

We also provide an evaluation of Blockmania, our novel consensus algorithm. First, we provide arguments about its safety (all honest parties will agree on the same value) and liveness (all honest parties will eventually agree) in a partial synchronous and byzantine context. We then present the performance evaluation of the Blockmania protocols using a real world implementation (work performed in collaboration with `chainspace.io`).

The third part of the deliverable evaluates SybilQuorum as a Sybil defence mechanism, and establishes the necessary preconditions to evaluate whether a set of nodes selected by the SybilQuorum algorithm is a valid and secure Federated Byzantine Agreement System. Such theorems are key to the empirical and simulation based evaluation of SybilQuorum, since it is otherwise difficult to otherwise establish its security.

Finally, we provide the evaluation and results from our study of censorship in Cypus, that used our initial privacy friendly statistics wisdom of the crowds collection techniques.

2 Four ε-Private PIR Systems

2.0.1 Direct Requests

The first ε-private PIR mechanism uses dummy queries on *multiple* PIR databases, of which d_a are adversarial and $(d - d_a)$ are honest. The user generates a query for the sought record, along with $p - 1$ random (distinct) other ones. The requests are partitioned into sets of equal size and sent to the PIR databases directly. Each database then responds with the list of records requested, encrypted as are all communications.

The database servers simply respond to requests by returning the index and the records sought over the encrypted channel. **pop**(Req) returns a random item from the set Req (and also removes it from Req). rec_Q is the sought record of index Q.

Security Theorem 1. *The* direct requests mechanism *is an ε-private PIR mechanism with*

$$\varepsilon = \ln\left(\frac{1}{d-d_a}\cdot\left(d\cdot\frac{n-1}{p-1}-d_a\right)\right),$$

where d is the number of databases, of which d_a are adversarial, n is the total number of records, and p is the total number of queries sent by the user.

Proof. See Appendix 2.4.2. □

Algorithm 2.1: Direct Requests (User)

Input:

Q: $(0 \leq Q < n)$;

p: $(p > 1) \wedge p \equiv 0 \mod d$;

$Req \leftarrow \{Q\}$;
while $|Req| < p$ **do**
  $Q' \leftarrow$ **random**(n);
  if $Q' \notin Req$ **then**
    $Req \leftarrow Req \cup Q'$;
for $1 \leq i \leq d$ **do**
  for $1 \leq j \leq p/d$ **do**
    $r \leftarrow$ **pop**(Req) $(index_r, rec_r) \leftarrow$ **sendreceive**(DB_i, r);
return rec_Q;

Costs. The client needs to communicate p/d record indices to each of d servers. How to best do this depends on the size of p/d. If p/d is small, the best way is simply to send the p/d indices (each of size $\lceil \lg n \rceil$ bits) to each of d servers, for a total cost of $p \cdot \lceil \lg n \rceil$ bits. However, if p/d is large, it is more efficient to simply send an n-bit bitmask to each server, for a total cost of $d \cdot n$ bits. Therefore the total client communication is $C_c = \min(d \cdot n, p \cdot \lceil \lg n \rceil)$ bits. The server communication cost is just $C_s = p$, as p records are requested and sent back to the user. As the databases do not XOR any records but just accesses and sends them, the computation cost is $C_p = p \cdot c_{acc}$.

Practical values. Fig. 1 illustrates Direct Request curves representing ε as a function of p for different adversaries in the reference scenario of Certificate Transparency. As millions of certificates have already been recorded in databases and hundreds of databases are supposed to be running all over the world, we have set $n = 10^6$ and $d = 10^2$. The security parameter ε starts above 10 and slowly diminishes until nearly all of the records have been requested where the curves follow the vertical asymptote $p = n$. If weaker adversaries decrease ε for any p, the difference becomes really noticeable only after requesting a tenth of the records. Further, in order to achieve even a mediocre security of $\varepsilon < 1$, for any d_a, more than $\frac{9}{10}$ of the records have to be requested. In the worst-case scenario where only one database is not colluding, we find the security parameter ε is approximately equal to 11.5. However if only half of the databases are corrupted, i.e. $d_a = \frac{1}{2} \cdot d$, we have $\varepsilon \approx 7.6$. To summarize for $n = 10^6$, $d = 10^2$ and $p = 10 \cdot d$, if $d_a = d - 1$ we have $\varepsilon \approx 11.5$ while if $d_a = \frac{d}{2}$, we have $\varepsilon \approx 7.6$. For any d_a, to obtain $\varepsilon < 1$, $p > \frac{9}{10} \cdot n$.

In the case of a small database system consisting of a few to tens of databases, each storing thousands of records, we set $n = 10^3$ and $d = 10$. When the adversary controls all databases but one, if the user only sends one request per database we have that $\varepsilon \approx 7$ while when half of the databases are corrupted, $d_a = \frac{1}{2} \cdot d$, we have $\varepsilon \approx 5.4$. To summarize for $n = 10^3$, $d = 10$ and $p = d$, if $d_a = d - 1$ we have $\varepsilon \approx 7$ while if $d_a = \frac{d}{2}$, we have $\varepsilon \approx 5.4$.

The above examples illustrate that for large databases, as the one considered in the motivating Certificate Transparency example, an adversary controlling about half the databases can extract a lot of information. Furthermore, information leakage does not diminish significantly based on the security parameter p, or for smaller databases. Thus we conclude the Direct Requests mechanism alone provides very weak privacy; however, we will show how its composition with an anonymity system can improve its performance.

2.0.2 Anonymous direct requests

Bundled anonymous request We compose the *direct requests* mechanism from the previous subsection with an anonymous channel. Each user, including the target user $\mathcal{U}_t$, sends a bundle of requests defined by the *direct*

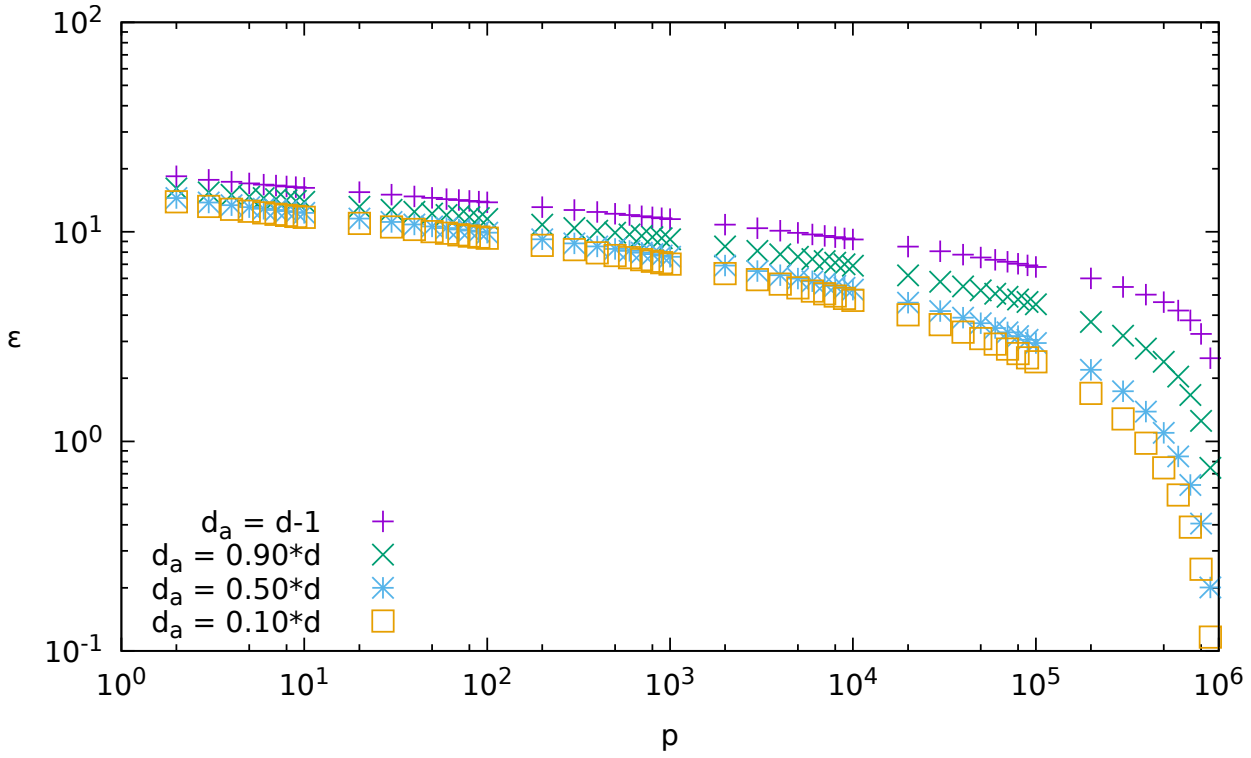


Figure 1: Direct requests: ε versus p for $d = 100$ and $n = 10^6$

requests PIR mechanism to databases through an anonymity system $\mathcal{AS}$.

The requests are *bundled*, in that all requests from a specific user are linkable with each other, allowing this mechanism to be implemented by sending a single anonymous message through the $\mathcal{AS}$ per user. The $\mathcal{AS}$'s exit node receiving the bundle forwards the different sets of queries (as usual, encrypted by the user to each respective database) to the relevant database and anonymously returns the requested records from each database.

The increased privacy of this scheme derives from the ability of the target user $\mathcal{U}_t$ to hide the use of the PIR system amongst $u-1$ other users. This strengthens the direct requests mechanism hiding $\mathcal{U}_t$'s query amongst $p-1$ random requests throughout d servers. The adversary's task becomes harder as any bundle, out of u, could be the target's, and any query, out of p, the correct one.

When seeing one of the non-target queries Q_{n_k}, the adversary can however link the corresponding bundle to a non-target user $\mathcal{U}_k$ with overwhelming probability, discard it, and thus reduce the his analysis to fewer users. Not to discard the target's bundle, the adversary should minimize the probability the target user chooses an element of Q_n, which equals $(p-1)\cdot\frac{|Q_n|}{n}$. To do so, the adversary thus should send the same non-target query Q_{n_k} to all non-target users, creating a non-target query set of size 1; that is, simply choosing $Q_n = \{Q_0\}$ would result in the best observation for the adversary.

The database servers simply respond to bundles by returning the index and the records sought over the encrypted channel, the anonymity system forwarding the answer to the corresponding users.

Security Theorem 2. *The* bundled anonymous requests mechanism *is* ε*-private with*

$$\varepsilon = \ln\left(\left(\frac{d}{d-d_a}\cdot\frac{n-1}{p-1} - \frac{d_a}{d-d_a}\right)^2 + u - 1\right) - \ln u.$$

Proof. By applying our Composition Lemma (see below), and the security parameter of the direct requests mechanism. □

Costs. As the only differences with the Direct Request case is the Anonymity system and the bundling of the

Algorithm 2.2: Bundled Anonymous Requests (User)

Input:
Q: $(0 \leq Q < n)$;
p: $(p > 1) \wedge p \equiv 0 \mod d$;

$Req \leftarrow \{Q\}$;
while $|Req| < p$ **do**
  $Q' \leftarrow$ **random**(n);
  if $Q' \notin Req$ **then**
    $Req \leftarrow Req \cup Q'$;
$Bundle \leftarrow \{\}$
for $1 \leq i \leq d$ **do**
  for $1 \leq j \leq p/d$ **do**
    $Bundle_i \leftarrow$ **pop**(Req)
  $Bundle \leftarrow (DB_i, Bundle_i)$
$(index_r, rec_r) \leftarrow$ **anonsendreceive**$(DS, Bundle)$;
return rec_Q;

messages, we find the same values for the communication costs $C_c = \min(d \cdot n, p \cdot \lceil \lg n \rceil)$ and $C_s = p$, and the computation cost $C_p = p \cdot c_{acc}$.

Separated anonymous request We may also compose the *direct requests* mechanism (Sect. 2.0.1) with an anonymous channel in a different manner. Each user, including the target user $\mathcal{U}_t$, sends distinct requests defined by the *direct requests* PIR mechanism to databases through an anonymity system $\mathcal{AS}$, whose queries are *unlinkable* at the mix output.

The requests are *separated*, in that all requests from a specific user are unlinkable with each other, allowing this mechanism to be implemented by sending separate anonymous messages through the $\mathcal{AS}$ to different databases. The $\mathcal{AS}$'s exit node receiving the message forwards it to the relevant database and anonymously returns the requested record.

The increased privacy of this scheme derives from the ability of the target user $\mathcal{U}_t$ to hide the real query of the PIR system amongst $u \cdot (p-1)$ other random queries.

Algorithm 2.3: Separated Anonymous Requests (User)

Input:
Q: $(0 \leq Q < n)$;
p: $(1 < p) \wedge p \equiv 0 \mod d$;

$Req \leftarrow \{Q\}$;
while $|Req| < p$ **do**
  $Q' \leftarrow$ **random**$(0, n)$;
  if $Q' \notin Req$ **then**
    $Req \leftarrow Req \cup Q'$;
forall $i \in p$ **do**
  $r \leftarrow$ **pop**(Req);
  $(index_r, rec_r) \leftarrow$ **anonsendreceive**(DS_i, r);
return rec_Q;

Since the Bundled Anonymous Requests mechanism leaks strictly more information than Separated Anonymous Request, the ε_{bundle} is also an upper bound of the Separated Anonymous Request.

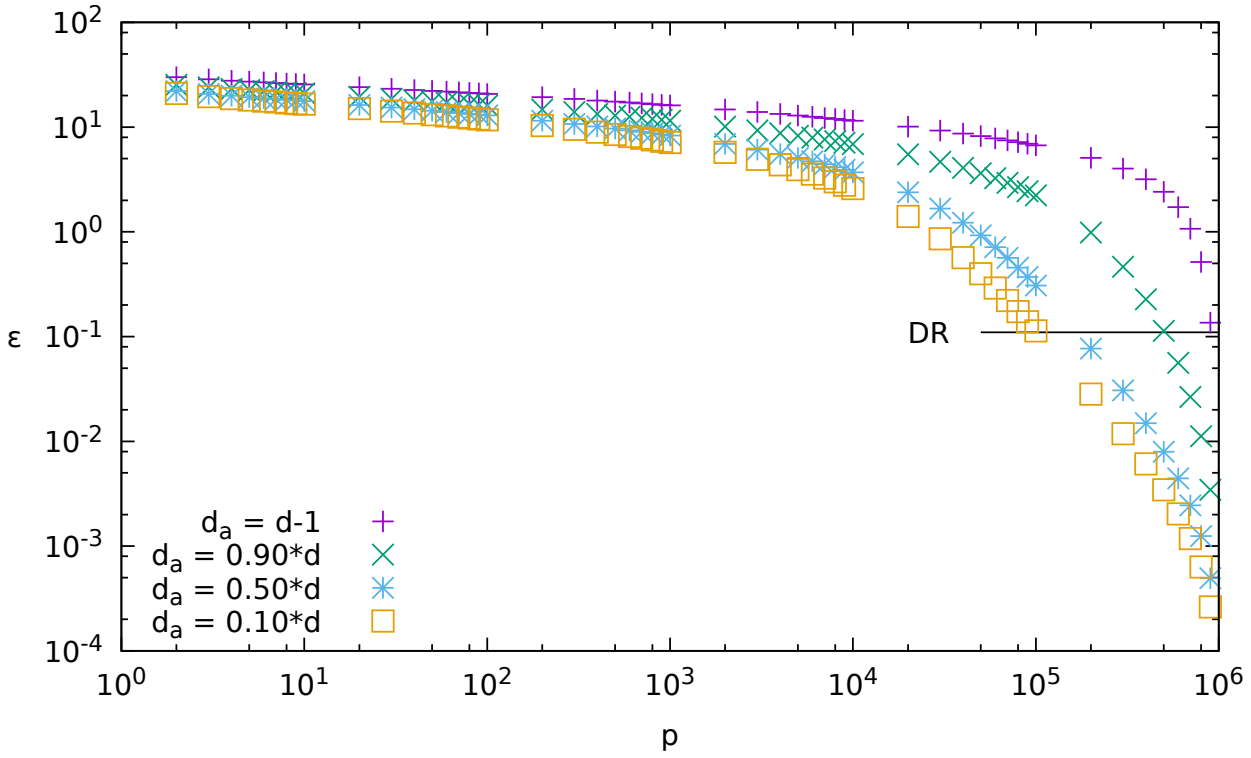


Figure 2: AS-Bundle: ε versus p for $d = 100$, $n = 10^6$, and $u = 10^3$

Costs. As this method is similar with the Bundled case, we have for costs $C_c = \min(d \cdot n, p \cdot \lceil \lg n \rceil)$, $C_s = p$ and $C_p = p \cdot c_{acc}$. However, the load on the anonymity system increases as there are $u \cdot p$ anonymous messages transmitted.

Practical values. Fig. 2 shows Direct Request composed with anonymity system curves representing ε as a function of p for different adversaries in the reference scenario of Certificate Transparency. As before, we set $n = 10^6$ and $d = 10^2$ and assumed $u = 10^3$. The security parameter ε starts above 10 and slowly diminishes until a tenth to most of the records have been recorded depending on d_a where the curves follow the vertical asymptote $p = n$. The anonymity system gain in privacy can be seen under the line indicating where the privacy of the Direct Request protocol, without an anonymity system, stops for the same amount of points. If the anonymity system gains appear negative at the beginning of the curves, this is due to the lack of tightness of the bound in the Composition Lemma. If weaker adversaries decrease ε for any p, the difference becomes really noticeable only after requesting a hundredth of the records. Further, in order to achieve even a mediocre security of $\varepsilon < 1$, for any d_a, at most half of the records have to be requested compared to 90% without an anonymity system. In the worst-case scenario where only one database is not colluding, we find the security parameter ε is approximately equal to 16. However if only half of the databases are corrupted, i.e. $d_a = \frac{1}{2} \cdot d$, we have $\varepsilon \approx 8$. To summarize for $n = 10^6$, $d = 10^2$, $u = 10^3$ and $p = 10 \cdot d$, if $d_a = d - 1$ we have $\varepsilon \approx 16$ while if $d_a = \frac{d}{2}$, we have $\varepsilon \approx 8$.

In the case of a small database system managing a few to tens of databases, each storing thousands of records, we again set $n = 10^3$ and $d = 10$. When the adversary controls all databases but one, each sending only one request per database, we have that $\varepsilon \approx 7$ while when half of the databases are corrupted, $d_a = \frac{1}{2} \cdot d$, we have $\varepsilon \approx 4$. To summarize for $n = 10^3$, $d = 10$, $u = 10^3$ and $p = d$, if $d_a = d - 1$ we have $\varepsilon \approx 7$ while if $d_a = \frac{d}{2}$, we have $\varepsilon \approx 4$.

We conclude that direct requests through an anonymity system is a stronger mechanism that direct requests alone. However, for very large databases, such as the one expected in Certificate Transparency, the quality

of protection is still low. It becomes better only as the total volume of requests from all users is in the order of magnitude of the number of records in the database. This requires either a large number of users, or a large number of dummy requests per user. However, even the weaker protection afforded by anonymous direct requests may be sufficient to protect privacy in applications where records only need to be accessed infrequently.

2.0.3 Sparse-PIR

We next adapt Chor's simplest IT-PIR scheme [Cho+95] to reduce the number of database records accessed to answer each query. As a reminder: in Chor's scheme the user builds a set of random binary vectors of length n (the number of records in the database), one for each server; we call these vectors the "request vectors". These are constructed so that their element-wise XOR yields a zero for all non-queried records, and a one for the record sought (we call this the "query vector"). Each server simply XORs all records corresponding to a 1 in its request vector, and returns this value to the user. The XOR of all responses corresponds to the sought record.

Sparse-PIR aims to reduce the computational load on the database servers $\mathcal{DB}_i$. To this end the binary request vectors are not sampled uniformly but are sparse, requiring the database servers to access and XOR fewer records to answer each query. Specifically, in Sparse-PIR each request is derived by independently selecting each binary element using a Bernoulli distribution with parameter $\theta \leq 1/2$. Furthermore, the constraint that the XOR of these sparse vectors yields the query vector is maintained. The intuition is that we will build a $d \times n$ query matrix M *column wise*: each column (of length d) corresponds to one record in the database, and will be selected by performing d independent Bernoulli trials with parameter θ, re-sampling if necessary to ensure the sum of the entries in the column (the Hamming weight) is even for non-queried records, or odd for the single queried record.

Equivalently, we may first select a Hamming weight for each column with the appropriate probability depending on d, θ, and whether the column represents the queried record or not, and then select a uniformly random vector of length d with that Hamming weight. Each row of the query matrix will then have expected Hamming weight $\theta \cdot n$, and the rows of the matrix (the request vectors) will XOR to the desired query vector, namely all 0 except a single 1 at the desired location.

Algorithm 2.4: Sparse-PIR (User)

Input:
Q: $0 \leq Q < n$;
θ: $0 < \theta \leq \frac{1}{2}$;

$M \leftarrow [\,]$;
for $0 \leq col < n$ **do**
  if $col = Q$ **then**
    $q \leftarrow d$ Bernoulli(θ) trials with Odd sum;
  else
    $q \leftarrow d$ Bernoulli(θ) trials with Even sum;
  $M \leftarrow M$ **append column** q;
for $1 \leq i \leq d$ **do**
  $r_i \leftarrow$ **row** i **of** M;
  $resp_i \leftarrow$ **sendreceive**(DB_i, r_i);
return $\bigoplus_{1 \leq i \leq d} resp_i$;

The database logic in Sparse-PIR is identical to the logic in Chor's IT-PIR: each database server receives a binary vector, XORs all records that correspond to entries with a 1, and responds with the result. In fact the database may be agnostic to the fact it is processing a sparse PIR request, aside from the reduction in the number of entries to be XORed. For $\theta < 0.5$ the costs of processing at each database is lowered due to the relative sparsity of ones, at no additional networking or other costs.

Security Theorem 3. *The Sparse-PIR mechanism is ε-private with*

$$\varepsilon = 4 \cdot \operatorname{arctanh}[(1-2\theta)^{(d-d_a)}],$$

where θ is the parameter of the Bernoulli distribution and $d - d_a$ represents the number of honest PIR servers.

Proof. See Appendix 2.4.3. □

As expected when $\theta = 1/2$ the privacy provided by the Sparse-PIR mechanism is the same as for the perfect IT-PIR mechanism. This fact can be derived from the tight bound on ε by observing that ε equals zero when $\theta = 1/2$.

Security Lemma 1. *For $\theta = 1/2$, and at least one honest server, the Sparse-PIR mechanism provides perfect privacy, namely with $\varepsilon = 0$.*

More interestingly, as the number of honest servers increases, the privacy of the Sparse-PIR increases for any θ, and in the limit becomes perfect as in standard IT-PIR:

Security Lemma 2. *For an increasing number of honest servers $(d - d_a) \to \infty$ the Sparse-PIR mechanism approaches perfect privacy, namely $\varepsilon \to 0$.*

Proof. Note that for $0 < \theta < 1$, $\lim_{x\to\infty}(1-2\theta)^x = 0$. Thus for $(d - d_a) \to \infty$ we have that $\varepsilon \to 0$, since $\operatorname{arctanh}(0) = 0$. □

Costs. In Sparse-PIR, the client requests the XOR of $\theta \cdot n$ records from each of the d servers. As above, how best to do this depends on the size of θ. If $\theta > \frac{1}{\lg n}$, then sending an n-bit bitmask to each of the d servers will do (just as in the standard Chor case, where effectively $\theta = \frac{1}{2}$); if θ is smaller, however, then listing the $\theta \cdot n$ indices, at a cost of $\theta \cdot n \lceil \lg n \rceil$ bits for each of the d servers is cheaper. The total client communication cost is therefore $C_c = d \cdot \min(n, \theta \cdot n \lceil \lg n \rceil)$. On each of the d servers, only $\theta \cdot n$ records are accessed and operated on per request, and a single record is sent. We thus have $C_s = d$ and $C_p = \theta \cdot d \cdot n \cdot (c_{acc} + c_{prc})$.

Practical values. Fig. 3 shows Sparse-PIR curves representing ε as a function of θ for different adversaries in the reference scenario of Certificate Transparency with $d = 10^2$. The security parameter ε starts below 10 and slowly diminishes until nearly all of the records have been accessed for $\theta = \frac{1}{2}$ where the curves follow a vertical asymptote. The difference in ε for different adversaries is noticeable at any point of the curves. In order to achieve even a mediocre security of $\varepsilon < 1$, except for the worst case $d_a = d - 1$, accessing 10% of the records at each database is enough. In the worst-case scenario where only one database is not colluding, we find the security parameter ε is approximately equal to 2 for $\theta = 0.25$. However if only half of the databases are corrupted, i.e. $d_a = \frac{1}{2} \cdot d$, we have $\varepsilon \approx 10^{-15}$ for the same θ. To summarize for $d = 10^2$ and $\theta = 0.25$, if $d_a = d - 1$ we have $\varepsilon \approx 2$ while if $d_a = \frac{d}{2}$, we have $\varepsilon \approx 10^{-15}$.

In the case of a small database systems managing a few to tens of databases, we set $d = 10$. When the adversary controls all databases but one, we have the $\varepsilon \approx 2$ while when half of the databases are corrupted, $d_a = \frac{1}{2} \cdot d$, we have $\varepsilon \approx 10^{-1}$. To summarize for $d = 10$ and $\theta = 0.25$, if $d_a = d - 1$ we have $\varepsilon \approx 2$ while if $d_a = \frac{d}{2}$, we have $\varepsilon \approx 10^{-1}$.

A sparse version of the simple Chor scheme can indeed protect the user's privacy better than the direct request, as we can observe a factor of 9 between the two epsilons. Yet, in the worst-case scenario, where the adversary controls all the databases except one, the risk is still significant: the adversary infers that the user is about 7 times more likely to seek a particular record over another. Thus we consider strengthening the system through composition with an anonymous channel.

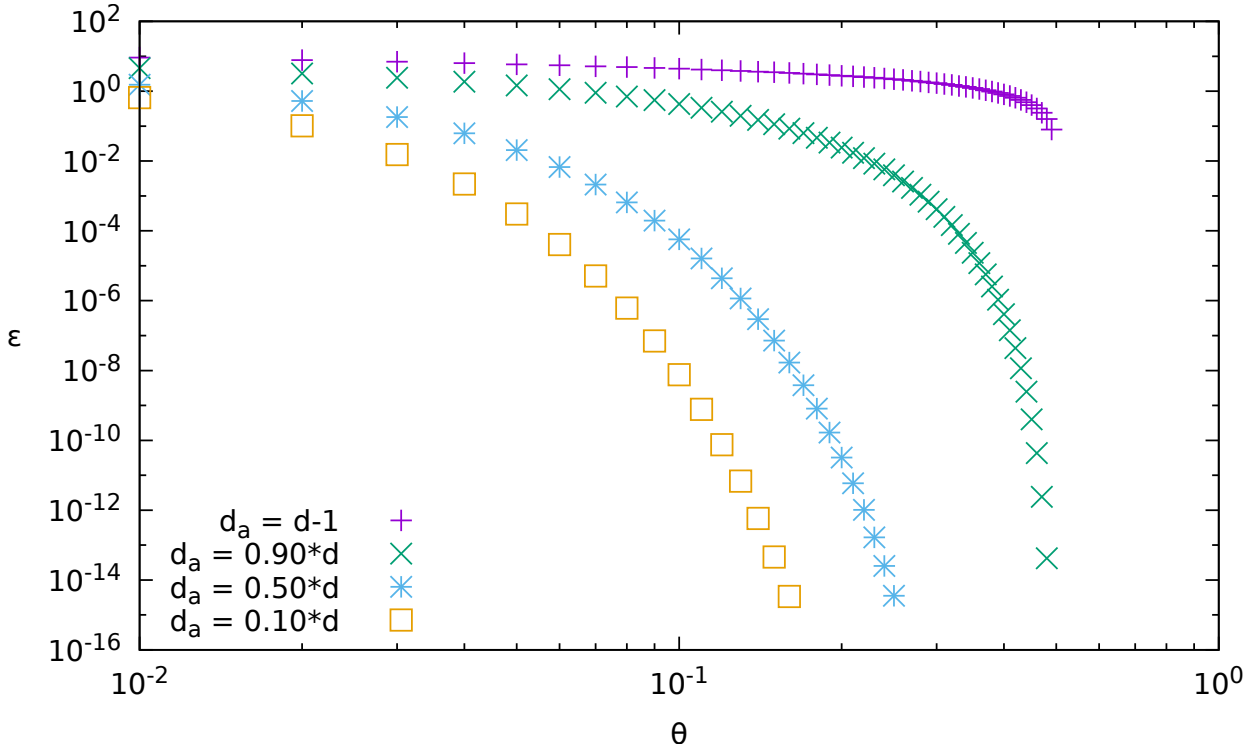


Figure 3: Sparse-PIR: ε versus θ for $d = 100$

2.0.4 Anonymous Sparse-PIR

We consider the composition of the Sparse-PIR mechanism with an anonymity system. In this setting, a number of users u select their queries to the database servers, and perform them anonymously through an anonymity system. We consider that all requests from the same user are linkable to each other at the input and output of the anonymity system. As per our standard setting, the adversary provides a target user $\mathcal{U}_t$ with queries Q_i and Q_j, one of which the user chose, and all other $u-1$ users with $Q_{n_k} \in Q_n$. They all use an *arbitrary* ε-private PIR mechanism through an anonymity channel to perform their respective queries.

Algorithm 2.5: Anonymous Sparse-PIR (User)

Input:
Q: $0 \leq Q < n$;
θ: $0 < \theta \leq \frac{1}{2}$;

$M \leftarrow [\,]$;
for $0 \leq col < n$ **do**
  if $col = Q$ **then**
    $q \leftarrow d$ Bernoulli(θ) trials with Odd sum;
  else
    $q \leftarrow d$ Bernoulli(θ) trials with Even sum;
  $M \leftarrow M$ **append column** q;
for $1 \leq i \leq d$ **do**
  $r_i \leftarrow$ **row** i **of** M;
  $resp_i \leftarrow$ **anonsendreceive**(DB_i, r_i);
return $\bigoplus_{1 \leq i \leq d} resp_i$;

We will show that this mechanism is ε-private, through first proving a general composition lemma. This could be of independent interest to designers of private query systems based on anonymous channels.

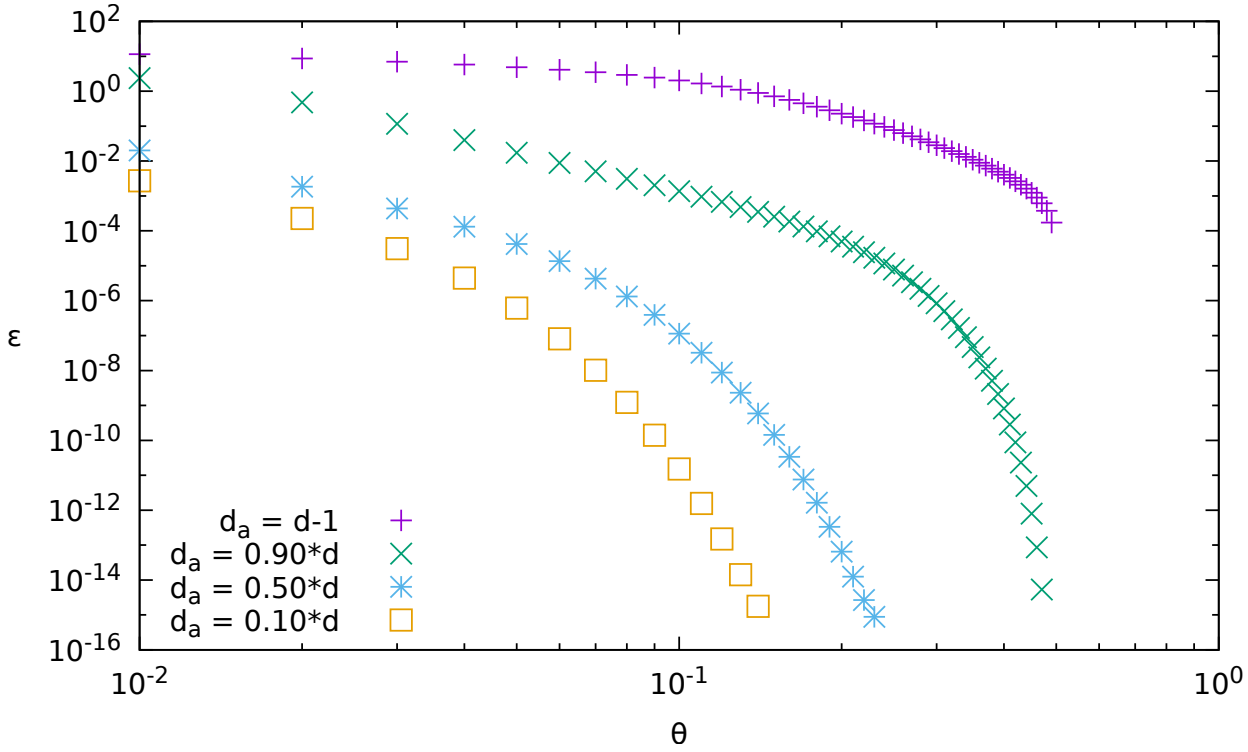


Figure 4: AS-Sparse-PIR: ε versus θ for $d = 100$ and $u = 10^3$

Composition Lemma. *The composition of an arbitrary ε_1-private PIR mechanism with a perfect anonymity system used by u users, for sufficiently large u, yields an ε_2-private PIR mechanism with:*

$$\varepsilon_2 = \ln(e^{2\varepsilon_1} + u - 1) - \ln u.$$

Proof. See Appendix 2.4.4. Note this is not a worst-case analysis, but an average-case analysis over the honest users' randomness, which can not be influenced by the adversary. Namely there is a negligible probability in u, the number of users in the anonymity system, this does not hold. A fuller (ε, δ)-privacy definition could capture the worst-case behaviour. □

It is easy to show that as $u \to \infty$, the parameter $\varepsilon_2 \to 0$, leading to a perfect IT-PIR mechanism, independently of the value of ε_1 (so long as it is finite). Conversely, when $u = 1$, we have $\varepsilon_2 = 2\varepsilon_1$ (the loss of a factor of 2 is due to the lack of tightness of the bound). Using this lemma, we can prove our main theorem.

Security Theorem 4. *The composition of the Sparse-PIR scheme with parameters θ, d, and d_a with an anonymity system with u users is also ε-private with security parameter*

$$\varepsilon = \ln\left(\left(\frac{1+(1-2\theta)^{(d-d_a)}}{1-(1-2\theta)^{(d-d_a)}}\right)^4 + u - 1\right) - \ln u.$$

Proof. This is a direct consequence of the Composition Lemma, the security parameter of Sparse-PIR, and the definition of arctanh. □

Cost The use of an anonymity system does not change any of the communication or computation costs. The communication costs remain $C_c = d \cdot \min(n, \theta \cdot n \lceil \lg n \rceil)$ and $C_s = d$, and the computation cost remains $C_p = \theta \cdot d \cdot n \cdot (c_{acc} + c_{prc})$.

Practical values. Fig. 4 shows Sparse-PIR composed with anonymity system curves representing ε as a function of θ for different adversaries in the reference scenario of Certificate Transparency, with $d = 10^2$ and $u = 10^3$. The security parameter ε starts below 10 and slowly diminishes until nearly all of the records have been accessed for $\theta = \frac{1}{2}$ where the curves follow a vertical asymptote. If the anonymity system gains appear negative at the beginning of the curves, this is due to the lack of tightness of the bound in the Composition Lemma. The difference in ε for different adversary is noticeable at any point of the curves. In order to achieve even a mediocre security of $\varepsilon < 1$, except for the worst case $d_a = d - 1$, accessing more than 10% of the records at each database is enough. In the worst-case scenario where only one database is not colluding, assuming there are 1000 users, we find the security parameter ε is approximately equal to 10^{-1}. However, if only half of the databases are corrupted (i.e., $d_a = \frac{1}{2} \cdot d$), we have $\varepsilon < 10^{-15}$. To summarize for $d = 10^2$, $u = 10^3$ and $\theta = 0.25$, if $d_a = d - 1$ we have $\varepsilon \approx 10^{-1}$ while if $d_a = \frac{d}{2}$, we have $\varepsilon < 10^{-15}$.

In the case of a small database system managing a few to tens of databases we set $d = 10$. When the adversary controls all databases but one, if there are 1000 users, each sending only one request per database, we have the $\varepsilon \approx 10^{-1}$ while when half of the databases are corrupted, $d_a = \frac{1}{2} \cdot d$, we have $\varepsilon \approx 10^{-3}$. To summarize for $d = 10$, $u = 10^3$ and $\theta = 0.25$, if $d_a = d - 1$ we have $\varepsilon \approx 10^{-1}$ while if $d_a = \frac{d}{2}$, we have $\varepsilon \approx 10^{-3}$.

Anonymous Sparse-PIR allows us to easily trade off θ (which governs the server-side cost of the protocol) with u (the number of simultaneous users of the database). If the number of users is high, then by composing Sparse-PIR with an anonymity system, we can reduce θ and still achieve a low ε.

2.1 Optimizing PIR

In this section, we propose an optimization for PIR systems to render them more scalable, but at a higher risk.

2.1.1 Subset-PIR

In order to lower both the communication and computation costs, when $d \gg 1$, one could consider doing IT-PIR on a subset of just t of the databases. We call this optimization *Subset-PIR*. This optimization applies to any IT-PIR protocol, so long as that protocol can be used with a *client-selected* number of replicated servers. Chor's protocol is an example of such a flexible IT-PIR protocol.

The communication and server side computation costs are thus multiplied by a factor of $\frac{t}{d}$ at the cost of a greater risk of all contacted databases being compromised. Consequently, even if an IT-PIR scheme were perfectly private, this optimization induces a non-zero probability of the adversary being able to breach it.

Security Theorem 5. *Subset-PIR is an (ε, δ)-private PIR optimization with*

$$\varepsilon = 0 \;\; \textit{and} \;\; \delta = \prod_{i=0}^{t-1} \frac{d_a - i}{d - i}$$

where d is the number of databases, of which d_a are compromised, and t represents the number of PIR servers contacted. When $t > d_a$ the mechanism provides unconditional privacy.

Proof. The probability of contacting t databases out of which t_a are compromised, knowing that there are in total d_a compromised databases out of d is:

$$\Pr(t_a, t \mid d_a) = \frac{\binom{d_a}{t_a} \cdot \binom{d - d_a}{t - t_a}}{\binom{d}{t}}$$

The probability of contacting only compromised databases is obtained by setting $t_a = t$, and so is $\frac{\binom{d_a}{t}}{\binom{d}{t}}$, which equals $\prod_{i=0}^{t-1} \frac{d_a - i}{d - i}$ if $t \leq d_a$, and 0 if $t > d_a$. □

Algorithm 2.6: Subset-PIR (User)

Input:

Q: $0 \leq Q < n$;

t: $2 \leq t \leq d$;

for $1 \leq j \leq t-1$ **do**

  $P_j \leftarrow n$ Bernoulli($\frac{1}{2}$) trials;

// e_Q is the vector with all 0s except a 1 at position Q

$P_t \leftarrow \left(\oplus_{j=1}^{t-1} P_j\right) \oplus e_Q$;

$DB \leftarrow \{\}$;

while $|DB| \leq t$ **do**

  server $\leftarrow$ **random**(d);

  if *server* $\notin DB$ **then**

    $DB \leftarrow DB \cup \{\text{server}\}$;

for $1 \leq j \leq t$ **do**

  $resp_j \leftarrow$ **sendreceive**$(DB_{DB[i]}, P_j)$;

return $\bigoplus_{i \in t} resp_i$;

Costs. For Subset-PIR, as we contact t databases, we have $C_s = t$ and using a Chor-like PIR protocol we have the computation cost $C_p = \frac{1}{2} \cdot t \cdot n \cdot (c_{acc} + c_{prc})$.

Practical values. Fig. 5 shows Subset-PIR curves representing δ as a function of the number of databases contacted t for different adversaries in the reference scenario of Certificate Transparency, with $d = 10^2$. The security parameter δ starts between 10^{-1} and 1 and slowly diminishes until a tenth to most of the databases have been contacting depending on d_a where the curves follow a vertical asymptote at $t = d$. The difference in δ for different adversaries is noticeable at any point of the curves. In order to achieve even a mediocre security of $\delta < 10^{-1}$, excluding the worst case $d_a = d - 1$, less than 20% of the databases have to be contacted. In the worst-case scenario where only one database is not colluding assuming the user contacts only a tenth of the databases, we find the security parameter δ is approximately equal to 0.9. However if only half of the databases are corrupted (i.e., $d_a = \frac{1}{2} \cdot d$), we have $\delta \approx 10^{-4}$. To summarize for $d = 10^2$ and $t = \frac{1}{10} \cdot d$, if $d_a = d - 1$ we have $\delta \approx 0.9$ while if $d_a = \frac{d}{2}$, we have $\delta \approx 10^{-4}$.

In the case of a small database system managing a few to tens of databases, each storing thousands of records, we set $d = 10$. When the adversary controls all databases but one, if the user contacts a tenth of the databases, we have that $\delta \approx 0.9$ while when half of the databases are corrupted, $d_a = \frac{1}{2} \cdot d$, we have $\delta \approx 0.5$. To summarize for $d = 10$ and $t = \frac{1}{10} \cdot d$, if $d_a = d - 1$ we have $\delta \approx 0.9$ while if $d_a = \frac{d}{2}$, we have $\delta \approx 0.5$.

Perfectly private ($\varepsilon = 0$) IT-PIR designs used in conjunction with the Subset-PIR optimization become (ε, δ)-private with $\varepsilon = 0$ and δ reasonably small, if the number of honest database servers is large. Indeed, Subset-PIR is still perfectly private, with $\varepsilon = \delta = 0$, if the number of servers contacted (t) exceeds the number of adversarial servers (d_a).

Demmler et al. [DHS14] explore a similar idea with their RAID-like design of a PIR system. In RAID-PIR, rather than the client only contacting a subset of the servers, it instead divides the database and the queries into *chunks*, and sends the query chunks corresponding to database chunk i to just t of the servers. By systematically picking which t servers to use for each chunk, however, RAID-PIR does contact all d servers, but each server only does t/d of the work it would ordinarily do, and indeed, each server need only store a t/d fraction of the database, if all clients are required to use the same value of t. Using only this feature of RAID-PIR yields a scheme with the same communication and computation costs as Subset-PIR. RAID-PIR goes further, however, and proposes to generate most of the random query values using a PRNG rather than a truly random string. This greatly reduces the client-to-server communication cost of RAID-PIR, at the cost of changing RAID-PIR from an IT-PIR protocol to a CPIR protocol [Gol07, footnote 1].

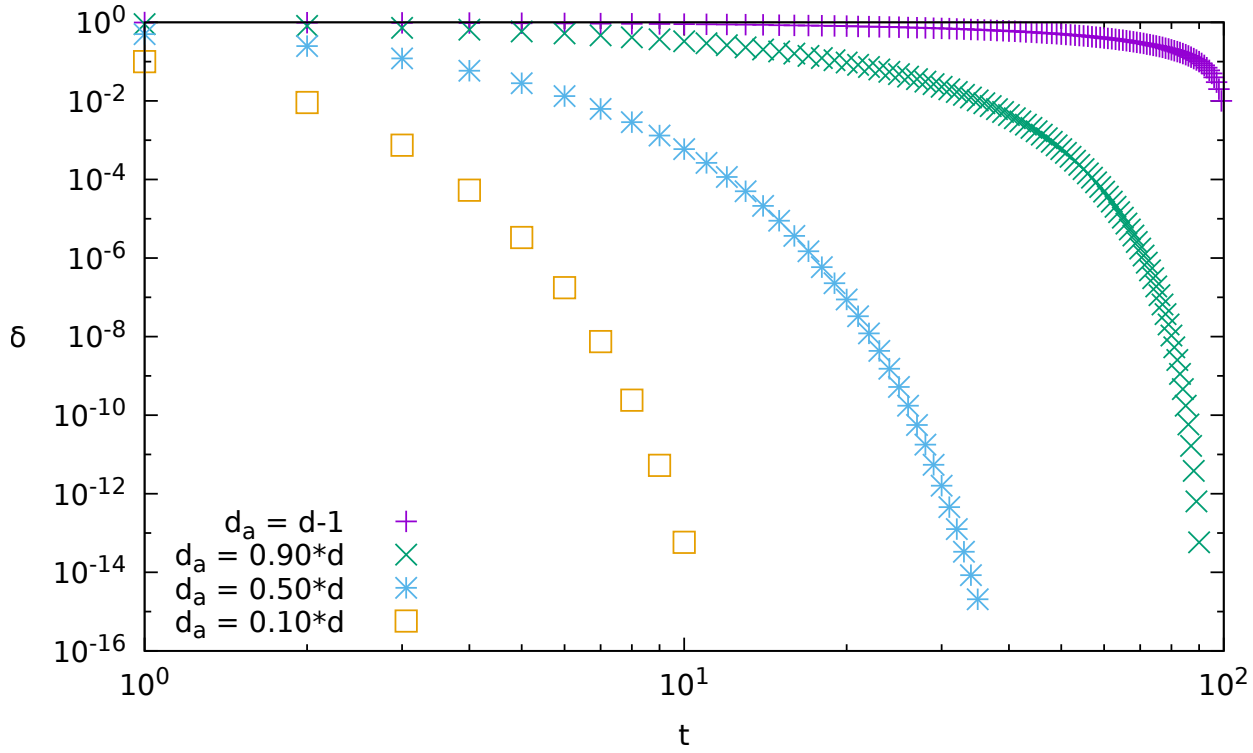


Figure 5: Subset-PIR: δ versus t for $d = 100$

2.2 Comparative Evaluation

In Table 1, we summarize for each scheme presented the security parameters ε and δ, the communication costs C_s, the number of blocks sent back to the user, and the computational cost C_p which depends on the access cost c_{acc} and the processing cost c_{prc}; i.e., the cost associated to the number of records XORed.

When the protocols are not fully private (i.e., $\varepsilon \neq 0$), we observe a reduction in the server computation costs. The Sparse-PIR scheme diminishes the computation cost by a factor of $2 \cdot \theta$ compared to Chor PIR [Cho+95], while the Direct Request schemes induce no record processing. As the use of an anonymity system raises the privacy level, the security parameter can be lowered to reach the same privacy level of the schemes at the cost of network delays. The Sparse-PIR methods do not influence the communication cost, but the Direct Request schemes drastically increase it as the number of requests p is a multiple of d.

The Subset-PIR optimization schemes helps scalability by reducing all costs by a factor of $\frac{t}{d}$, but turns ε-private protocols into (ε, δ)-private ones.

The two main approaches for decreasing the computation are contacting fewer databases and accessing (or processing) fewer records per server. It can be noted, for example, that in order for Sparse-PIR to achieve a similar level of computation to Subset-PIR with a given t, the parameter θ must be particularly low, $\theta = \frac{t}{4 \cdot d}$. The first approach would be relevant in the case of a quasi-trusted database system while the second in the case of a large untrusted one.

In Figure 6, we compare the computation cost C_p, the number of records accessed, and the communication cost C_s, the number of records sent, of the Direct Request and Sparse-PIR schemes, and their compositions with an anonymity system, for a system comparable to Certificate Transparency when the adversary controls half of the databases. If the costs of the designs with an anonymity system first appear greater than in the simple case, this can be explained by the lack of tightness of the bound in the Composition Lemma. The gains of the anonymity system can be seen by the values ε takes under the lines “DR” and “SP” which represent the last security value respectively for the Direct Request and Sparse-PIR designs without an anonymity system.

	ε	δ	C_c	C_s	C_p
Chor PIR [Cho+95]	0	0	$d \cdot n$	d	$\frac{1}{2} \cdot d \cdot n \cdot (c_{acc} + c_{prc})$
Direct Requests	$\ln\left(\frac{1}{d-d_a} \cdot \left(d \cdot \frac{n-1}{p-1} - d_a\right)\right)$	0	$\min(d \cdot n,\, p\lceil \lg n \rceil)$	p	$p \cdot c_{acc}$
Sparse-PIR	$4 \cdot \operatorname{arctanh}[(1-2\theta)^{(d-d_a)}]$	0	$d \cdot \min(n,\, \theta \cdot n\lceil \lg n \rceil)$	d	$\theta \cdot d \cdot n \cdot (c_{acc} + c_{prc})$
$\mathcal{AS}$-Request	$\ln\left(\frac{1}{u}\left(\frac{d}{d-d_a} \cdot \frac{n-1}{p-1} - \frac{d_a}{d-d_a}\right)^2 + \frac{u-1}{u}\right)$	0	$\min(d \cdot n,\, p\lceil \lg n \rceil)$	p	$p \cdot c_{acc}$
$\mathcal{AS}$-Sparse-PIR	$\ln\left(\frac{1}{u}\left(\frac{1+(1-2\theta)^{(d-d_a)}}{1-(1-2\theta)^{(d-d_a)}}\right)^4 + \frac{u-1}{u}\right)$	0	$d \cdot \min(n,\, \theta \cdot n\lceil \lg n \rceil)$	d	$\theta \cdot d \cdot n \cdot (c_{acc} + c_{prc})$
Subset-PIR	0	$\prod_{i=0}^{t} \frac{d_a - i}{d - i}$	$t \cdot n$	t	$\frac{1}{2} \cdot t \cdot n \cdot (c_{acc} + c_{prc})$

Table 1: Security and Cost Summary of the Schemes. The client communication C_c is measured in bits, while the server communication C_s is measured in units of b-bit records.

In Figures 6a and 6c, we show the privacy parameter ε as a function of the whole database system computation cost C_p and compare it between the two PIR designs and their composition with an anonymity system. For the Direct Request cases, C_p represents the total number of records accessed p while for Sparse-PIR ones this is the sum of the records accessed by each database $\theta \cdot d \cdot n$. This difference is worth mentioning as by definition a record can be accessed and sent only once in the Direct Request cases, while in the Sparse-PIR ones, a record can be accessed and processed at different servers. Thus, the privacy level will converge to 0 for $p = d$ with the Direct Request protocols but for $\theta = \frac{1}{2}$, or $p = \frac{1}{2} \cdot d \cdot n$ in the graphs, with the Sparse-PIR protocols. While both figures show ε decreasing with C_p, the Direct Request protocols perform better for a given C_p than the Sparse-PIR ones which however appear more flexible as the security parameter ε can be selected in a wider interval.

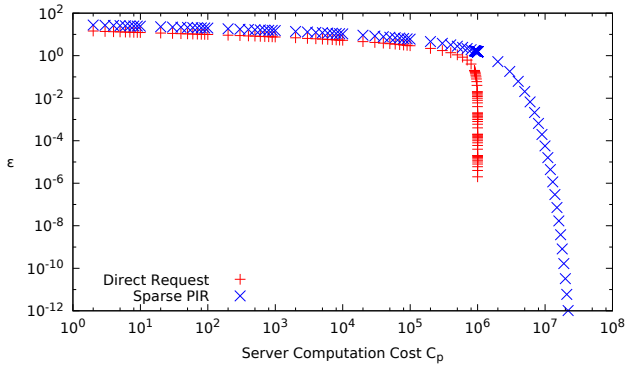
In Figures 6b and 6d, we show ε as a function of the number of records sent back by the whole database system to the user and compare it between the PIR designs and their compositions with an anonymity system. While the privacy level does not depend on C_s for the Sparse-PIR protocols, as the number of requests sent and record received is a constant, C_s has to greatly increase to reach an adequate ε in the Direct Request cases.

While the Direct Requests protocols present lower computational costs than the Sparse-PIR ones, they vastly increase the communication costs. This is not a surprise as PIR was conceived in order to limit the communication cost of private search in public databases. Choosing which method to use thus depends on the database system characteristics, not only the number of database servers and the level of trust the user has, but also the hardware. One method can be used to counter the system bottleneck, Sparse-PIR would suit servers with fast processors while Direct Request would adapt better with high-speed networks. As both processing and networking capabilities are continually increasing, the question of whether Direct Request schemes have a future is still open.

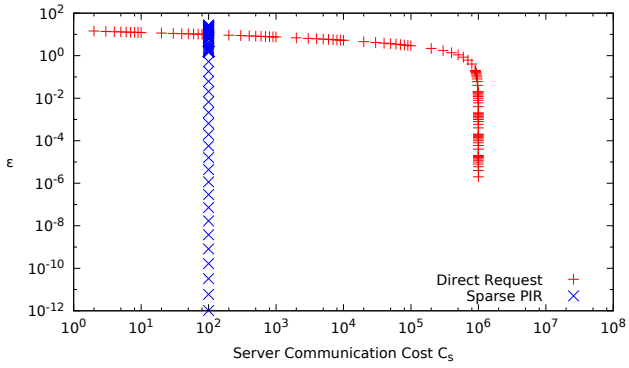
2.2.1 Discussion

2.2.2 Sybil Attacks

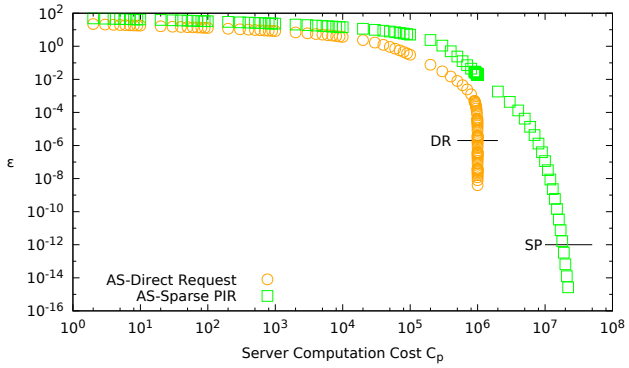
In a $n-1$, or Sybil, attack, an adversary acquires a disproportionately large influence on a system by creating a large number of identities. In this work, such an attack would translate when using the anonymity system to the adversary controlling or being most to all of the non-target clients. As a result, the number of honest users is drastically diminished, and so the adversary can guess with higher probability which queries are the target's. In the worst case when there are no honest non-target users, the system can be reduced to one not using an anonymity system (choosing $u - 1 = 0$ in the composition lemma leads to $\varepsilon_2 = 2 \cdot \varepsilon_1$ because of the bound



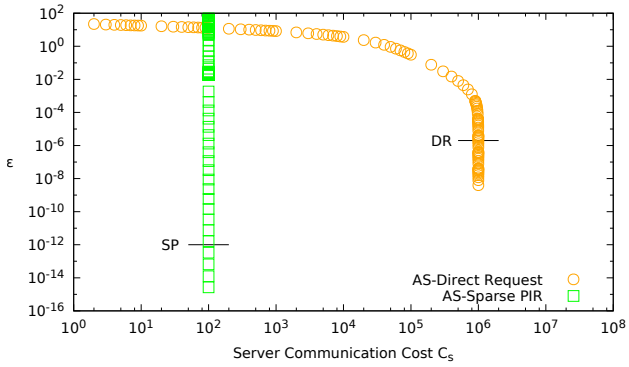
(a) C_p versus ε for Direct Request ($C_p = p$) and Sparse-PIR ($C_p = \theta \cdot d \cdot n$)



(b) C_s versus ε for Direct Request ($C_s = p$) and Sparse-PIR ($C_s = d$)



(c) C_p versus ε for AS-Direct Request ($C_p = p$) and AS-Sparse-PIR ($C_p = \theta \cdot d \cdot n$)



(d) C_s versus ε for AS-Direct Request ($C_s = p$) and AS-Sparse-PIR ($C_s = d$)

Figure 6: Parameterized plots for Direct Request and Sparse-PIR, AS-Direct Request, and AS-Sparse-PIR, for $d = 10^2$, $d_a = \frac{d}{2}$, $n = 10^6$, and $u = 10^3$. The dots in the figures show the varying parameter p (for the Direct Request schemes) or θ (for the Sparse-PIR schemes).

looseness).

Without using a central entity, solutions to counter Sybil attacks are limited, especially when maintaining anonymity. Usual techniques to counter Sybil attacks could be adapted, such as admission control by verifying external identifiers in order to submit requests, requiring a proof of work to increase the cost of Sybil attacks, or deploying social-graph-based Sybil detection.

2.3 Conclusions

We show that ε-private PIR can be instantiated by a number of systems, using dummy queries, anonymous channels, and variants of the classic Chor protocol. Yet some popular naive designs based on dummies or anonymous channels alone fail to provide even this weaker notion of privacy. We argue that the weaker protection provided by ε-private PIR may be sufficient to provide some privacy in systems that are so large in terms of database size, but also so popular, that current IT-PIR techniques are impossible to apply. With a large fraction of honest servers even weak (but still ε-private) variants of PIR, such as Sparse-PIR, provide near-perfect privacy. Showing that a system is ε-private enables smooth composition with an anonymity system, which guarantees that any anonymized ε-private PIR mechanism becomes near perfect given a large enough anonymity set.

2.4 Proofs of Theorems

2.4.1 Proof of Composing Naive Mechanisms

Proof. We want to prove in our indistinguishability game scenario that the probability the adversary observes exactly u queries Q_i is bounded above by $\delta_u \leq \left(\frac{p-1}{n-1}\right)^{u-1}$ while the probability they receive no Q_i queries is bounded above by $\delta_0 \leq \left(\frac{n-p}{n-1}\right)^{u-1}$. We first assume that the probability a user chooses one of the two queries (Q_i) given by the adversary is $\Pr_T$.

The probability a non-target user $\mathcal{U}_k$ selects this very Q_i out of his $p-1$ randomly selected requests (the p^{th} one being the adversarially provided non-target query Q_{n_k}) is $\frac{\binom{n-2}{p-2}}{\binom{n-1}{p-1}} = \frac{p-1}{n-1}$ as each record can only be requested once by any given user. As each user is independent, the probability all the users select Q_i is the product of the probabilities, we thus have $\delta_u = \Pr_T \left(\frac{p-1}{n-1}\right)^{u-1}$. Similarly, the probability a non-target user does not select this very Q_i out of his $p-1$ randomly selected requests is $\frac{\binom{n-2}{p-1}}{\binom{n-1}{p-1}} = \frac{n-p}{n-1}$. As each user is independent, the probability none of the users selects Q_i is the product of the probabilities, so $\delta_0 = \Pr_T \left(\frac{n-p}{n-1}\right)^{u-1} \leq \left(\frac{n-p}{n-1}\right)^{u-1}$, and similarly for δ_u. □

2.4.2 Proof of Security Theorem 1 (Direct Requests)

Proof. We want to prove the following result.

$$\begin{aligned}\mathcal{L} = \frac{\mathcal{P}_i}{\mathcal{P}_j} &= \frac{\Pr(Observation \mid Q_{Target} = Q_i)}{\Pr(Observation \mid Q_{Target} = Q_j)} \\ &\leq \frac{1}{d-d_a} \cdot \left(d \cdot \frac{n-1}{p-1} - d_a\right)\end{aligned}$$

We first note that the best observation for the adversary is to see exactly one of the adversarially provided target requests, for instance Q_i.

In the first case, the adversary supposes Q_i was sent. Q_j may also have been sent, but in this case a non-colluding database would have received it.

$$\begin{aligned}\mathcal{P}_i &= \frac{d_a}{d} \cdot \binom{n-1}{p-1}^{-1} \cdot \left[\binom{n-2}{p-1} + \frac{d-d_a}{d} \cdot \binom{n-2}{p-2}\right] \\ &= \frac{d_a}{d} \cdot \binom{n-1}{p-1}^{-1} \cdot \left[\binom{n-1}{p-1} - \frac{d_a}{d} \cdot \binom{n-2}{p-2}\right] \\ &= \frac{d_a}{d} \cdot \left[1 - \frac{d_a}{d} \cdot \frac{p-1}{n-1}\right]\end{aligned}$$

In the second case, the adversary supposes Q_j was sent however she only sees Q_i. Q_j must thus have been received by a non-colluding database.

$$\begin{aligned}\mathcal{P}_j &= \frac{d_a}{d} \cdot \frac{d-d_a}{d} \cdot \binom{n-2}{p-2} \cdot \binom{n-1}{p-1}^{-1} \\ &= \frac{d_a}{d} \cdot \frac{d-d_a}{d} \cdot \frac{p-1}{n-1}\end{aligned}$$

Therefore we obtain the result:

$$\begin{aligned}\mathcal{L} = \frac{\mathcal{P}_i}{\mathcal{P}_j} &\leq \frac{\frac{d_a}{d} \cdot \left[1 - \frac{d_a}{d} \cdot \frac{p-1}{n-1}\right]}{\frac{d_a}{d} \cdot \frac{d-d_a}{d} \cdot \frac{p-1}{n-1}} \\ &\leq \frac{d}{d-d_a} \cdot \frac{n-1}{p-1} - \frac{d_a}{d-d_a} \\ &\leq \frac{1}{d-d_a} \cdot \left(d \cdot \frac{n-1}{p-1} - d_a\right)\end{aligned}$$

This concludes the proof. □

2.4.3 Proof of Security Theorem 2 (Sparse-PIR)

Proof. We represent the p requests sent by the user by $\{0,1\}^{1\times n}$ vectors listed in a $d \times n$ matrix, each column representing a record and each row a request. The adversary $\mathcal{A}$ controlling only a set of the databases will only see some of the rows. $\mathcal{A}$ is interested in the number of ones in the columns, these numbers representing how many times each record has been requested.

We first note that the probability an (d, θ)-Binomial variable is even is $\frac{1}{2} + \frac{1}{2}(1-2\theta)^d$. [Sar11]

The adversary observes only the part of each column v_i corresponding to the corrupt servers d_a. We call the adversary observation for column i, o_i, and the hidden part of the vector h_i. Without loss of generality we consider that $v_i \leftarrow o_i|h_i$ namely that the column for entry i is the concatenation of the observed and the hidden part of the column.

We denote the event the user queried for record α as Q_α. For such a query our mechanism would set the column α, namely v_α, to have odd Hamming weight, and all other column $v_\beta, \beta \neq \alpha$ to have even Hamming weight.

To prove that the mechanism is differentially private we need to show that:

$$\frac{\Pr[\forall i.o_i|Q_\alpha]}{\Pr[\forall i.o_i|Q_\beta]} \leq e^\varepsilon$$

However, each column of the query is sampled independently of all others, and thus it suffices to prove that:

$$\frac{\prod_{\forall i.} \Pr[o_i|Q_\alpha]}{\prod_{\forall i.} \Pr[o_i|Q_\beta]} \leq e^\varepsilon$$

Since $\Pr[o_i|Q_\alpha]/\Pr[o_i|Q_\beta] = 1$ for $i \notin \{\alpha,\beta\}$, this expression simplifies to:

$$\frac{\Pr[o_\alpha|Q_\alpha] \cdot \Pr[o_\beta|Q_\alpha]}{\Pr[o_\alpha|Q_\beta] \cdot \Pr[o_\beta|Q_\beta]} \leq e^\varepsilon$$

We have the following cases depending on the observed parity of o_i, based on the expected parity of the full, and partly unobserved, v_i and v_j:

$$\begin{aligned}\Pr[o_i \text{ odd}|Q_i] &= \Pr[h_i \text{ even}] \\ \Pr[o_i \text{ even}|Q_i] &= \Pr[h_i \text{ odd}] = 1 - \Pr[h_i \text{ even}] \\ \Pr[o_j \text{ odd}|Q_i] &= \Pr[h_j \text{ odd}] = 1 - \Pr[h_j \text{ even}] \\ \Pr[o_j \text{ even}|Q_i] &= \Pr[h_j \text{ even}]\end{aligned}$$

For $\theta < 1/2$, it is the case that $\Pr[h_i \text{ even}] > \Pr[h_i \text{ odd}]$ and the differential privacy bound is minimized for:

$$\frac{\Pr[o_\alpha \text{ odd}|Q_\alpha] \cdot \Pr[o_\beta \text{ even}|Q_\alpha]}{\Pr[o_\alpha \text{ odd}|Q_\beta] \cdot \Pr[o_\beta \text{ even}|Q_\beta]} =$$
$$\frac{\Pr[h_\alpha \text{ even}] \cdot \Pr[h_\beta \text{ even}]}{\Pr[h_\alpha \text{ odd}] \cdot \Pr[h_\beta \text{ odd}]} =$$
$$\frac{\Pr[h_\alpha \text{ even}]^2}{\Pr[h_\alpha \text{ odd}]^2} =$$
$$\left(\frac{1/2+1/2(1-2\theta)^{|h_i|}}{1-(1/2+1/2(1-2\theta)^{|h_i|})}\right)^2 =$$
$$\left(\frac{1+(1-2\theta)^{|h_i|}}{1-(1-2\theta)^{|h_i|}}\right)^2$$

The value of ε such that this expression is bounded above by e^ε can be expressed in terms of an inverse hyperbolic tangent ($\operatorname{arctanh} x = \frac{1}{2}\ln\left(\frac{1+x}{1-x}\right); |x| < 1$):

$$\varepsilon = 4 \cdot \operatorname{arctanh}(1-2\theta)^{|h_i|}$$

This concludes the proof and the upper bound is tight. □

2.4.4 Proof of the Composition Lemma

Proof. We consider the observations $O_0 \dots O_{u-1}$ as originating from the ε_1-private PIR mechanism used by users $\mathcal{U}_0$ to $\mathcal{U}_{u-1}$ respectively. Without loss of generality we consider the target user $\mathcal{U}_t$ is $\mathcal{U}_0$. We try to determine a bound on the following quantity to prove ε-privacy:

$$\frac{\Pr(O_0 \dots O_{u-1}|Q_i, Q_{n_1} \dots Q_{n_{u-1}})}{\Pr(O_0 \dots O_{u-1}|Q_j, Q_{n_1} \dots Q_{n_{u-1}})} \leq e^{\varepsilon_2}$$

However, due to the use of the anonymity system the adversary has a uniform belief about the matching of all observations to all queries, out of the $u!$ possible matchings. Thus we have that:

$$\begin{aligned}\Pr(O_0 \dots O_{u-1}|Q_x, Q_{n_1} \dots Q_{n_{u-1}}) &= \\ &= \frac{1}{u!}\sum_{i=0}^{u-1}(u-1)!\Pr(O_i|Q_x)\prod_{j\neq i}\Pr(O_j|Q_{n_j}) \\ &= \frac{1}{u}\sum_{i=0}^{u-1}\Pr(O_i|Q_x)\prod_{j\neq i}\Pr(O_j|Q_{n_j})\end{aligned}$$

The quantity to be bounded can therefore be re-written as:

$$\frac{\frac{1}{u}\sum_{i=0}^{u-1}\Pr(O_i|Q_x)\prod_{j\neq i}\Pr(O_j|Q_{n_j})}{\frac{1}{u}\sum_{i=0}^{u-1}\Pr(O_i|Q_y)\prod_{j\neq i}\Pr(O_j|Q_{n_j})} =$$

$$\frac{\begin{array}{l}\Pr(O_0|Q_x)\prod_{j\neq 0}\Pr(O_j|Q_{n_j}) \\ \quad +\sum_{i\neq 0}\Pr(O_i|Q_x)\prod_{j\neq i}\Pr(O_j|Q_{n_j})\end{array}}{\sum_{i=0}^{u-1}\Pr(O_i|Q_y)\prod_{j\neq i}\Pr(O_j|Q_{n_j})}$$

We are now making a simplifying assumption: We consider that $\Pr(O_i|Q_z) = \mu$ if the observation O_i was indeed produced by the query Q_z, and ν otherwise, and also $\mu > \nu$. We rely on the law of large numbers for this

assumption to approximate reality, and it is not sensitive to adversary inputs. Since the PIR mechanism is ε-private we know that $\mu \leq e^{\varepsilon_1}\nu$. This simplifying assumption holds for large numbers of u, since products of multiple individual $\Pr(O_i|Q_z)$ will tend to be products of the average μ and ν.

The quantity to be bounded now reduces to:

$$\begin{aligned}
\frac{\mu^2\mu^{u-2} + (u-1)\nu^2\mu^{u-2}}{\nu^2\mu^{u-2} + (u-1)\nu^2\mu^{u-2}} = \\
\frac{\mu^2 + (u-1)\nu^2}{u\nu^2} = \\
\frac{\left(\frac{\mu}{\nu}\right)^2 + u - 1}{u} \leq \\
\frac{\left(e^{\varepsilon_1}\right)^2 + u - 1}{u} = \\
e^{\ln(e^{2\varepsilon_1}+u-1)-\ln u}
\end{aligned}$$

This concludes the proof. □

3 Blockmania

3.1 Related Work

'Blockchain' designs rely on Nakamoto consensus [Nak08], originating with Bitcoin, in which a proof-of-work puzzle determines a proposal for a block, and the 'most work' rule disambiguate between alternate chains. Alternative proposals consider instead mechanisms based on proof-of-stake [Dai98]: in such systems, such as Ouroboros [Kia+17], nodes are associated with some 'stake' value, which is unforgeable and transferable, and decisions on consensus are based on the amount of stake nodes possess. All those provide only probabilistic finality guarantees.

Fischer, Lynch and Paterson [FLP85] prove in 1985 that deterministic consensus is impossible under full network asynchrony and in the presence of even a single byzantine fault. Dwork et al [DLS88] introduce the partial synchrony model, and present a set of algorithms that achieve byzantine consensus within it. Castro and Liskov [CL+99b] introduce Practical Byzantine Fault Tolerance (PBFT) which is deterministic and require partial synchrony for liveness, but is safe under asynchrony. The most mature implementation is SmartBFT [BSA14], which advertises up to 50K transactions per second (tps). A mixture of those is used today by the tendermint [Kwo14; BKM18] smart contract platform. They report a throughput of 10K tps, with consensus reached within 1sec[1]. Cachin et al. [Cac+01] took the alternative approach and present a probabilistic byzantine consensus scheme based on secure shared randomness.

The worse-case communication cost of the standard PBFT, under repeated view-change, is $O(N^4)$ where N is the number of nodes in the quorum. This stems from the $O(N^2)$ size of the new-view messages (containing $2f+1$ prepares from each of the $2f+1$ nodes sending a new-change message, where $2f+1 \sim O(N)$); the message is sent to all $O(N)$ nodes; and up to $f \sim O(N)$ consecutive view changes may be needed before a view with an honest leader is reached. The Tendermint consensus [BKM18] has a worse case cost of $O(N^3)$, with cost $O(N^2)$ per round, and potentially $O(N)$ rounds with corrupt leaders until a decision is reached.

Abraham et al. [AGM18], introduce a linear view change (LVC) tweak to PBFT, to reduce the overall cost to $O(N^3)$, and further use threshold signatures to reach $O(N^2)$ including view change. Byzcoin [Kog+16], also employs collective signing to achieve $O(N)$ only for the steady-state protocol, excluding view change which remains $O(N^2)$ (each new-view includes $O(1)$ collectively signed commits, is broadcast to $O(N)$ nodes, and $O(N)$ need to be executed to reach a good leader).

[1] https://github.com/tendermint/tendermint/wiki/Introduction

Concurrently to our work LinBFT [Yan18] presents a variant of PBFT with $O(N)$ communication cost. They achieve this by combining the Linear View Change, from Hot Stuff [AGM18], a central point of distribution using Cosi [Kog+16] leading to a latency of $O(\log N)$, and a verifiable random function to determine the leader at every turn (the performance analysis is in terms of asymptotic costs and no specific throughput figures are reported). Byzcoin [Kog+16] achieves $O(N)$ communication complexity for the happy path, but not the view change. Neither of those protocols inherit the beneficial properties of separating the broadcast of block DAG from its interpretation as consensus.

A few systems interpret DAGs to get finality or consensus. Casper the Friendly Finality Gadget [BG17], provides conditions under which miners' votes lead to a finalized set of blocks. This system influenced Wendy [AGM18], which combines a block proposer with a finality layer based on an efficient PBFT variant. The Hashgraph protocol [Bai16] is closest to Blockmania in that it also uses a DAG describing transactions sent and received, and a subsequent interpretation step of 'virtual voting' to reach consensus without further communication—and identifies the saving in terms of communication complexity that result. Blockmania embeds a different interpretation logic, that clearly maps to a simplified PBFT protocol, and as a result does away with the need for electing "Famous witnesses" to facilitate consensus, shared randomness generation, and does not require novel consensus correctness arguments. However, the similarity of the approach highlights that different distributed algorithms can benefit from the paradigm of separating the block DAG creation from its interpretation as consensus.

3.2 Performance

How to measure communication cost. The FLP theorem [FLP85] establishes that consensus is impossible in a fully asynchronous setting, through a deterministic algorithm. Blockmania, like PBFT, ensures safety under asynchrony but requires partial synchrony [DLS88] for liveness. Therefore, communication costs are measured in the number of bytes that need to be exchanged per consensus decision reached, within a period of synchrony, since under asynchrony a byzantine network can delay decisions indefinitely. We also consider the amortized cost over multiple decisions.

Communication costs. Blockmania's communication complexity is $O(N^2)$, where N is the size of the quorum. Each block from a node is broadcast to all other $O(N)$ nodes. Further, all nodes include, at some point, a hash of all other blocks into their own chain, which results in blocks of average size $O(N)$ — resulting in an overall $O(N^2)$ communication cost.

Concretely, the communication cost for each block sealed includes the block meta-data (of fixed size), including a hash to the previous block (20 bytes for 80 bit security), and additionally on average a hash to all other blocks ($(N-1)\cdot 20$ bytes) as well as the size of all sealed transactions in the block. The overhead, for 31 nodes ($f = 10$) is $20 \cdot 31^2 \approx 19$ kbytes per block; for 1000 nodes ($f = 333$) the overhead is $20 \cdot 1000^2 \approx 19$ Mbytes.

It is worth reflecting on how we save $O(N^2)$ communication cost compared with the PBFT protocol (which has cost $O(N^4)$ under view change), without sacrificing any of its properties in terms of liveness or safety, including under view change. In traditional PBFT, nodes execute the protocol and directly exchange messages pertaining to commit, view-change and new-view. Those messages need to carry sufficient evidence to convince other nodes they are well formed: for commit and view-change this evidence is $O(N)$ and for new-view it is of size $O(N^2)$. In Blockmania such evidence does not need to be explicitly sent: each honest node interprets correctly the block DAG, which represents the pattern of message delivery to other nodes, and recreates the correct messages to drive the state of all other nodes in the network — it does not have to convince itself of the correctness of the interpreted messages — leading to the reductions in communication cost.

Constant factors in the amortized communication cost of Blockmania are also greatly reduced by using useful subsequent blocks, and interpreting them to conclude the consensus on previous blocks. Block exchanges for subsequent blocks are interpreted as prepare and commit, as well as view change and new view phases of the terminating reliable broadcast protocol — as well as phases of the protocol for subsequent blocks. This leads to the concrete overhead of only 20 bytes per node (for 80 bit cryptographic strength) for each block — which

is minimal. Given this small overhead communication complexity is $20 \cdot N^2 + \ell \cdot N + c$, where ℓ is the size of transactions in a block and framing, and c a small constant factor. Therefore for smaller quorum size N the cost is dominated by the volume of transactions in each block.

Concrete Performance Measurements. We implemented a networked prototype of Blockmania in `golang`, and measured its performance under different conditions. Each node generates multiple transactions, of 100 bytes length, that it includes in its own blocks emitted every second. Their volume is controlled by a bang-bang controller aiming to keep latency of consensus under 4 seconds. Operating 4 nodes on a single machine leads to a performance of over 1M transactions per second (tps).

When operating different nodes[2] within the same data center (Google cloud, London) the performance of Blockmania incurs actual TCP networking overheads. For 7 nodes ($f = 2$) Blockmania can process 650K tps, for 10 nodes ($f = 3$) 700K tps, for 13 nodes ($f = 4$) 680K tps, and for 16 nodes ($f = 5$) 640K tps. The differences are not statistically significant (and depends on the exact placement of virtual machines). We observe that the performance of Blockmania remains stable in terms of transactions per second: this is expected as the bottleneck is the capacity of the smallest node to received all blocks, which remains stable.

We also perform measurements across a wide area network on the Google Cloud platform. The nodes were distributed across Asia (E), North America (NW, W), Europe (W), Australia (SE), and South America (E). For 10 nodes ($f = 3$) we observe a performance of 430K tps, 13 nodes ($f = 4$) 440K tps, and 16 nodes ($f = 5$) 520K tps (the differences are not statistically significant across runs). This compares with an advertised 50K tps for SmartBFT, and 10K tps for Tendermint. We do not observe a significant drop in performance, as we increase the quorum size for small quorums, which validates that the concrete cost is dominated by the volume of transactions exchanged.

3.3 Theorems and their Proofs

Theorem 1 (DAG Availability 1)**.** *An honest node has received and stored the full block DAG, starting from any genesis blocks, to the last block it emits.*

Proof of Theorem 1

Proof. Since only valid blocks may be included as references into a block emitted, this implies that all their references have been received and stored. The argument applies inductively to all previous blocks, that must also be valid, up to the genesis blocks that contain no past references. □ □

Theorem 2 (DAG Availability 2)**.** *If an honest node emits a block, all honest nodes will eventually receive and consider valid this block, and all blocks referenced directly or indirectly by this block.*

Proof of Theorem 2

Proof. Since the honest node must have stored the full block DAG referenced by any block it emits, other honest nodes may request any blocks they are missing. Since the emitting node is honest and available they will eventually receive the missing blocks. Note this is irrespective of whether the blocks were created by honest or byzantine nodes. □ □

Theorem 3 (Terminating Reliable broadcast (safety))**.** *If two honest nodes reach a decision for a position (n, k), it will be the same decision, assuming at most f byzantine nodes.*

Theorem 4 (Terminating Reliable broadcast (liveness))**.** *A decision will eventually be reached for any position (n,k) by all honest nodes, assuming at most f byzantine nodes and partial synchrony. If n is honest, it will be for the block n prepared for position (n, k).*

[2] `n1-standard` instances (4 vCPUs, 15 GB memory, debian-9, with 100GB storage).

Proof of Theorems 3 & 4.

Proof. The proofs of safely and liveness follow closely the arguments for PBFT, as best detailed in [CL+99a].

The first pre-prepare in view 0, for an instance of the protocol (n, k) has to be signed by node n. The need for $2f$ other nodes to broadcast a *prepare* for any pre-preprepared block in a view ensures that only a single block in a view, if any, may ever result in a commit message by any honest node (in effect this is Bracha reliable broadcast [BT85]). If that occurs within the timeout delay, then it will result in this unique block being delivered for (n, k), after $2f + 1$ commits are received by all — ensuring safety.

The message *view-change* is triggered by nodes upon a logical or physical time-out, when no progress was made in time. It guarantees liveness, and importantly preserves safety. The *view-change* message contains evidence of the latest block committed, if any. The resulting view change, preserves safety by ensuring that if any node could have received $2t + 1$ commit messages for a block B in any past view, then the new-view message will also result in a new view with a pre-prepare for the same value B. On the other hand, if no node may have delivered a block in previous views, then the new view will *pre-prepare* 'nil'.

If fewer than $f + 1$ commit messages were emitted in a previous view, two competing new-view messages may appear in the next view, one for B and one for nil. In this case the protocol maintains safety, and is equivalent to a corrupt leader sending two contradictory pre-prepare messages: such a conflict may result in a new view where it will be resolved. Thus the protocol ensures safety and liveness without the need for a leader driving the new-view process. □ □

Theorem 5 (Safe Interpretation 1)**.** *Two honest nodes interpreting a block they both consider valid, using the same quorum of nodes, will interpret it in an identical manner. Namely, they will derive the same state for all state machines associated with the block, and the same set of messages.*

Proof of Theorem 5

Proof. As per Theorem 1, relating to validity, the two nodes will have all the blocks referenced by the block—forming the full DAG from the genesis blocks to this block. Through the collision resistance properties of secure hash functions these sub-graphs of the DAG will be identical in their content. Since the interpretations step is driven entirely by the order blocks are included in other blocks in this sub-graph of the DAG, and is otherwise deterministic, both honest nodes will execute identical steps and reach the same states for the state machines and messages associated with all valid blocks of the DAG. □ □

Theorem 6 (Safe Interpretation 2)**.** *Assuming at most f byzantine nodes and a quorum of $3f + 1$ nodes. Once an honest node interprets a state machine within any valid block as having reached a decision for a position, any other DAG of valid blocks interpreted by an honest node, with the same quorum, will eventually reach the same decision.*

Proof of Theorem 6

Proof. This theorem requires arguing that interpreting the block DAG is equivalent to nodes exchanging messages directly through the terminating reliable broadcast, and therefore the interpretation inherits its safety properties. Consider, without loss of generality an honest node interpreting the state machine relating to a decision for a specific position in a specific block. This honest node will interpret other *honest* node's blocks as messages; and those messages will be exactly the same as if the other nodes sent the terminating reliable broadcast messages directly (see Theorem 5). On the other hand, lets assume that the adversary has the ability to inject arbitrary messages into the honest node's interpretation for all blocks that created by *byzantine* nodes (as long as they are authenticated as being sent by one of the byzantine nodes). The result of this strictly more powerful adversary, would be for the state machine interpreted by the honest node to be driven by correct messages from $2f + 1$ honest nodes, and arbitrary messages from the f byzantine nodes. Therefore the state machine within

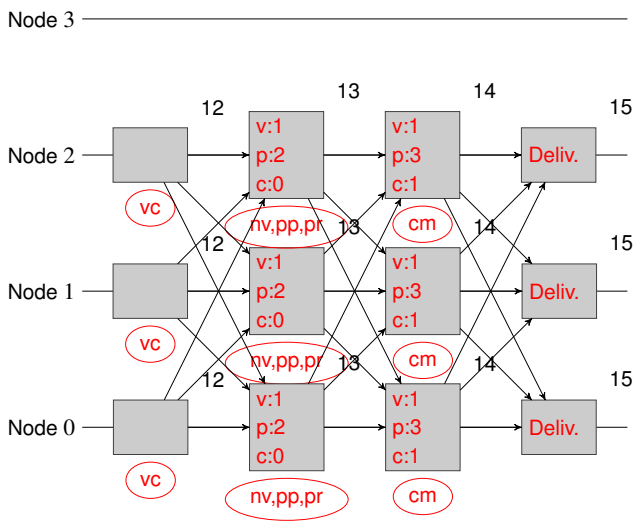


Figure 7: An illustration of a block DAG (black and gray), and its interpretation as a state machine per block for position $(n = 3, k = 2)$, when node 3 has failed and no block for this position has been broadcast. Each state machine for $(n = 3, k = 2)$ includes a view number (v), and a count for *prepare* (p) and *commit* (c) messages received by the block, and is illustrated within each block in red (along with a *Deliv.* delivery decision being reached). The *out* buffer for each block for this decision, is summarized in a red circle below the corresponding block: pp for *pre-prepare*, pr for *prepare* and cm for *commit*, vc for *view change* and nv for *new view*. None of the interpreted information (in red) is materialized as network communication.

the blocks of honest user n will follow the same states as if the terminating reliable broadcast was executed explicitly; and the state machines across blocks of two different honest users will therefore deliver the same block for this position (due to the safety properties of the terminating reliable broadcast) as long as there are at most f byzantine nodes. □ □

3.4 View change illustration

Figure 7 illustrates the interpretation of the block DAG in the case a block is missing, and has to be decided as 'nil' by honest nodes. In the example node 3 is unavailable, and nodes are called to decide on block $(n = 3, k = 2)$. They set their timers for view zero at their respective blocks $(n, 2)$, which timeout at their respective blocks $(n, 12)$ where the illustrated trace begins. The honest nodes 0 to 2, then initiate a *view-change* each that they broadcast. Upon those *view-change* messages reaching all nodes (in block round 13) they initiate a *new-view* for $v = 1$, that is also interpreted as a *pre-prepare* and *prepare* for the value 'nil' for block position $(3, 2)$.

The protocol then proceeds as in the steady case, with all nodes sharing the *prepare* messages (received at round 14), followed by the *commit* messages (received at round 15) for the decision $(3, 2) \rightarrow$ 'nil' to be agreed upon.

4 SybilQuorum

4.1 Background and Related work

The Sybil attack was introduced by Douceur [Dou02], in relation to engineering peer-to-peer systems, and identified types of defences: admission control through central authentication, and resource constraints. Permissioned ledgers, such as Hyperledger [And+18] or Quora, take the first approach, and only allow known and designated nodes to participate in consensus. Open distributed ledgers, including Bitcoin and Ethereum follow the second

paradigm. Proof of Work was first proposed by Back, as Hashcash [Bac02], to prevent Denial of Service. In the context of spam its economic efficiency was questioned by Clayton and Laurie [LC04].

Proof-of-stake systems were proposed first in the 90s by Wei Dai, in B-money [Dai98]. Modern proof-of-stake systems, such as Ouroboros [Kia+17] allow users to lock and delegate stake, and sample those users proportionately to their stake to determine an order in which blocks are produced in a blockchain system. Consensus therefore remains open, in that anyone who can buy some currency and lock it as stake can participate. However, there are serious concerns with this approach: the most fundamental one being that very wealthy parties may afford to acquire a lot of stake, and abuse it to extract value out of the system. Since stake often allows nodes to mine blocks, and reap rewards, the economics of proof-of-stake may lead to oligarchies through a "rich get richer" dynamic.

A number of blockchain systems consider some trust judgments between nodes and leverage them to achieve consensus. Stellar [Maz15] considers that each node links to other nodes, and uses these direct trust judgments to form quorums in which byzantine consensus may be run — this is the closest related system to SybilQuorum. We use the definition and security concepts introduced in Stellar, such as a Federated Byzantine Agreement System (FBAS) and a disposable set (DSet), as a basis for our security arguments and evaluation. However, Stellar only considers direct judgments to form an FBAS, rather than the topology of the full social network. Ripple [Arm+15] allows participants to connect to each other, but does not solve the Sybil defence problem directly, and does not achieve inter-node consensus in a strong manner. Instead, each node may run its own currency and economy, and rely on others' willingness to act as an exchange to transfer value between nodes that are not directly connected.

Besides blockchains, systems leveraging social networks — and explicit trust judgments of users about each other — have been proposed to combat Sybil attacks. Early work considers leveraging the 'introduction graph', by which nodes get to access a Distributed Hash Table through other nodes, to ensure routing security [Dan+05]. Raph Levien productized those ideas to extract reputation of developers in 'Advogato' [Lev09]; and Sam Lessin [Les18] proposed using trust graphs backed by financial commitments to infer the financial trustworthiness of users in a graph in the context of blockchains.

Academic works within this family of systems consider general social network information distributed in a peer to peer network to allow each node to determine which other nodes are genuine or Sybils. In this line of work SybilGuard [Yu+08] and SybilLimit [Yu+10] perform a distributed computation, using random walks in a network, to determine the honest regions within it. SybilInfer [DM09] takes a centralized approach, and analyzes a stored social graph to identify potential Sybil regions.

These defences make some security assumptions related to the topology of 'honest' social graphs: those need to be fast mixing, have small diameter, and contain relatively few links to nodes being part of a Sybil attack. Those systems allow each node to extract a degree of belief about whether any other node in the system is a genuine participant or a Sybil. However, this belief depends on the position of the node in the social graph, and may not be exactly the same even for two honest nodes — thus they do not directly lead to any form of consensus, not even about who is a Sybil node. Ultimately, each node uses those degrees of belief to define their own set of nodes considered honest.

Subsequent work questions a number of assumptions based on the analysis of real-world social graphs [Moh+12]. This work is influential in that it highlights that the social graphs on which these defences rest, but truly capture trust judgments, and provide incentives for users to not accept any links, including to malicious nodes. In this work we also highlight a further limit of SybilInfer as originally proposed: it is an effective mechanism to detect Sybil regions in the presence of an attack, however it is also presenting a large number of "false positives" when the network is free of such attacks — by misclasifying a large number of honest nodes as Sybils. We provide a solution to this problem.

Besides 'blockchain' based consensus, based on a chain of blocks and a fork choice rule, modern distributed ledgers consider and reimagine more traditional forms of byzantine consensus. An exemplary system is Tendermint [Kwo14], that combines a quorum based byzantine consensus protocol, with a proof-of-stake mechanism.

In Tenderming, and in general, decisions are made as part of the consensus protocol when over two-thirds of 'stake' supports a decision — abstracting from the actual identities of nodes and only considering their weight in stake. The advantages of this approach is low latency, quick finality, and higher throughput than Nakamoto consensus. This family of systems also includes Blockmania [DH18], which separates messages materialized and exchanged in a network forming a directed acyclic graph of blocks, from the process of nodes independently interpreting it to reach consensus and order transactions. This separation is key for the practical and efficient implementation of SybilQuorum.

4.2 The Security of an FBAS

Both proposed variants of SybilQuorum define a Federated Byzantine Agreement System, as defined in [Maz15]. We therefore use some further definitions to achieve two security properties: (1) safety means that two honest nodes will agree to the same outcome of the consensus; and (2) liveness ensures that progress towards reaching consensus may be made despite some byzantine nodes.

Definition 1 (Quorum Intersection)**.** *An FBAS enjoys* quorum intersection *iff any two of its quorums share a node—i.e., for all quorums U_1 and U_2, $U_1 \cap U_2 \neq \varnothing$. (From [Maz15])*

Definition 2 (Delete)**.** *If $\langle V, Q\rangle$ is an FBAS and $B \subseteq V$ is a set of nodes, then to delete B from $\langle V, Q\rangle$, written $\langle V, Q\rangle^B$, means to compute the modified FBAS $\langle V \setminus B, Q^B\rangle$ where $Q^B(v) = \{q \setminus B | q \in Q(v)\}$. (From [Maz15])*

Definition 3 (DSet)**.** *Let $\langle V, Q\rangle$ be an FBAS and $B \subseteq V$ be a set of nodes. We say B is a dispensible set, or* DSet*, iff:*

1. *(quorum intersection despite B) $\langle V, Q\rangle^B$ enjoys quorum intersection.*
2. *(quorum availability despite B) Either $V \setminus B$ is a quorum in $\langle V, Q\rangle$ or $B = V$.*

(From [Maz15])

The concept of 'dispensible set' (DSet) in an FBAS is key to understanding its security. The DSet contains a set of nodes that can act adversarially, without jeopardizing the safely and liveness properties of the FBAS for the remaining (honest) nodes. The 'quorum intersection despite B' property ensures safety, since it requires any two quorums within a system without B nodes to intersect, and thus agree on the same result. The 'liveness despite B property' ensures the set of honest nodes in the FBAS can form a consensus to agree on a result, even if the nodes in B do not participate.

In our experiments, to establish the security of SybilQuorum as an FBAS system, we will need to determine the necessary DSet, containing byzantine nodes, as well as potentially some honest nodes for whom Sybil defences failed. However, establishing the 'quorum intersection despite B' property of a DSet not easy: naively it would require computing all quorums and testing their pairwise intersection—which is computationally unfeasible for larger number of nodes. Therefore we devise two algorithms to determine the DSet is a quorum (property 2) and also to check quorum intersection despite the DSet (property 1), efficiently. The correctness of those algorithms depends on original theorems related to an FBAS, which may also be of independent interest.

We first prove a lemma, on which we rely for the correctness of our algorithm.

Lemma 1. *Consider a node $v \in V$ in an FBAS $\langle V, Q\rangle$, and a quorum U, such that $v \in U$, and a quorum slice $q \in Q(v)$ for v contained in U, namely $q \in U$. If another node v' is in the same slice, namely $v' \in q$, and the minimum cardinally of any quorum that contains v' is h—i.e. for all quorum U', such that $v' \in U'$, $h \leq |U'|$. Then the minimum cardinally of U is also h—i.e. $h \leq |U|$.*

Proof. The theorem seems complex, but really is the result of a simple symmetry: since the quorum slice q is contained in U, both $v \in q$ and $v' \in q$ are within the quorum, $\{v, v'\} \subseteq U$. Since h is the minimal cardinality of a quorum containing v', and the quorum U also contains v', it trivially follows that $h \leq U$. □

Definition 4 (Quorum Slice Cardinality Map)**.** *The function $C(v)$ is the* quorum slice cardinality Map *for an an FBAS $\langle V, Q\rangle$. For each node $v \in V$ in , it returns the cardinality of the smallest quorum slice in $Q(v)$—i.e. $C(v) = \min\{|q| \,|\, q \in Q(v)\}$.*

Security Theorem 1 (Trivial Intersection)**.** *For an FBAS, if all values of the quorum slice cardinality map are larger than half the number of nodes, it enjoys quorum intersection—i.e.* $\forall v \in V, C(v) > |V|/2$.

Proof. Consider two nodes and quorums in the FBAS, $v_1 \in U_1$ and $v_2 \in U_2$. By the definition of quorums there must exist two slices $q_1 \in Q(v_1), q_2 \in Q(v_2)$ of v_1, v_2 respectively such that $q_1 \subseteq U_1$ and $q_2 \subseteq U_2$. Since $|q_1| > |V|/2$ and $|q_2| > |V|/2$, it must be that they intersect at least in one element. And $q_1 \cup q_2 \neq \varnothing \Leftrightarrow q_1 \cap U_1 \cap q_2 \cap U_2 \neq \varnothing \Rightarrow U_1 \cap U2 \neq \varnothing$. □

This first theorem provides a trivial way to check a FBAS for quorum intersection: if all quorum slices, for all nodes contain more than half the nodes, then all quorums will intersect. However, this condition is much stronger than necessary for quorum intersection, and in practice not always achievable. However, it is the property on which the security of traditional BFT systems can be proven in when those are encoded as FBAS. However, we will seek a weaker property that still implies quorum intersection.

Lemma 2 (Minimum Quorum Cardinality in FBAS)**.** *Consider the FBAS* $\langle V, Q \rangle$*. We define a function* $F_i(v)$ *providing a lower bound on the cardinality of any quorum* U *containing* v*, namely* $\forall U, i$ *such that* $v \in U$ *it holds that* $F_i(v) \leq |U|$*. We initialize* F *as* $F_0(v) = C(v)$*, where* C *is the quorum slice cardinality map. We also define the sets* $\bar{q}_v \subseteq$ *for each* $v \in V$*, containing all nodes in* v*'s quorum slices—i.e.* $\bar{q}_v = \bigcup_{q \in Q(v)} q$*.*

Define as S *the sequence of values* $[F_i(v') \,|\, v' \in \bar{q}_v]$ *in ascending order, and* $S_i[C(v)]$ *is its* $C(v)^{th}$ *element. If we assign* $F_{i+1}(v) \leftarrow \max\{S_i[C(v)], F_i(v)\}$*, the value* $F_{i+1}(v)$ *is also a lower bound on the cardinally of any quorum containing* v*.*

Proof. By the definition of the quorum slice cardinality map C it is trivial to argue that $F_0(v) = C(v)$ is a lower bound on the cardinality of all quorums including v, since they each need to fully contain at least one quorum slice from v. We need to show that the value $F_i(v)$ and therefor $S_i[C(v)]$ is a lower bound on the cardinality of any quorum U containing v. Since the minimum quorum slice size of v is $C(v)$ it must contain at least that number of nodes, out of the set $\bar{q}_v$. By lemma 1 we know that including a node v' from $\bar{q}$ into a quorum U, would yield a quorum of cardinality at least $F_i(v')$. Since $C[v]$ such nodes from $\bar{q}_v$ must be included the minimum cardinality of U is $S_i[C(v)]$, since S_i is defines as the ordered sequence of minimum cardinality sizes for quorums each node in $\bar{q}_v$. □

We use this lemma in building an algorithm that computes lower bounds on the cardinalities quorums of all nodes in the FBAS iterativelly. It starts with an estimation equal to the cardinality of the smallest quorum slice for each node (the function L(v)), and then increases the estimate as $F_{i+1}(v) \leftarrow \max(F_i(v), S_i[L(v)])$, where $S_i = [F_i(v') \,|\, v' \in \bar{q}_v]$.

Based on the above we can efficiently estimate minimum bounds on the cardinality of all quorums in the SybilQuorum FBAS. Then we can test them to show quorum intersection:

Security Theorem 2 (Quorum Intersection due to Minimum Size)**.** *If within a FBAS, all quorums* U *have cardinality* $|U| > |V|/2$*, it enjoys quorum intersection.*

Proof. Trivially, if we have two sets $U_1 \subseteq V$ and $U_2 \subseteq V$ with a number of elements greater than half the number of elements in V, they must intersect in at least one element. □

We note that Quorum Intersection due to Minimum Size is a sufficient condition to guarantee a FBAS enjoys quorum intersection, but it is too strong to be necessary. For example an FBAS with a dictator node v_0 present in all quorums, will satisfy trivially quorum intersection, without the need for all quorums to be of a certain size.

Computing DSets and FBAS safety. We leverage the theorems above to test the concrete FBAS extracted from SybilQuorum for safety and liveness. We define a set of nodes V, each with a list $H(v) \subseteq V$ of nodes they consider honest, and a set of malicious nodes B. The quorum function $Q(v)$ contains all subsets of $H(v)$ of size over $2/3|H(v)|$.

function DETERMINEDSET($\langle V, Q\rangle, B$)
$H(v) \leftarrow \bigcup_{q \in Q(v)} q$
exit $\leftarrow$ False **while *not* *exit* do**
exit $\leftarrow$ True **forall** $V \setminus B$ **do**
$|H(v) \setminus B| > 2 \cdot |H(v) \cap B|$
$B \leftarrow B \cup \{v\}$
exit $\leftarrow$ False
return B
end function

function DETERMINESAFETY($\langle V, Q\rangle$)
$H(v) \leftarrow \bigcup_{q \in Q(v)} q$
$C(v) \leftarrow \min\{|q| \text{ for } q \in Q(v)\}$
$i \leftarrow 0$
$F_i(v) \leftarrow C(v)$
exit $\leftarrow$ False **while *not* *exit* do**
exit $\leftarrow$ True **forall** $V \setminus B$ **do**
$S \leftarrow \text{sorted}([F_i(v') \text{ for } v' \in H(v)])$
$F_{i+1}(v) \leftarrow \max\{F_i(v), S_i[C(v)]\}$ **if** $F_{i+1}(v) \neq F_i(v)$ **then**
exit $\leftarrow$ False
$i \leftarrow i + 1$
return $\forall v \in V.F_i(v) > |V|/2$
end function

Figure 8: Algorithms to determine safe set and quorum intersection

First we execute a procedure DETERMINEDSET using the initial bad nodes B to determine a set B' of nodes that cannot reach agreement, due to having accepted too many bad nodes as honest. Following the terminology from Stellar we call the set of nodes $B' \setminus B$ befouled nodes. A node is befouled if it has accepted in its set $H(v)$ more than a third of bad or befouled nodes. By definition the nodes $V \setminus B'$ still constitute a quorum. (Thus proving liveness despite B').

Once we have identified the set of bad and befouled nodes B', we define the FBAS $\langle V, Q\rangle^{B'}$ and try to establish whether it enjoys quorum intersection. We use algorithm DETERMINESAFETY to test for quorum intersection: we iteratively determine an increasingly better lower bound $F_i(v)$ on the quorum cardinality of each node. Once the bound converges, we check that each $F_i(v) > |V \setminus B'|/2$, which according to our theorems ensures quorum intersection (Thus proving quorum intersection despite B'). If this condition is true we label our FBAS as safe, since the set B' is a DSet.

4.3 Experimental Evaluation

Datasets & Pre-processing. An evaluation of SybilQuorum necessitates the use of real-world datasets of social connections, since the mechanism relies on the dynamics of connections within 'real' social networks. There are methodological challenges to doing this. First, there does not exist a network embodying the proposed mechanism of establishing links backed by mutual token on links as necessary by SybilQuorum-hybrid. Second, existing datasets are based on networks for casual socializing, in which incentives are not aligned for careful selection of links, but rather provide advantages and incentives for users to be prosmicuous in their connections. Those issues present threats to validity.

For our evaluation we chose to use the *pokec* network dataset, that is open and available on the Stanford large network dataset collection[3]. This is a snapshop of the largest social network provider in Slovakia, collected in 2012. It contains 1632803 nodes and 30622564 edges.

We pre-process this network in two ways, to produce evaluation datasets: (1) We sub-sample 200000 nodes from the network, and create a subgraph with all their edges (including those to nodes not in the set of nodes);

[3] https://snap.stanford.edu/data/soc-Pokec.html

(2) We then recursively prune the network to a core of nodes with degree at least 3. Pruning is performed by removing nodes with degree less than 3, until all nodes have a higher degree.

These operations result in sub-graphs of size about 10000 nodes, which is comparable to the number of miners in systems such as Bitcoin and Ethereum. We extract the degree three core of the network as a proxy for nodes that have strong connections to each other, excluding nodes with weaker trust connections between them. (Note such pre-processing can also done as part of a production SybilQuorum pipeline.)

Sybil Attack Simulation. To evaluate SybilQuorum we simulate Sybil attacks on the graph datasets, in a what that is most generic. We parametrize the attack through a number of parameters: (1) the number of Sybil nodes (n_s); (2) the number of links or amount of stake on links purely in the Sybil region (l_s); (3) the amount of stake on links between the honest region and Sybil region (l_n); (4) the fraction of honest nodes that are naive, and connect to Sybil nodes (f_n).

Given those parameters, we instanciate a set of Sybil nodes of size n_s, and establish l_s mutual links between them at random. We sample at random a set of honest nodes to be 'naive', as a fraction f_n of the honest nodes. We then create mutual connections between random Sybil nodes in that set, and honest nodes from the naive set, according to the budget of links or stake available (l_n).

SybilInfer [DM09] provides an argument that the exact composition of the Sybil region does not impact security, but what matters is rather the relative size of the Sybil region and the links between honest and Sybil regions. However, we there might be optimizations in connecting Sybils in specific ways to the naive nodes, which we have not explored. This is a further threat to the validity of our results. However, our methodology is in line with previous works.

Purely Benign or Byzantine Conditions. We first evaluate the SybilQuorum mechanism in a network composed of overwhelmingly benign nodes. We instanciate such a network by only attaching a single Sybil ($n_{=}1$), no stake in the Sybil region ($l_s = 0$), and minimal stake between the honest and dishonest nodes ($l_n = 2$). On the other hand we allow this single Sybil nodes to connect to any honest node ($f_n = 1.0$). We primarily use this condition to ensure that SybilQuorum does not suffer from false positives in detecting Sybils, that compromise agreement, as the raw SybilInfer mechanism does.

We also evaluate SybilQuorum under conditions of byzantine attacks that can be accommodated within the traditional Byzantine fault tolerance paradigm, with just a standard proof-of-stake system: where the number of Sybils n_s is $1/3$ of the size of the honest nodes, and the stake of all links connected to the Sybil nodes is at most $1/2$ of the total stake in the honest region. We also allow Sybil nodes to connect to any honest node ($f_n = 1.0$). This condition simulates an attack that can be tolerated even if all Sybil nodes are accepted as honest by all – but we need to assess whether this is the case in SybilQuorum and whether the resulting FBAS is secure.

In both conditions SybilQuorum honest nodes reach safe agreement. In the benign condition the cut-off is determined as $y = 0.49$, and all nodes are accepted by all other honest nodes as honest. This includes the single Sybil node. All quorums are larger than half the number of honest nodes, and global agreement is reached. In the byzantine condition agreement is also reached. The cut-off value is set automatically as $y = 0.50$. And since all honest nodes have fewer than 1/3 links to Sybil nodes (none is confused) they reach agreement. (Those are the results of 10 repeats of the experiments, for different configurations of the Sybil attack).

Those results confirm that SybilQuorum performs no worse than not using SybilQuorum— which is not a given: (1) when there is no Sybil attack it does not impede agreement through false positives; and (2) when there are fewer than $1/3$ dishonest nodes, the FBAS is secure and preserves agreement and liveness without any negative interference from SybilQuorum.

5 Analysis of Internet censorship measurements in Cyprus

This section presents the analysis of the crowd sourced measurements of Internet censorship in Cyprus. For a description of the experiment setup, and the particular geopolitical case of Cyprus, please refer to sections 5.2 and 5.3 of deliverable D2.4.

5.1 Methodology for data collection and analysis

We are using a variety of common free and open source software networking tools for gathering, categorizing, distributing, analyzing data and comparing the results. Acquiring results from a number of different ISPs is crucial to form a representative sample. We have conducted network measurements and used publicly available data based on OONI reports [OON] submitted by volunteers. We were able to collect and process network measurement data from the following residential landlines and cellular ASes (Autonomous Systems): AS15805 (MTN Cyprus Limited), AS24672 (CallSat International Telecommunications Ltd.), AS35432 (Cablenet Communication Systems Ltd.), AS6866 (Cyprus Telecommunications Authority), AS8544 (Primetel) and AS197792 (Multimax Iletisim Limited). Even so, this remains a limited sample and the findings presented here are tentative and preliminary.

5.2 Data Set Used for the Tests

First, we compiled a list of all URLs that are reported to be blocked in Cyprus as published and curated by the RoC NBA [Cypa], the Greek gamling authority's blocklist [Com] the Lumen database [IHU] for Turkey, and the community-collected global test list maintained by Citizenlab[LO14]. Additionally, we have used the public open DNS servers list provided by Digineo GmbH [Gmb].

5.3 Collection of Network Measurements

The collection of the network measurements took place during the months of March to May 2017, though we were able to process relevant data submitted by volunteers from the months January and February earlier in 2017. Volunteers collected and submitted network measurements by using a custom set of tools and test lists [Obs17] populated from the data sets enumerated in section 5.5 in deliverable D2.4. For our censorship research we used ooniprobe, an application developed by the OONI project [FA12] and used by volunteers and organizations to probe their network for signs of network tampering, surveillance or censorship. Developed with the idea of ensuring the detection of any interference to network communications, it aims to collect and provide high quality reports by using open and transparent data methodologies freely available to anyone that would like to process and analyze.

Ooniprobe is the application that was used to conduct the measurements on the ISP networks (both landline and cellular networks) where we detected network tampering and content blocking. Ooniprobe provides a variety of test cases and classes that could be used to probe the networks. More analytically, in our research we have deployed and analyzed a number of network measurements tests, precisely instant messaging, HTTP header fields manipulation and invalid request line tests, Tor and pluggable transports reachability tests as well as the web connectivity test. We were not able to identify any certain case of network interference in all of the tests apart from the web connectivity test. However this does not necessarily mean that there is no other sort of network interference happening on the network during different date periods or from different vantage points.

Web connectivity is an ooniprobe test methodology where we were able to identify and detect if a website is reachable and the reason or cause in case a website is not reachable. This test reaches a non censored control measurement endpoint (test helper) to assist with the comparison of the measurements for a given website. At first, the test performs an *A* DNS lookup to a special domain name service in our experiments;

whoami.akamai.com that will respond to the *A* DNS lookup request with the resolver of the probe. Upon DNS resolver identification, the test will perform a DNS lookup querying the *A* record of the default resolver for the hostname of the URL tested. Following the test will try to establish a TCP session on port 80 or port 443 if the URL in question begins with the prefix *http* or *https* accordingly for the list of all IPs returned by the previous DNS query. Finally, the test performs a HTTP GET request for the path specified in the uniform resource identifier using the most widely used web browser user agent; *Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/47.0.2526.106 Safari/537.36* [Sta] as the HTTP header. Upon completion of the test, the gathered data are compared with the ones of the control measurement test helper.

6 Preliminary Findings

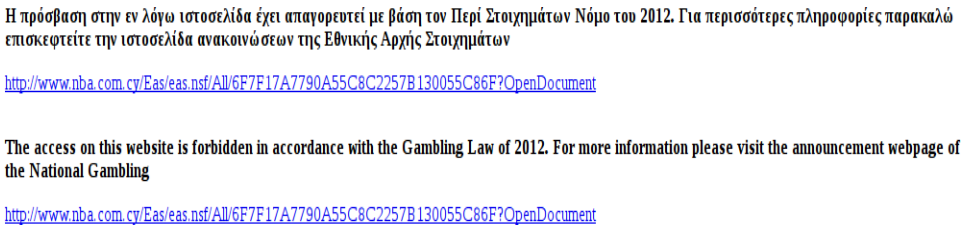


Figure 9: Callsat ISP NBA regulation landing page

We were able to perform measurements on the following ISPs: Cytanet (AS 6866), Cablenet (AS 35432) and Multimax (AS 197792). Additionally, we were able to identify block pages based on reports contributed by volunteers to the OONI data repository [OON] on ISPs Callsat (AS 24672) and MTN (AS AS15805).

The most common identified method of content blocking on Cypriot ISPs is DNS hijacking. Since ISPs are in control of the DNS servers used by their users in residential broadband or cellular connections, they can manipulate the DNS servers' responses and can redirect the requesting users to anywhere they want. Taking advantage of this privilege, ISPs modify their resolvers to override censored domains' legitimate DNS replies by creating local zone entries [AA]. These entries usually point to a server that they control where they run a web server that displays a webpage with the warning message to users or block page.

6.1 Differences between ISPs

All ISPs, with the exception of Multimax in the north of the island, were using DNS hijacking as the blocking to control the access of the entries in NBA's list. Comparing the network measurements from all ISPs we found multiple cases of websites (entries of the blocklist) not being blocked, providing error messages (specifically HTTP status codes 403 and 404) or were unable to connect (connection failed) to HTTPS entries instead of the blocking page or the reason (legislation) why a user cannot access the specific website in question. Additionally, we were able to detect instances where email communication to the specific websites was also blocked although the law does not imply blocking email communication but only restricting access to the website that is included in the blocklist.

Additionally, at least one ISP was found redirecting the user to the website of NBA [Cypb], leaking the IP addresses and possible the web browser's specific user metadata.

6.2 Callsat ISP

Network measurements analyzed from Callsat ISP [Cal] (AS 24672) on the entries of the NBA blocklist revealed an outdated *landing* block page with a URL that points to a non-existent web resource (HTTP status code 404). The blockpage is illustrated in Figure 9.

6.3 Cablenet ISP

Our findings from the network measurements reveal that the Cablenet ISP [Cab] (AS 35432) was directing users trying to access the entries of the blocklist to a generic error webpage (HTTP status code 403) without providing any justification of the blocking. The user may falsely assume that the website in question experiences technical issues. The blockpage is illustrated in Figure 10.

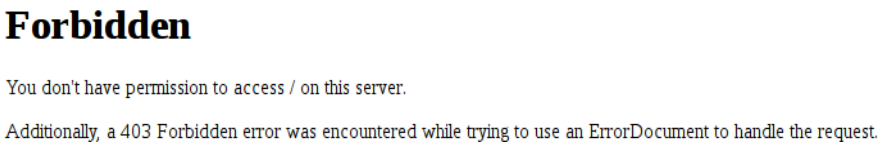


Figure 10: Cablenet ISP NBA regulation landing page

6.4 Cyta ISP

Cyta ISP [Cyt] (AS 6866) does not point the users to a blockpage but rather redirects the users trying to access the blocked entries from the blocklist to the NBA website. The excerpt from the HTML markup code is illustrated in Listing 1.

```
<!doctype html>
<html class="no-js">
<head>

<meta http-equiv="content-type" content="text/html; charset=utf-8">
<meta name="copyright" content="copyright 2013">
<meta name="author" content="Designed & Developed by Cyta">
<meta name="distribution" content="global">
<meta http-equiv="refresh" content="0; url=http://www.nba.gov.cy/" />

</head>
</html>
```

Listing 1: Cyta ISP's HTML markup landing page

6.5 MTN ISP

Network measurements collected from MTN ISP [MTN] (AS 15805) on one day (02/04/2017) show no evidence of blocking.

https://www.wikipedia.org
https://www.torproject.org
http://www.islamdoor.com
http://www.fepproject.org
http://www.no-ip.com
https://wikileaks.org
https://psiphon.ca

Table 2: Multimax ISP: List of blocked websites

6.6 Multimax ISP

Multimax [Mul] (AS 197792) is one of the ISPs that operates in the north of Cyprus. We have not identified any block pages, however, upon closer analysis, we found many similarities to the Turkish ISPs and specifically the blocking of web resources using IP blocking. We can conclude that the websites in Table 2 have not been accessible for the period of time during our network measurements. Note that the list of websites in Table 2 is not exhaustive and there could more websites or service that may be potentially blocked by this ISP.

6.7 Collateral damage

```
; <<>> DiG 9.9.5-9+deb8u10-Debian <<>> MX williamhill.com @82.102.93.140
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 19519
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1280
;; QUESTION SECTION:
;williamhill.com.        IN   MX

;; AUTHORITY SECTION:
williamhill.com.     3600     IN   SOA ns1.cablenet-as.net. noc.wavespeed.net.
1483803163 10800 3600 604800 3600

;; Query time: 97 msec
;; SERVER: 82.102.93.140#53(82.102.93.140)
;; WHEN: Tue Apr 04 02:35:07 CEST 2017
;; MSG SIZE  rcvd: 113
```

Listing 2: Empty (no answer) DNS MX records for williamhill.com (dig output)

In our research we identified that the Mail Exchange (MX) records are absent, and do not contain the relevant DNS records that point to the email server of the domain name in question. That is rendering email delivery to the specific domain name impossible.

In the Listing 2 we have requested the MX records of the domain name *williamhill.com* from the DNS server *82.102.93.140* (DNS resolver in Cyprus operated by Cablenet) compared to the Google's DNS resolver *8.8.8.8* as illustrated in Listing 3. Google's DNS resolver answered with 2 entries in the query (*ANSWER: 2* for the domain name in question whereas Cablenet's DNS resolver sent no answers (*ANSWER: 0*. The DNS queries took place on 4 April, 2017.

```
; <<>> DiG 9.9.5-9+deb8u10-Debian <<>> MX williamhill.com @8.8.8.8
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 16093
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;williamhill.com.                IN      MX

;; ANSWER SECTION:
williamhill.com.        605     IN      MX      10
mxb-0010e301.gslb.pphosted.com.
williamhill.com.        605     IN      MX      10
mxa-0010e301.gslb.pphosted.com.

;; Query time: 40 msec
;; SERVER: 8.8.8.8#53(8.8.8.8)
;; WHEN: Tue Apr 04 02:43:51 CEST 2017
;; MSG SIZE  rcvd: 116
```

Listing 3: Google DNS MX records for williamhill.com (dig output)

6.8 Circumventing blocking

Using a block-list/allow-list model (sort of an ON/OFF model) is not granular enough and tends to fail; it will have negative impact on users and customers of the ISPs that may or may not find routes around the blocking. Furthermore, creating such an ineffective blocking gives a false sense of security as the entities that enforced such blocking would assume that the content is being blocked although blocking is easily circumvented.

6.9 Using alternative DNS resolver

Circumventing the blocking enforced by the ISPs is just a tweak in the network configuration and requires no technical expertise by using a different DNS resolver such as Google DNS [Dev] (*8.8.8.8*) or OpenDNS [Inc] (*208.67.222.222*).

References

[AA] D. Atkins and R. Austein. *Threat analysis of the domain name system (DNS)*. URL: `http://web.archive.org/web/20140826081656/http://www.ietf.org/rfc/rfc3833.txt` (visited on 08/26/2014).

[AGM18] Ittai Abraham, Guy Gueta, and Dahlia Malkhi. “Hot-Stuff the Linear, Optimal-Resilience, One-Message BFT Devil”. In: *arXiv preprint arXiv:1803.05069* (2018).

[And+18] Elli Androulaki et al. “Hyperledger fabric: a distributed operating system for permissioned blockchains”. In: *Proceedings of the Thirteenth EuroSys Conference, EuroSys 2018, Porto, Portugal, April 23-26, 2018*. 2018, 30:1–30:15.

[Arm+15] Frederik Armknecht et al. “Ripple: Overview and outlook”. In: *International Conference on Trust and Trustworthy Computing*. Springer. 2015, pp. 163–180.

[Bac02] Adam Back. *Hashcash-a denial of service counter-measure*. 2002.

[Bai16] Leemon Baird. *Hashgraph consensus: fair, fast, byzantine fault tolerance*. Tech. rep. Swirlds Tech Report, 2016.

[BG17] Vitalik Buterin and Virgil Griffith. "Casper the friendly finality gadget". In: *arXiv preprint arXiv:1710.09437* (2017).

[BKM18] Ethan Buchman, Jae Kwon, and Zarko Milosevic. "The latest gossip on BFT consensus". In: *arXiv preprint arXiv:1807.04938* (2018).

[BSA14] Alysson Bessani, João Sousa, and Eduardo EP Alchieri. "State machine replication for the masses with BFT-SMaRt". In: *Dependable Systems and Networks (DSN), 2014 44th Annual IEEE/IFIP International Conference on*. IEEE. 2014, pp. 355–362.

[BT85] Gabriel Bracha and Sam Toueg. "Asynchronous consensus and broadcast protocols". In: *Journal of the ACM (JACM)* 32.4 (1985), pp. 824–840.

[Cab] Cablenet. *Cablenet ISP official website*. URL: http://archive.is/UBxqc (visited on 06/05/2017).

[Cac+01] Christian Cachin et al. "Secure and efficient asynchronous broadcast protocols". In: *Annual International Cryptology Conference*. Springer. 2001, pp. 524–541.

[Cal] Callsat. *Callsat ISP official website*. URL: http://archive.is/CAuFL (visited on 06/04/2015).

[Cho+95] Benny Chor et al. "Private Information Retrieval". In: *Presented at the 36th Annual IEEE Symposium on Foundations of Computer Science* (1995).

[CL+99a] Miguel Castro, Barbara Liskov, et al. *A Correctness proof for a practical byzantine-fault-tolerant replication algorithm*. Tech. rep. Technical Memo MIT/LCS/TM-590, MIT Laboratory for Computer Science, 1999.

[CL+99b] Miguel Castro, Barbara Liskov, et al. "Practical Byzantine fault tolerance". In: *OSDI*. Vol. 99. 1999, pp. 173–186.

[Com] EEEP Greek Gaming Commision. URL: https://www.gamingcommission.gov.gr/images/Anakoinoseis/BlackListVersion4_11072014.pdf (visited on 07/11/2015).

[Cypa] National Betting Authority of Cyprus. *National Betting Authority of Cyprus Blocklist website*. URL: https://web.archive.org/web/20170605133936/http://blocking.nba.com.cy (visited on 06/05/2015).

[Cypb] National Betting Authority of Cyprus. *National Betting Authority of Cyprus official website*. URL: https://web.archive.org/web/20170605132348/http://nba.gov.cy (visited on 06/05/2015).

[Cyt] Cyta. *Cyta ISP official website*. URL: http://archive.is/NBDpH (visited on 06/05/2017).

[Dai98] Wei Dai. "b-money, 1998". In: *URL: http://www. weidai. com/bmoney. txt (visited on 10/12/2017)* (1998).

[Dan+05] George Danezis et al. "Sybil-Resistant DHT Routing". In: *Computer Security - ESORICS 2005, 10th European Symposium on Research in Computer Security, Milan, Italy, September 12-14, 2005, Proceedings*. 2005, pp. 305–318.

[Dev] Google Developers. *Using Google Public DNS*. URL: http://web.archive.org/web/20140829223153/https://developers.google.com/speed/public-dns/docs/using (visited on 08/29/2014).

[DH18] George Danezis and David Hrycyszyn. "Blockmania: from Block DAGs to Consensus". In: *arXiv preprint arXiv:1809.01620* (2018).

[DHS14] Daniel Demmler, Amir Herzberg, and Thomas Schneider. "RAID-PIR: Practical Multi-Server PIR". In: *6th ACM Workshop on Cloud Computing Security (CCSW)*. 2014, pp. 45–56.

[DLS88] Cynthia Dwork, Nancy Lynch, and Larry Stockmeyer. "Consensus in the presence of partial synchrony". In: *Journal of the ACM (JACM)* 35.2 (1988), pp. 288–323.

[DM09] George Danezis and Prateek Mittal. "SybilInfer: Detecting Sybil Nodes using Social Networks". In: *Proceedings of the Network and Distributed System Security Symposium, NDSS 2009, San Diego, California, USA, 8th February - 11th February 2009*. 2009.

[Dou02] John R. Douceur. "The Sybil Attack". In: *Peer-to-Peer Systems, First International Workshop, IPTPS 2002, Cambridge, MA, USA, March 7-8, 2002, Revised Papers*. 2002, pp. 251–260.

[FA12] Arturo Filastò and Jacob Appelbaum. "OONI: Open Observatory of Network Interference". In: *Free and Open Communications on the Internet*. USENIX, 2012. URL: https://www.usenix.org/system/files/conference/foci12/foci12-final12.pdf.

[FLP85] Michael J Fischer, Nancy A Lynch, and Michael S Paterson. “Impossibility of distributed consensus with one faulty process”. In: *Journal of the ACM (JACM)* 32.2 (1985), pp. 374–382.

[Gmb] Digineo GmbH. *Public DNS Server List*. URL: https://web.archive.org/web/20170606195759/https://public-dns.info/ (visited on 06/06/2017).

[Gol07] Ian Goldberg. “Improving the Robustness of Private Information Retrieval”. In: *28th IEEE Symposium on Security and Privacy*. 2007, pp. 131–148.

[IHU] Berkman Klein Center for Internet and Society at Harvard University. *Lumen*. URL: http://archive.is/BwyBv (visited on 06/05/2015).

[Inc] OpenDNS Inc. *OpenDNS IP Addresses*. URL: http://web.archive.org/web/20140829223158/http://www.opendns.com/opendns-ip-addresses/ (visited on 08/29/2014).

[Kia+17] Aggelos Kiayias et al. “Ouroboros: A provably secure proof-of-stake blockchain protocol”. In: *Annual International Cryptology Conference*. Springer. 2017, pp. 357–388.

[Kog+16] Eleftherios Kokoris Kogias et al. “Enhancing bitcoin security and performance with strong consistency via collective signing”. In: *25th USENIX Security Symposium (USENIX Security 16)*. 2016, pp. 279–296.

[Kwo14] Jae Kwon. “Tendermint: Consensus without mining”. In: *Draft v. 0.6, fall* (2014).

[LC04] Ben Laurie and Richard Clayton. “Proof-of-work proves not to work; version 0.2”. In: *Workshop on Economics and Information, Security*. 2004.

[Les18] Sam Lessin. “Venmo Trust and The Blockchain”. In: *https://www.theinformation.com/articles/venmo-trust-and-the-blockchain* (2018).

[Lev09] Raph Levien. “Attack-resistant trust metrics”. In: *Computing with Social Trust*. Springer, 2009, pp. 121–132.

[LO14] Citizen Lab and Others. *URL testing lists intended for discovering website censorship*. https://github.com/citizenlab listshttps://github.com/citizenlab/test-lists. 2014. URL: https://github.com/citizenlab/test-lists.

[Maz15] David Mazieres. “The stellar consensus protocol: A federated model for internet-level consensus”. In: *Stellar Development Foundation* (2015).

[Moh+12] Abedelaziz Mohaisen et al. “On the mixing time of directed social graphs and security implications”. In: *7th ACM Symposium on Information, Compuer and Communications Security, ASIACCS ’12, Seoul, Korea, May 2-4, 2012*. 2012, pp. 36–37.

[MTN] MTN. *MTN ISP official website*. URL: http://web.archive.org/web/20170605020143/http://www.mtn.com.cy/ (visited on 06/05/2017).

[Mul] Multimax. *Multimax ISP official website*. URL: http://web.archive.org/web/20170605015656/http://www.mmcyp.com (visited on 06/05/2017).

[Nak08] Satoshi Nakamoto. “Bitcoin: A peer-to-peer electronic cash system”. In: (2008).

[Obs17] Hack66 Observatory. *A custom set of tools to perform ooniprobe network measurements*. https://github.com/hack6 2017. URL: https://github.com/hack66/bet2512.

[OON] OONI. *OONI measurements files repository*. URL: https://web.archive.org/web/20170606210652/https://measurements.ooni.torproject.org/ (visited on 06/06/2017).

[Sar11] Dilip Sarwate. http://math.stackexchange.com/questions/82841/probability-that-a-n-frac12-binomial-random-variable-is-even. 2011.

[Sta] Statcounter. *StatCounter GlobalStats*. URL: http://gs.statcounter.com/ (visited on 06/05/2015).

[Yan18] Yin Yang. “LinBFT: Linear-Communication Byzantine Fault Tolerance for Public Blockchains”. In: *arXiv preprint arXiv:1807.01829* (2018).

[Yu+08] Haifeng Yu et al. “SybilGuard: defending against sybil attacks via social networks”. In: *IEEE/ACM Trans. Netw.* 16.3 (2008), pp. 576–589.

[Yu+10] Haifeng Yu et al. “SybilLimit: A Near-Optimal Social Network Defense Against Sybil Attacks”. In: *IEEE/ACM Trans. Netw.* 18.3 (2010), pp. 885–898.