Federated Flow-Based Approach for Privacy Preserving Connectivity Tracking

Mentari Djatmiko
NICTA & UNSW
Sydney, Australia
mentari.djatmiko@nicta.com.au

Dominik Schatzmann
ETH Zurich
Zurich, Switzerland
schadomi@gmail.com

Xenofontas Dimitropoulos
ETH Zurich
Zurich, Switzerland
fontas@tik.ee.ethz.ch

Arik Friedman
NICTA
Sydney, Australia
arik.friedman@nicta.com.au

Roksana Boreli
NICTA & UNSW
Sydney, Australia
roksana.boreli@nicta.com.au

ABSTRACT

Network outages are an important issue for Internet Service Providers (ISPs) and, more generally, online service providers, as they can result in major financial losses and negatively impact relationships with their customers. Troubleshooting network outages is a complex and time-consuming process. Network administrators are overwhelmed with large volumes of monitoring data and are limited to using very basic tools for debugging, e.g., ping and traceroute. Intelligent correlation of measurements from different Internet locations is very useful for analyzing the root cause of outages. However, correlating measurements of user traffic across domains is largely avoided as it raises privacy concerns. A possible solution is secure multi-party computation (MPC), a set of cryptographic methods that enable a number of parties to aggregate data in a privacy-preserving manner. In this work, we describe a novel system that helps diagnose network outages by correlating passive measurements from multiple ISPs in a privacy-preserving manner. We first show how MPC can be used to compute the scope (local, global, or semi-global) and severity (number of affected hosts) of network outages. To meet near-real-time monitoring guarantees, we then present an efficient protocol for MPC multiset union that uses counting Bloom filters (CBF) to drastically accelerate MPC comparison operations. Finally, we demonstrate the utility of our scheme using real-world traffic measurements from a national ISP and we discuss the trade-offs of the CBF-based computation.

Categories and Subject Descriptors

C.2.3 [**Network Operations**]: Network Monitoring

Keywords

Network monitoring; outages; troubleshooting; secure multi-party computation; privacy preservation

1. INTRODUCTION

Internet outages affect Internet Service Providers (ISPs), as well as a huge number of industry players, who provide services and applications that rely on uninterrupted online connectivity for conducting their business. For ISPs, such outages may have a direct impact on service level agreements (SLAs) with their business customers, or an indirect but nevertheless important impact on their reputation for service quality for consumers. Outages may also have a significant impact on online service providers. For example, according to their reported earnings, *Amazon* may have lost about 4.9 million USD when its website was unreachable for approximately one hour in January 2013[1]. The outages may be caused by a number of different events, ranging from targeted attacks like prefix highjacking, routing issues due to misconfiguration, to equipment malfunction, power failures and problems with physical transmission media, e.g., fiber cuts. Detecting and troubleshooting outages in a timely manner is essential for the successful operations of all these businesses.

Current troubleshooting tools, including a number of industry solutions (e.g., from Cisco[2]), rely heavily on closed (single) domain network information [2]. As the automated tools require processing of private data (e.g., customers' traffic, if passive measurement tools are utilized) and commercially sensitive data (e.g., information about the network quality of an ISP or an enterprise), aggregation of such data from multiple domains is not a current practice. Yet it appears quite obvious that there are a number of potential benefits that may be achieved with such aggregation, as demonstrated by researchers [9]. Root cause analysis is an integral part of troubleshooting outages, which requires determining

[1]`http://www.networkworld.com/news/2013/013113-amazoncom-suffers-outage-nearly-5m-266314.html`
[2]`http://www.cisco.com/en/US/docs/net_mgmt/assurance_manager/mpls/1.0/user/guide/cod.pdf`

the scope of the outage, that is, whether an outage is local to a domain or affects multiple domains (a global outage), and the number of clients that are impacted. To assist with troubleshooting outages by comparing connectivity information from a number of different vantage points, network operators currently utilize mailing lists (e.g., NANOG [25] and Outages [26]) to request assistance from peers. Other tools and services used for monitoring include external monitoring systems and customer support, which directly liaise with the affected customers, and subsequently direct the initial information to technical support for resolution. Needless to say, the latter is ineffective as a means of enabling fast reactivity to service disruptions.

To enable correlating private data (both private for the ISP's business confidentiality and the customer's data point of view), it is important to include a privacy preserving mechanism within the outage detection system. To illustrate this, consider the following scenario. A network operator notices an outage towards a destination. To troubleshoot the outage, the operator would typically ask if other operators experience a problem with the same destination in mailing lists, such as NANOG. However, if the destination hosts controversial content, the operator might not want to disclose that they experience a problem to this destination and therefore the operator would not be able to receive help in troubleshooting the outage. Recent advances in privacy preserving computations have enabled real-time aggregation of large scale network data [7] using secure multi-party computation (MPC) [11]. MPC is a set of cryptographic methods that enable multiple parties to jointly compute a function without revealing their inputs. In other words, MPC enables a number of parties to jointly provide the functionality of a trusted third party, without having to trust anyone of the parties.

In this paper, we build on the SEPIA MPC framework [8] and propose a privacy preserving approach for collaborative connectivity tracking that aggregates passive network traffic measurements across multiple ISPs to assist with analyzing the root cause of outages. Our contributions are as follows. First, we propose Multi-FACT, a novel system for detecting the scope and severity of network outages by aggregating traffic data from multiple domains, while preserving the privacy of the source data. Multi-FACT builds on earlier work on the Flow-based Approach for Connectivity Tracking (FACT) [31]. It enables multi-domain collaboration, while keeping confidential all associations between domains and the destinations towards which they experience outages. Second, to enable efficient (near real-time) outage detection, we present an efficient method for multiset union operations using counting Bloom filters (CBF). We demonstrate the benefits of the proposed system through experimental evaluation based on real (live) network datasets from a Swiss research and academic network. We show that it is possible to achieve up to 97% reduction on the outages that need to be investigated, thereby enabling more productive use of the operators' technical resources. To aid with practical system design considerations, we further analyze the estimation accuracy of the CBF and the trade-offs introduced by the CBF-based computations.

We start by discussing related work on network outage monitoring systems and privacy preserving aggregation for networking applications in Section 2. In Section 3, we discuss two important prior works that our system design relies on, namely FACT and MPC. Our proposed system is presented in Section 4, including the details of the efficient privacy preserving aggregation scheme based on MPC. Section 5 demonstrates the utility of our proposal in real world events. We present the experimental evaluation of our system in terms of the protocol runtime and result accuracy in Section 6. Finally, in Section 7 we conclude the paper.

2. RELATED WORK

The related work falls into two categories: 1) network outage monitoring approaches and 2) privacy preserving aggregation.

2.1 Network outage monitoring systems

Existing approaches to monitoring Internet reachability problems can be classified based on three criteria: the monitoring approach, the system architecture and the specific goals of a system.

The monitoring approach can be active, passive or hybrid. Active measurement systems (e.g., PlanetSeer [36]) continuously probe remote destinations using tools such as traceroute and ping, and typically introduce network overhead. Passive measurements may be collected either from the control or the data plane. Control plane passive measurement systems typically use BGP data (e.g., from RouteViews [30] or RIPE RIS [27]). However, these approaches typically rely on anomaly detection techniques for discovering, e.g., prefix hijacking, and are prone to false positives. Passive measurements from the data plane leverage user traffic, like in FACT [31] and Crowdsourcing Event Monitoring [9]. To exploit the benefits of active and passive measurements, some systems combine both approaches. Examples of hybrid systems include the approaches proposed in [37], [22] and [33]. In hybrid systems, passive measurements are used to determine the remote destinations (usually BGP prefixes) that are affected by an outage, while active measurements are used to verify and localize the outages. Besides, the collaborative monitoring platform PerfSonar [19, 35] includes a number of both active and passive measurement tools to troubleshoot network performance problems. However, PerfSonar shares non-sensitive network monitoring data, such as traceroute or SNMP measurements, and does not provide privacy-preserving computation mechanisms for analyzing more sensitive data, which is the focus of this work. Furthermore, there are also emerging techniques that use aggregated information from multiple (non-network) sources, including social stream analysis of user complaints [2, 15].

The monitoring approaches can also be categorized according to the system architecture. We refer to entities that perform network monitoring as *monitors* and the process to determine outages as the *decision making process*. Some monitoring systems, such as Hubble [22] and Argus [33], use distributed monitors and centralized decision making processes. In other words, they employ monitors that are distributed across the Internet, but the decision making is performed by a centralized entity. Monitoring systems using this architecture detect Internet-wide (or global) outages, which may be irrelevant for some domains. In contrast, some monitoring system, such as FACT [31], only monitor the traffic of a domain (i.e., ISP or AS) and the decision making process is performed by the same domain. While this means that only relevant outages are detected, the system cannot determine whether the outage is limited to a domain or is more widespread. Other monitoring systems (e.g., Crowdsourcing Event Monitoring [9]) use distributed monitors, and the decision making process is performed by each monitor individually based on its data and the data from other end-user systems. The clear advantage of this architecture is that the system is capable of detecting global outages and determining which ones are relevant to the domain.

While the main goal of all the monitoring approaches is the same, that is, to detect outages, some approaches are also designed to achieve additional goals, such as troubleshooting or remediation. For example, Hubble [22] attempts troubleshooting outages by locating or narrowing down the potential locations of outages. Meanwhile, Lifeguard [23] redirects traffic away from outages.

Our approach, Multi-FACT, uses passive data-plane measurements. However, unlike FACT, Multi-FACT correlates data from multiple monitors. The decision making process is run by each domain individually, but it exploits information from other domains to diagnose local and global outages. In our recent work [14], we provide a tutorial-nature introduction to leveraging MPC for troubleshooting outages with Multi-FACT.

2.2 Privacy preserving aggregation

Several privacy preserving aggregation mechanisms for networking applications have been proposed. We present some of the mechanisms below.

In [29], Roughan and Zhang propose privacy preserving mechanisms for joint network performance computation. In particular, the work focuses on traffic summation and Internet anomaly detection. It proposes a secure summation protocol using secret sharing. (We discuss secret sharing in more details in Section 3.2.) The secure summation protocol is also used to compute average delay in [28]. Furthermore, the work discusses the use of sketch [10], a small-space data structure that allows an approximate reconstruction of the value associated with any given key, to compress the input data. Anomaly detection is performed by applying a time-series forecast model to the sketch of the input data. These works use a similar privacy mechanism to ours, the main difference with respect to our proposal is the application to passive outage monitoring and the validation of the mechanism on an experimental platform with real ISP data.

Applebaum *et al.* [1] propose a privacy preserving aggregation mechanism for a large number of participants for generic networking applications. The work focuses on the aggregation mechanism to compute the intersection of the participants' input data. For efficiency reasons, the mechanism uses a semi-centralized architecture (as opposed to the fully distributed architecture used in our system), which also allows clients to asynchronously provide inputs without requiring complex scheduling. It includes (as a minimum) a proxy and a database. The proxy obliviously blinds client inputs and transmits the blinded inputs to the database, while the database builds a table that is indexed by the blinded key and performs the aggregation. The mechanism utilizes several cryptographic protocols (such as oblivious pseudorandom functions) and it can be extended to include multiple proxies and databases. Compared to our solution, this work relies on both a different architecture and a different privacy mechanism. Furthermore, this work is not directly applicable to the problem (network outage monitoring) addressed with our proposal.

3. BACKGROUND

We provide an overview of the privacy preserving computation mechanism used in this paper and the passive outage detection approach we use as a basis for Multi-FACT.

3.1 Passive Monitoring with FACT

A number of recent studies have identified passive network traffic measurements as a new, promising data source for detecting and measuring the impact of network outages [31, 17, 12]. These approaches analyze network traffic over time to identify changes in the traffic signal observed from different networks. Compared to the traditional approach of running active measurements, they leverage an existing rich data source, i.e., the innate traffic of a network, without introducing additional probing. In this work we exploit in particular the tool introduced in the Flow-based Approach for Connectivity Tracking (FACT) [31], which is based on passive flow-level traffic measurements.

FACT captures unsampled network traffic data using NetFlow from the border routers of a network, like an ISP or an enterprise network. It extracts traffic between local and remote hosts and performs bi-flow pairing. A two-way flow suggests a successful connection and refers to outgoing traffic (as perceived by the border routers), matched with the corresponding incoming traffic. In particular, the outgoing and incoming traffic include the same 5-tuple (e.g. IP addresses, port numbers, and protocol number) with reverse values in the source and destination fields. On the other hand, the presence of outgoing traffic without a corresponding incoming traffic flow is classified as an one-way flow, which indicates an unsuccessful connection with a remote destination.

Network outages are identified by a rapid increase in the number of unsuccessful connections (one-way flows) and a simultaneous decline in the number of successful connections (two-way flows) to a remote destination. FACT classifies networks with only one-way flows as a potential network outage. Networks that produce a mixture of one-way and two-way flows are considered operational. FACT can also detect outages at different levels of granularity, i.e., a remote destination may refer to an IP address, prefix or AS. Furthermore, this approach only detects outages that affect the clients of a monitored network, which helps to prioritize outages by the number of affected local clients.

3.2 Secure multi-party computation (MPC)

Our privacy-preserving aggregation scheme is based on secure multi-party computation (MPC) [11]. Given input data from multiple parties, MPC is a cryptographic protocol that enables distributed parties to jointly compute functions, while providing formal guarantees on the confidentiality of the input data and on the correctness of the computation result.

The most popular MPC method uses Shamir secret sharing (SSS) (e.g., in [4]). The SSS scheme [32] is a well-known secret sharing scheme based on polynomial interpolation. Given a secret D, the (k, n) SSS scheme generates n shares from D and distributes them between n entities. D is only computable given the knowledge of at least k shares (where $k < n$), and the knowledge of $k - 1$ shares does not reveal any information about D. SSS is suitable for MPC since it has a homomorphism property, where the results of certain types of computations performed on the shares match the results of the computations performed on the actual data. The SSS scheme guarantees the confidentiality of the input data and the correctness of the results. Provided that the polynomial in SSS is of rank $t = \lfloor \frac{n-1}{2} \rfloor$, in the honest-but-curious adversary model (which is the adversary model used in this work) MPC is secure up to $t < \frac{n}{2}$ colluding entities who, jointly, would not be able to obtain any information about either the secret input data or the intermediate results of the computations [4].

In our MPC scheme, input peers provide the input data, and privacy peers perform the secure computation on this data. An entity participating in MPC may have either a single role (i.e., as input peer or privacy peer) or both roles. In summary, the input peers use SSS to secretely distribute their input data to the privacy peers. The privacy peers then perform the computation on the shares and return the final computation results to the input peers. As the privacy peers only receive the shares of the input data and SSS guarantees that a privacy peer (on its own) cannot recover the input data, there is no requirement for a trust relationship between the input and privacy peers. The number of input peers directly affects the computational complexity of MPC, while the number of privacy peers affects both the computational complexity and the resilience of the MPC mechanism against collusion attacks. Generally, MPC's resilience to attacks scales linearly with the number of privacy peers. Hence, the selection of the number of privacy peers reflects a complexity-resilience trade-off.

For many years, the research field of MPC had been almost exclusively of theoretical interest. The majority of the works focused on improving the security of MPC under more sophisticated attack models (e.g., [21]) and reducing the computational complexity of different operations (e.g., [5]). While the real-world use of MPC is limited due to the computational overhead, there are a number of on-going efforts to exploit MPC for real-world problems. For example, a real-world MPC application was demonstrated on January 2009, when MPC was used to facilitate an actual multi-party sugar beet auction in Denmark [6]. An example MPC application in networking is for privacy-enhanced inter-domain route computation [18, 20]. Furthermore, several efficient MPC frameworks, such as SEPIA [8], VIFF [13] and FairPlayMP [3], have been recently introduced.

We use SEPIA [8], an efficient MPC library based on SSS, to evaluate our proposed mechanism (in Section 6). SEPIA is written in Java, is optimized for parallel execution, and is specifically designed to aggregate network events and statistics from multiple domains. In SEPIA, SSL is used to secure the communications among the peers. SEPIA uses the semi-honest (i.e., honest-but-curious) adversary model, where the adversaries (semi-honest privacy peers) conform to the protocol, but use the information received and may collude to find the secret input data.

We choose MPC as the specific privacy preserving mechanism (as opposed to selecting, for example, differential privacy [16]), as our goal is to perform a privacy-preserving computation on sensitive input data without using a trusted third party. In contrast, differential privacy is useful for publishing an obfuscated version of the sensitive data, which is not relevant for our goal. The input peers only share the output of our MPC computation, which is a counting Bloom filter that encodes aggregate information about the number of domains (or hosts) that cannot reach different destinations. Therefore, the output of the computation does not

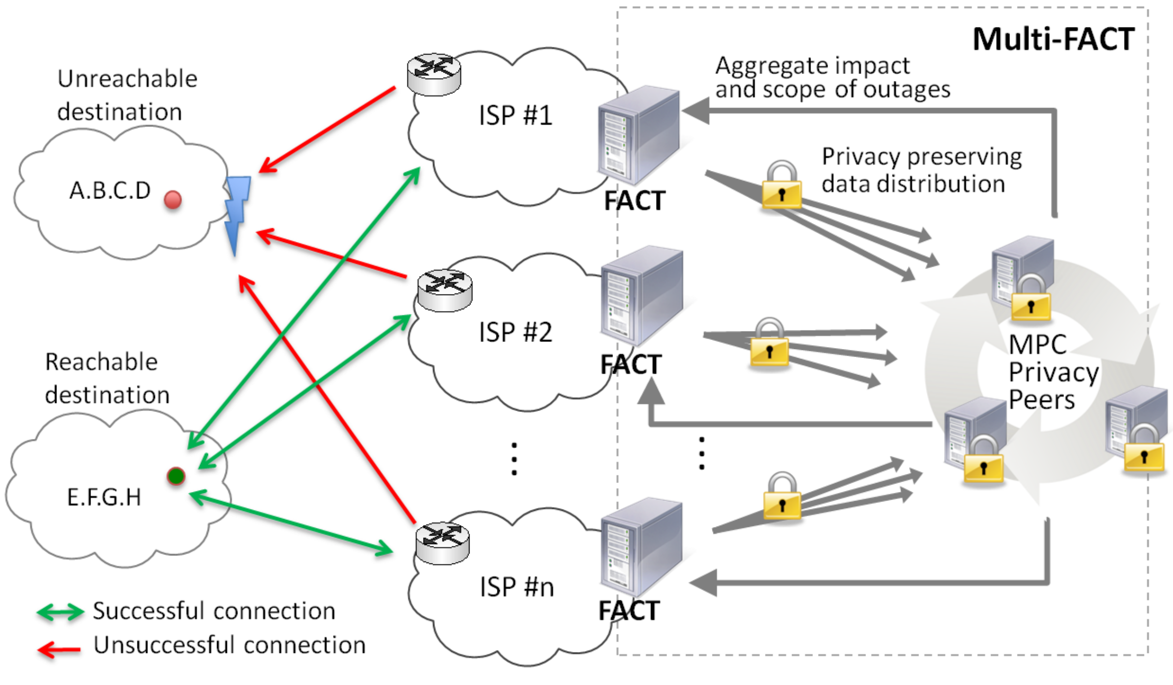


Figure 1: The architecture of our proposal.

Algorithm 1: Multiset union using a combination of comparison and addition operations.

```
Input: Input multiset from $m$ domains:
       $[\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_m]$
Output: $\mathcal{S}_{out}$
1  $\mathcal{S}_{out} = \mathcal{S}_1$;
2  for $a = 2..m$ do
3      foreach $(x_i, y_i) \in \mathcal{S}_a$ do
4          inSet = False;
           // If $x_i \in \mathcal{S}_{out}$, update the count
5          foreach $(x_j, y_j) \in \mathcal{S}_{out}$ do
6              if $x_i = x_j$ then
7                  $y_j = y_j + y_i$;
8                  inSet = True;
9                  break;
10             end
11         end
           // Otherwise add $(x_i, y_i)$ to $\mathcal{S}_{out}$
12         if !$inSet$ then
13             append $(x_i, y_i)$ to $\mathcal{S}_{out}$;
14         end
15     end
16 end
```

provide any association between network domains and the destinations towards which they experience outages.

4. MULTI-FACT

While FACT has many advantages compared to other network outage monitoring systems, it does not have sufficient information to troubleshoot outages. This follows from FACT only using the traffic data collected in the domain where it was deployed, and therefore it has limited visibility into the scope of an outage. Efficiently combining traffic data from multiple domains in a privacy preserving manner can greatly help in troubleshooting outages. To make this feasible we propose Multi-FACT, a privacy preserving distributed network outage monitoring system based on FACT. Given that multiple domains run FACT independently, Multi-FACT aggregates the outputs of FACT from these domains using MPC to preserve the confidentiality of input data, i.e., of IP addresses.

Combining FACT outputs from multiple domains generates more information about an outage, such as the scope and the severity of an outage. The scope of an outage refers to how widespread an outage is, that is, the number of domains that detect the same outage. The scope can be local, semi-global or global. A local outage is only experienced by one domain. In contrast, a global outage is experienced by all (participating) domains. A semi-global outage is experienced by multiple (but not all) domains. Using the scope of an outage, we can narrow down the possible location and cause of the outage. The probable cause of a local outage is a fault in the domain that detected the outage, while a global outage indicates a fault in the remote location. The severity of an outage refers to the total number of local clients in all participating domains that fail to reach a specific destination. This information is useful to prioritize troubleshooting, that is, outages that affect more clients should be addressed with a higher priority.

4.1 High level architecture

Figure 1 illustrates the architecture of Multi-FACT. In Multi-FACT, the individual domains (ISPs) run FACT independently. Based on the traffic flows in a domain, FACT outputs a list of unreachable destinations and the corresponding number of local clients that cannot reach the destinations. To privately aggregate FACT outputs, each domain generates secret shares based on the FACT outputs using SSS and sends the shares to the privacy peers. The privacy peers then perform the private data aggregation and return the result to the individual domains.

4.2 Aggregating FACT outputs

The scope and the severity of an outage are computed separately. Recall that FACT outputs the unreachable destinations and the corresponding number of local clients that cannot reach the destinations. To compute the scope, we need to count the number of domains that cannot reach each unreachable destination. To compute the global severity of an outage, we need to count the number of clients in all domains that cannot reach each unreachable destination. One possible operation that can be used to compute the scope and the severity of an outage is multiset union. We note that multiset is a set in which elements can be repeated multiple times. In our case, a multiset is used to represent the number of clients or domains that cannot reach specific unreachable destinations. Given several multisets as the input, the multiset union operation computes all the distinct elements and the corresponding total number of occurrences of each element in the input multisets. The multiset representation and the aggregation process differ for the computations of the scope and the severity of an outage. In particular, they are used as follows:

- **Scope computation:** each domain prepares a multiset, where the elements represent the unreachable destinations, and each unreachable destination only occurs once. In this case, an input multiset simply represents the destinations unreachable from a domain. The multisets from all domains are the input to the multiset union operation. The resulting aggregate multiset contains the unreachable destinations, where each destination is repeated as many times as the total number of domains that detected the same outage.

- **Severity computation:** each domain prepares a multiset, where the elements represent the unreachable destinations, and the number of occurrences for each unreachable destination corresponds to the number of local clients in the domain that cannot reach the destination. Then, the multiset union operation computes an aggregate multiset, which contains the unreachable destinations. Each destination is repeated as many times as the total number of clients that cannot reach it.

4.3 Multiset union in MPC

The multiset union operation is used to aggregate the outputs of FACT from multiple domains to determine the scope and severity of outages. In MPC, multiset union operation is typically performed using one of two methods. The first method is by combining comparison and addition operations. The second method is based on using counting

Algorithm 2: Inserting y_i, the count of x_i, to a CBF.

```
Input: $(x_i, y_i)$
// Given $k$ hash functions, $h_1, \dots h_k$
for $j = 1..k$ do
    $pos_j = h_j(x_i)$;
    if $CBF(pos_j) < y_i$ then
        $CBF(pos_j) = y_i$;
    end
end
```

Algorithm 3: Retrieving y_i, the count of x_i, from a CBF.

```
Input: $x_i$
Output: $y_i$
// Given $k$ hash functions, $h_1, \dots h_k$
for $j = 1..k$ do
    $pos_j = h_j(x_i)$;
    // If the CBF position is 0, then $x_i \notin$ CBF
    if $CBF(pos_j) = 0$ then
        return $x_i \notin$ CBF;
    end
    // Find the minimum value
    if $j = 1$ then
        $y_i = CBF(pos_j)$;
    end
    if $CBF(pos_j) < y_i$ then
        $y_i = CBF(pos_j)$;
    end
end
return $y_i$
```

Bloom filter (CBF). We discuss the two methods in detail below.

We use the following notation in the rest of the paper. A multiset $\mathcal{S}$ contains r input elements, which represent unreachable destinations, and each input element can be repeated multiple times. $\mathcal{S}$ is represented as a vector of tuples, $((x_1, y_1), (x_2, y_2), \dots (x_r, y_r))$. For the tuple (x_i, y_i), x_i represents an *element* of the multiset and y_i is the number of occurrences or the *count* of x_i in the multiset.

4.3.1 Comparison-based method

A straightforward method to perform multiset union in MPC is by using a combination of comparison and addition operations. Algorithm 1 shows multiset union using this method. The comparison (equal test) operation is used to determine whether an unreachable destination appears in different multisets (line 6 in Algorithm 1) and the addition operation is used to sum the total number of occurrences of the unreachable destinations (line 7 in Algorithm 1).

While the approach is seemingly straightforward, the computation relies on comparison operations, which require several rounds of MPC multiplication operations and result in a high computation overhead [8]. We analyze the runtime of this approach in Section 6.1. Furthermore, the aggregate multiset discloses all the unreachable destinations, including destinations that may be reachable from some domains, but unreachable from others. This might not be a desirable privacy guarantee: it reveals, to domains that can reach a destination, that some other domains (potentially competing ISPs) cannot reach the same destination, but without revealing the identity of these other domains. To avoid disclosing that information, a better approach (considering privacy) would be to reveal how many other domains cannot reach a destination, if and only if this destination is unreachable from the local domain.

4.3.2 CBF-based method

As proposed in [24], an alternative method to computing multiset union is by using counting bloom filters (CBF). In this case, a counting bloom filter (CBF) is used to represent a multiset, and multiset union operation is computed as a fixed array summation. A CBF is an integer array of size s, where initially all the elements are set to 0. Recall that we have a multiset $\mathcal{S}$ consists of r input data, $((x_1, y_1), (x_2, y_2), \dots (x_r, y_r))$. Converting a multiset $\mathcal{S}$ to a CBF requires k independent hash functions $h_1, h_2, \dots, h_k$ where the range of each hash function is between 1 and s. The algorithm to insert (x_i, y_i) is provided in Algorithm 2. The hash functions are used to determine the positions in the CBF to insert (x_i, y_i) and the values in the positions are updated to y_i only if the existing value in the CBF is less than y_i. The algorithm to retrieve the count of x_i, y_i, from the CBF is provided in Algorithm 3. Note that we need the set of hash functions that are used to convert the multiset to CBF. As before, the hash functions are used to determine the positions in the CBF. If any of the CBF positions is 0 then $x_i \notin \mathcal{S}$. Otherwise, y_i is the minimum of all the positions that x_i was hashed to.

All the domains select the same set of hash functions to convert their multiset (i.e., the list of unreachable destinations) to the input CBF. As a result, elements that appear in different input multisets are hashed to the same positions. Therefore, the CBF representation transforms multiset union operation to summation of integer arrays. This means that CBF-based multiset union only requires addition in MPC, which is an inexpensive operation since the operation is performed locally. However, CBF-based multiset union affects the accuracy of the computation since CBF introduces errors through hash collisions.

4.4 Multi-FACT operation using multiset union

We use the CBF-based multiset union in Multi-FACT. Figure 2 illustrates the MPC protocol for CBF-based multiset union. Each domain runs FACT on its traffic independently and uses the output of FACT as the input multiset. Next, each domain converts its input multiset to a CBF using a set of hash functions. The multiset content depends on the type of computation, as discussed in Section 4.2. We assume that all the domains use the same hash functions for converting the input multisets to CBFs. Each domain then uses SSS to generate secret shares for each element in the CBF and distributes the shares to the privacy peers. The privacy peers then collaboratively perform the private computation (summation) on the shares and recover the resulting CBF. The resulting CBF is then sent back to the domains. To determine the total count for each unreachable destination (which corresponds to either the scope or the severity), each domain retrieves the count for each unreachable destination.

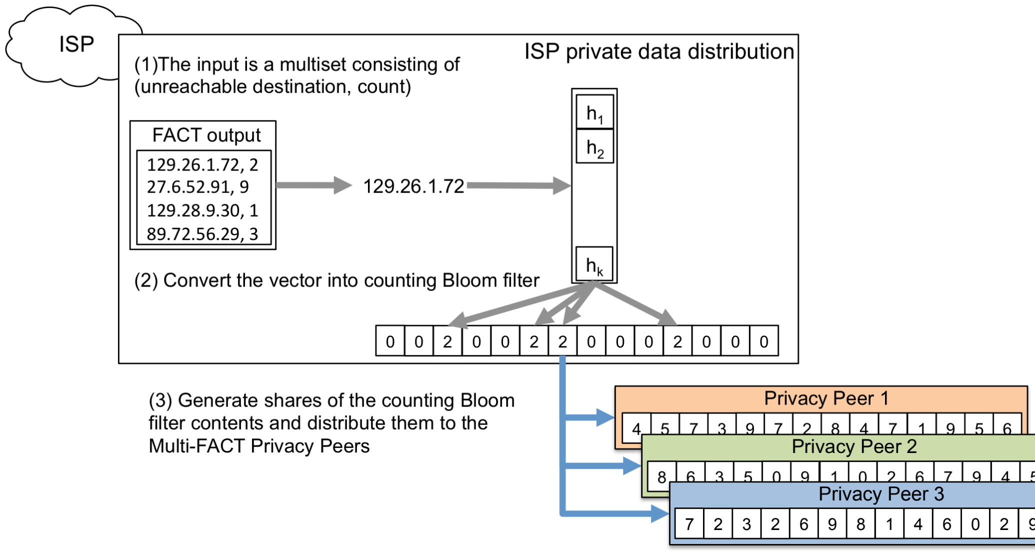

ISP
ISP private data distribution
(1)The input is a multiset consisting of (unreachable destination, count)
FACT output
129.26.1.72, 2
27.6.52.91, 9
129.28.9.30, 1
89.72.56.29, 3
129.26.1.72
h1
h2
hk
(2) Convert the vector into counting Bloom filter
0 0 2 0 0 2 2 0 0 0 2 0 0 0
(3) Generate shares of the counting Bloom filter contents and distribute them to the Multi-FACT Privacy Peers
Privacy Peer 1
Privacy Peer 2
Privacy Peer 3
7 2 3 2 6 9 8 1 4 6 0 2 9 3

Figure 2: CBF-based multiset union in MPC.

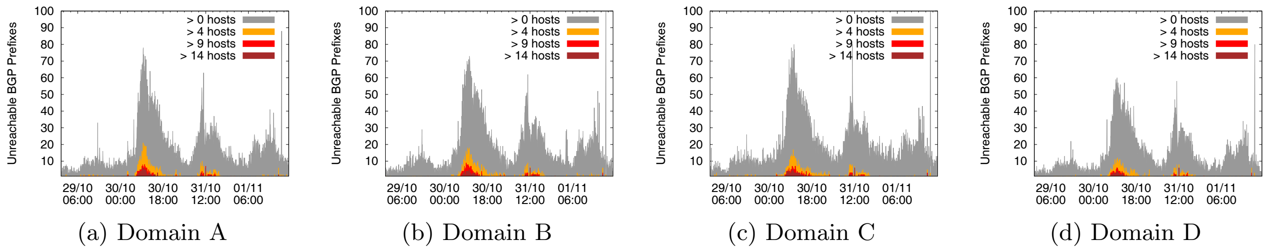


Figure 3: The local severity (number of affected local hosts) of the detected outages in different domains of SWITCH for Hurricane Sandy event.

5. REAL-WORLD CASE STUDIES

In this section we analyze the impact of two important real-world events to illustrate the usefulness of Multi-FACT. Through our case studies, we also provide a novel perspective on the impact of these events, based on a new dataset. As input we use traffic traces from SWITCH [34] – a national backbone ISP in Switzerland that connects approximately 40 universities, governmental institutions and research laboratories to the Internet. Specifically, we use unsampled NetFlow data collected from the border routers of SWITCH during: 1) Hurricane Sandy and 2) a major misconfiguration in one of the largest Internet Exchange Points (IXP) in Europe that partitioned its client base.

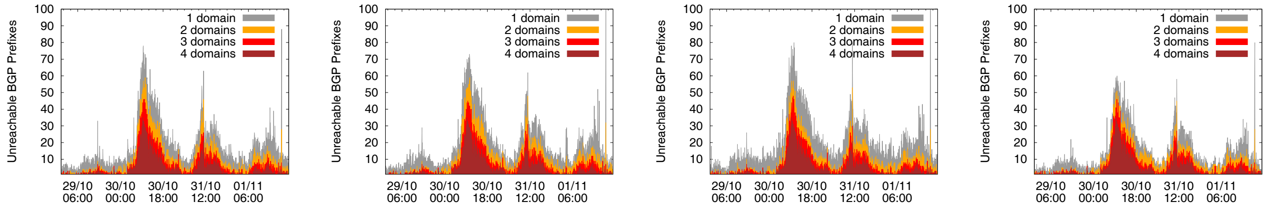
We note that, in this section, we use a Ruby based implementation of Multi-FACT that does not include MPC functionality. Our purpose was to demonstrate the benefits provided by the use of Multi-FACT, rather than the privacy feature, which is an additional operational requirement (to enable ISPs to privately share data for Multi-FACT computations). We also note that consequently, there is no error introduced by the use of CBF operations. We estimate this error, in a fully functional Multi-FACT that includes MPC, in Section 6.

5.1 Troubleshooting Hurricane Sandy

In the evening of October 29 2012, Hurricane Sandy hit the east coast of North America. It was the deadliest and most destructive hurricane in the 2012 Atlantic hurricane season with damage estimated around 75 billion USD. Some of the worst hit areas were New York and New Jersey, which are the locations of major data centers. Hurricane Sandy resulted in power loss, fiber cuts and other damages that affected Internet connectivity.

We investigate the impact of Hurricane Sandy on the Internet connectivity of a remote location, as measured with FACT, and demonstrate the utility of correlating data from multiple domains. To simulate a multi-domain setting, we divide the SWITCH traffic equally into four domains, based on the IP addresses of the local hosts. Next, we analyze the Hurricane Sandy event independently in each of these domains using FACT. Finally, we determine the overall scope of the detected outages using Multi-FACT.

Figure 3 and 4 show the impact of Hurricane Sandy in terms of BGP prefixes unreachable in each of the four domains. We observe that almost 80 BGP prefixes were unreachable from the four domains during the time of the event. We observe a sharp increase in the number of unreachable BGP prefixes during the event (from October 30 00:00 in Figure 3 and 4) compared to the day before. Fur-



(a) Domain A (b) Domain B (c) Domain C (d) Domain D

Figure 4: The overall scope of the detected outages as viewed by individual domain of SWITCH for Hurricane Sandy event.

thermore, while the number of unreachable prefixes decreased over time, a significant number of prefixes was still unreachable two days after the event. In addition, the passive measurement reveals a diurnal pattern in user traffic and the resulting detected network outages.

Figure 3 shows the local severity for each domain during Hurricane Sandy. In the figure, we annotate the number of prefixes that had a higher impact (i.e., affected more local hosts) using darker colors. We observe that in all four domains, most prefixes affect a small number of local hosts (denoted in grey). This means the prefixes affected by Hurricane Sandy were not particularly popular in Switzerland. Note that the affected prefixes are registered to organizations located around New York.

Figure 4 shows the scope of the detected unreachable BGP prefixes at each domain. We use colors to highlight the number of domains in SWITCH that were unable to connect to certain BGP prefixes. In particular, darker colors denote more domains are affected. From the figure, we observe that the majority of the unreachable prefixes detected at each domain also affect other domains in SWITCH.

5.2 Troubleshooting A Partitioned IXP

On March 25 2010, part of the SWITCH network experienced a partial blackhole after a scheduled maintenance at a large European IXP. We investigate the impact of this event in terms of the overall severity and the scope of outages using Multi-FACT, as seen by the six main customers of SWITCH. To obtain the individual traffic of each of the six main customers, we filter SWITCH traffic based on the IP address ranges of each of the customers during the event. We use Multi-FACT to detect the unreachable destinations in each domain and aggregate them to determine the scope and severity of the outages. Note that Multi-FACT can detect outages in three different levels of granularity, IP addresses, /24 networks and BGP prefixes, and perform the aggregation separately.

Figure 5 shows the scope of the detected outages in the BGP prefix level. In this experiment, Multi-FACT only aggregates prefixes that are unreachable by at least two local hosts at each domain. As before, we use colors to denote the number of domains that are affected by unreachable prefixes, where dark colors indicate that more domains are affected. We observe that half of the unreachable prefixes only affect one domain. Since we only consider outages that affect at least two local hosts at each domain, we do not detect global outages (i.e., outages common to all six domain) for this event. Figure 6 shows the overall severity of the detected outages at the level of BGP prefixes obtained using

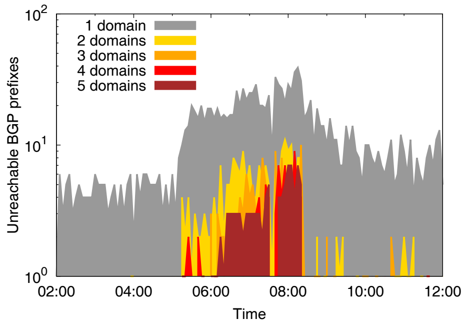


Figure 5: Scope of detected outages during partitioned IXP event.

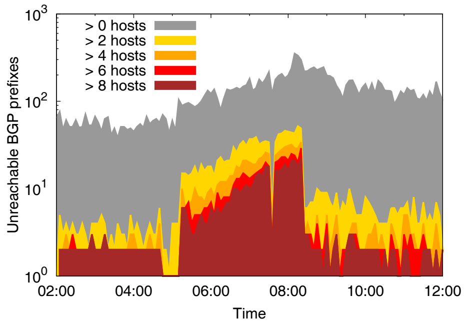


Figure 6: Overall severity of detected outages during partitioned IXP event.

Multi-FACT. We use different colors to highlight the total number of local hosts that are affected by unreachable BGP prefixes. We observe that half of the detected outages are experienced by two or more local hosts, while the other half only affect one local host.

5.3 Discussion

In this section, we evaluated the scope and the severity of the outages detected during Hurricane Sandy and the partitioned IXP events. Scope is useful for determining the root cause of outages, by narrowing down the possible source of the outage, while severity helps in prioritizing troubleshooting.

Typically, outages that affect a large number of local clients (i.e., have more severe impact) should receive troubleshooting priority, as the number of affected local clients directly relates to the popularity of the remote destination.

Domain	The scope of outages	
	Local	Semi-global/global
A	45.4%	54.6%
B	45.69%	54.31%
C	48.53%	51.47%
D	42.59%	57.41%

Table 1: The percentage of outages detected during Hurricane Sandy event, classified by the scope per individual domain.

For example, for the Hurricane Sandy event, each domain can prioritize troubleshooting for the 3.8% of outages (on average) that affect more than 14 local clients. In the partitioned IXP events, 2.9% of the outages (affecting more than eight clients) should receive troubleshooting priority.

Semi-global and global outages affect multiple domains. This indicates that the root cause is not in the individual domains that detected the outage. Global outages are likely caused by problems in the remote destination's network, and the source of semi-global outages is in the path between the domains and the remote destination. For these types of outages, the problem is beyond the control of the network administrators of the local domains. In this case, the local administrators need to notify their counterpart at the domain where the problem originates. As shown in Table 1, more than 50% of the outages experienced in each domain during Hurricane Sandy were either semi-global or global outages, while only 18.5% of the outages detected during the partitioned IXP event were semi-global or global. On the other hand, local outages only affect one domain, and hence the likely cause of these outages is local misconfiguration. In this case, the local administrator needs to further troubleshoot the infrastructure. On average, 45% of the outages detected by the domains during Hurricane Sandy were local, while the percentage of local outages during the partitioned IXP event was 81.5%.

6. IMPLEMENTATION TRADEOFFS

In this section, we compare the performance of the comparison-based and CBF-based multiset union. The goal is to assess the implementation tradeoffs between the two multiset union methods. We perform the evaluation using the SEPIA MPC library [8], which contains the operations that are required for comparison-based multiset and a built-in CBF-based multiset union operation.

To run the experiment, we implement a CBF-based multiset union protocol in SEPIA, using an example protocol as the code base. Furthermore, we generate all the required materials (including running scripts and configuration files for each peers) using a built-in configuration editor. The evaluation is performed in an OpenStack cloud, which consists of six workers. Each worker has 12 CPU cores of Intel Xeon 2.67GB and a Gigabit network connection. We use nine virtual machines, where each machine has 2GB of memory and one virtual CPU based on KVM virtualization. Due to the system limitation, we cannot assign all the input and privacy peers in the experiments to different machines (recall the definitions of input and privacy peers from Section 3.2). Hence we simulate the input and privacy peers, where the peers are uniformly distributed across the virtual machines. The number of input data, r, for each of the input peers, corresponds to the highest number of unreachable IP addresses in the Hurricane Sandy event, which is 2,114 IP addresses. We vary the number of ISPs aggregating input data (i.e., the input peers) and the number of MPC privacy peers. The number of input peers are varied from 5 to 25 and the number of privacy peers are varied from 3 to 9.

Input peers	Privacy peers			
	3	5	7	9
5	866	894	899	929
10	2,825	2,914	2,933	3,030
15	5,702	5,882	5,919	6,116
20	9,522	9,823	9,884	10,214
25	14,311	14,763	14,855	15,351

Table 2: Estimated runtime in days for comparison-based multiset union operation.

6.1 Protocol runtime

We compare the protocol runtime of the comparison-based and the CBF-based multiset union. The protocol runtime refers to the total computation and communication time required to perform multiset union on one set of input data. Note that the protocol time does not include the peers discovery time.

For comparison-based multiset union, we estimate the time taken to run the protocol for different number of input peers and privacy peers. Given the input data size and the number of input peers, we run a non-MPC comparison-based multiset union protocol and count the number of comparison operations (n_c) and addition operations (n_a) required. The non-MPC protocol is an original implementation in Ruby. To estimate the time, we measure the runtime of one addition of two input data operation (t_a) and one comparison (i.e., equality check) of two input data operation (t_c) in SEPIA for different numbers of privacy peers. For estimating the runtime of the addition and comparison operations, we conduct the experiment 10 times and compute the average for the estimated runtime. The estimated protocol runtime is computed as $n_c \times t_c + n_a \times t_a$. Therefore, given the numbers of input and privacy peers, the estimated runtime is the total number of operations required for the number of input peers multiplied by the total time required for the number of privacy peers.

Table 2 shows the estimated runtime for the comparison-based multiset union. We observe that the estimated runtime is generally quite high. For example, the runtime for a 5 input peers and 3 privacy peers configuration is around 866 days, while the estimated runtime for 25 input peers and 9 privacy peers is around 15,351 days. For a given number of input peers, the estimated runtime increases as there are more privacy peers. For example, the estimated runtime for 5 input peers and 5 privacy peers is 894 days, in contrast to 866 days required for the same number of input peers and 3 privacy peers. Note that our estimation only considers the addition and comparison operations, and there may be other operations required for the actual comparison-based protocol in MPC, which would increase the protocol runtime.

Figure 7 shows the runtime of the CBF-based multiset union in SEPIA. The figure presents the protocol runtime for different peer configurations, each averaged over 5 runs. We observe that the protocol runtime increases as the number of privacy peers increases. On average, the runtime for a configuration with 9 privacy peers is almost one second more than the runtime for a configuration with 3 privacy peers.

Comparing the results in Table 2 and Figure 7, there is a significant runtime difference between the two multiset union methods. For example, for 3 privacy peers configuration, the runtime for comparison-based method is at least 74 million times the runtime for CBF-based method. The CBF-based method has shorter runtime since the method is essentially a summation of integer arrays. CBF-based multiset union only requires the use of the addition operation, which is the most efficient operation in MPC. In contrast, comparison-based multiset union requires comparison and addition operations. The implementation of the equality check operation in SEPIA is quite efficient [8], however it still requires several multiplication operations, which results in higher runtime.

6.2 Accuracy of the results

We compare the accuracy of the results of the comparison-based and the CBF-based multiset union. The main advantage of the comparison-based multiset union is that the results are error free. On the other hand, CBF-based multiset union introduces overestimation error. The goal of this evaluation is to determine the error rate of the CBF-based multiset union when compared to the results of the comparison-based multiset union and the possible ways to minimize the overestimation error.

Recall from Section 4.3 that an input $(x_i, y_i) \in \mathcal{S}$ consists of an element (i.e., unreachable destination) x_i, and its corresponding count y_i. Furthermore, each (x_i, y_i) is represented by k copies of y_i in the CBF. Overestimation error occurs when the computation result of an input data is higher than what it should be. This error is triggered by collision, that is, when two or more input points are hashed to the same positions in the CBF. Note that overestimation error only occurs when all k copies of an input data entry in the CBF are affected by collision.

Collision can occur locally, between two or more inputs of an input peer, or globally, between two or more inputs from different input peers. Local collision occurs during the input data insertion process. When this happens, the input data with the smaller count is overwritten by the input data with the larger count (lines 3-5 in Algorithm 2). This also means that scope computation is not affected by overestimation error due to local collision, since the counts of all the input data are equal. Global collision becomes apparent when aggregating the CBFs from different input peers. It occurs because different input peers have different sets of input data. Some of the inputs of an input peer are common to all input peers, some are common to multiple input peers and the rest are unique to an input peer. In our case, the distribution of the input data relates to the popularity of a destination across different input peers. Another factor contributing to global collision is the existence of local collisions.

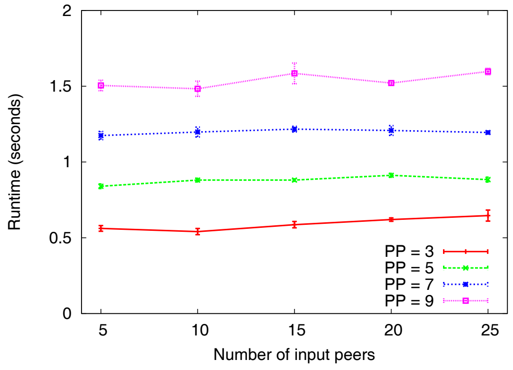


Figure 7: Average runtime for CBF-based multiset union operation.

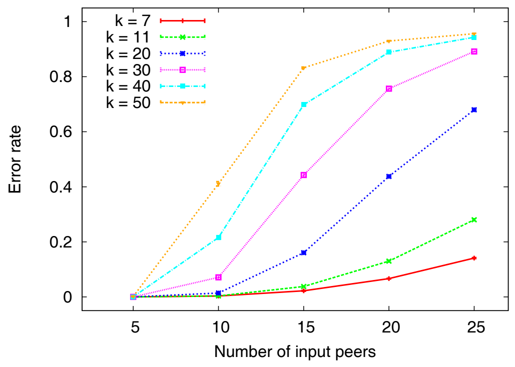


Figure 8: Error rate for varying number of hash functions and CBF size 131,072.

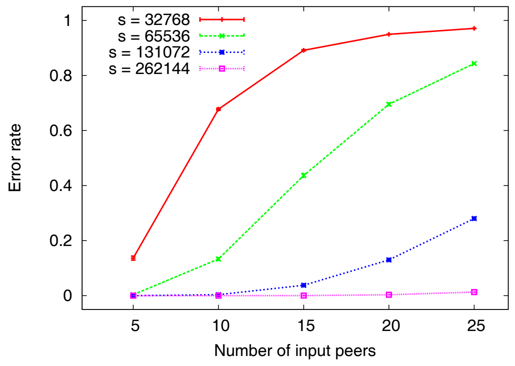


Figure 9: Error rate for varying CBF sizes and 10 hash functions.

In Figure 8 and 9, we show the error rate for different numbers of hash functions (k) with CBF of size 131,072 and the error rate for different sizes of CBF (s) with 10 hash functions. The error rate is computed as the number of incorrect results divided by the total number of unique inputs across all the input peers (obtained from the non-MPC protocol). Note that in this evaluation, the number of privacy peers does not affect the accuracy of the results, and the inputs are always hashed to the same positions in the CBF.

We observe from Figure 8 that for 10 or more input peers the error rate varies significantly as the number of hash functions increases. For example, for 15 input peers, the error rate is 18% when $k = 5$ and 87% when $k = 50$. However, the error rate for 5 input peers does not significantly change as the number of hash functions used increases. In this case, the error rate remains close to 0% regardless of k. Due to limited overlap between inputs from different input peers, a higher number of input peers results in a higher number of unique inputs. This in turn increases the probability of global collision and therefore increases the error. Furthermore, the number of hash functions used also affects the error rate in two opposing ways. On one hand, having more hash functions provides more redundancy, that is, the computation result is correct if at least one of the hashes of an input is not affected by collision. However, a higher number of hash functions also contributes to a higher chance of collision, as a large value of k corresponds to more copies of an input data stored in the CBF, which increases the density of the CBF.

Figure 9 shows the error rate for 10 hash functions for different CBF sizes. We observe that the CBF size significantly affects the error rate. For a CBF size of 32,768, we can see that the error rate is extremely high, especially as the number of input peers increases. For example, the error rate for 25 input peers is 99%. This is because there are not enough spaces to store the k copies of each input in the CBF without inducing collisions. In contrast, the error rate for a CBF size of 262,144 is close to 0% regardless of the number of input peers. Hence, we can conclude that a larger CBF size which minimizes the error rate should be used, with consideration of the hardware limitations of the computing platforms used for the Multi-FACT system.

6.3 Discussion

We compared the protocol runtime and accuracy of the results of the comparison-based and CBF-based methods. The runtime of the CBF-based method is significantly better than that of the comparison-based method. Given the same number of input and privacy peers, the runtime of comparison-based method is at least 74 million times that of the runtime of the CBF-based method. While the CBF-based method has low runtime, it introduces error to the computation results. Based on Figures 8 and 9, we note that the error can be minimized or eliminated by carefully selecting the size of the CBF and the number of hash functions. As a rule of thumb, the combination of a large CBF and a small number of hash functions reduces the error. Also note that the number of inputs and the number of input peers also contribute to the error.

In a real-world scenario, we believe that tier-1 ISPs are interested in collaborative troubleshooting of outages. Considering that there are around 15 tier-1 ISPs globally,[3] Multi-FACT provides a viable solution to help troubleshooting outages.

7. CONCLUSIONS

We proposed a novel approach to help diagnose network outages by correlating traffic measurements across multiple ISPs. The proposed system is based on MPC to protect the sensitive traffic data and is capable of detecting the scope and severity of network outages. For near-real-time performance, we also proposed an efficient method to aggregate traffic measurements using the CBF-based multiset union operation. We evaluated the utility of our system by applying the system to datasets from a medium-sized ISP, which contain real-world incidents, to determine the scope and severity of the detected outages. We showed how troubleshooting can be prioritized based on the severity and the scope, to narrow down the source of the outages. We showed that the performance of CBF-based multiset union is close to real-time. Furthermore, the overestimation error introduced by the CBF can be minimized or eliminated by carefully selecting the number of hash functions used and the size of the CBF.

[3] `http://en.wikipedia.org/wiki/Tier_1_network`

8. ACKNOWLEDGMENTS

The authors would like to acknowledge the contributions of Dilip Many and Martin Burkhart, who first developed the CBF-based multi-set union protocol in [24] before graduating for the industry. This research work has been supported by funding from National ICT Australia (NICTA). NICTA is a research organization funded by Australian Government research initiatives through the Australian Research Council (ARC). We are grateful to SWITCH for providing us data.

9. REFERENCES

[1] B. Applebaum, H. Ringberg, M. J. Freedman, M. Caesar, and J. Rexford. Collaborative, privacy-preserving data aggregation at scale. In *Proceedings of the 10th international conference on Privacy enhancing technologies*, PETS'10, pages 56–74, Berlin, Heidelberg, 2010. Springer-Verlag.

[2] E. Augustine, C. Cushing, A. Dekhtyar, K. McEntee, K. Paterson, and M. Tognetti. Outage detection via real-time social stream analysis: leveraging the power of online complaints. In *Proceedings of the 21st international conference companion on World Wide Web*, pages 13–22, New York, NY, USA, 2012. ACM.

[3] A. Ben-David, N. Nisan, and B. Pinkas. Fairplaymp: a system for secure multi-party computation. In *Proceedings of the 15th ACM conference on Computer and communications security*, CCS '08, pages 257–266, New York, NY, USA, 2008. ACM.

[4] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, STOC '88, pages 1–10, New York, NY, USA, 1988. ACM.

[5] R. Bendlin, I. Damgård, C. Orlandi, and S. Zakarias. Semi-homomorphic encryption and multiparty computation. In *Proc. EUROCRYPT'11*, pages 169–188, Berlin, Heidelberg, 2011. Springer-Verlag.

[6] P. Bogetoft, D. Christensen, I. Damgård, M. Geisler, T. Jakobsen, M. Krøigaard, J. Nielsen, J. Nielsen, K. Nielsen, J. Pagter, M. Schwartzbach, and T. Toft. Secure multiparty computation goes live. In R. Dingledine and P. Golle, editors, *Financial Cryptography and Data Security*, volume 5628 of *Lecture Notes in Computer Science*, pages 325–343. Springer Berlin Heidelberg, 2009.

[7] M. Burkhart and X. Dimitropoulos. Privacy-preserving distributed network troubleshooting? bridging the gap between theory and practice. *ACM Trans. Inf. Syst. Secur.*, 14(4):31:1–31:30, Dec. 2008.

[8] M. Burkhart, M. Strasser, D. Many, and X. Dimitropoulos. SEPIA: privacy-preserving

aggregation of multi-domain network events and statistics. In *Proc. USENIX Security'10*, 2010.

[9] D. R. Choffnes, F. E. Bustamante, and Z. Ge. Crowdsourcing service-level network event monitoring. In *Proceedings of the ACM SIGCOMM 2010 conference*, SIGCOMM '10, pages 387–398, New York, NY, USA, 2010. ACM.

[10] G. Cormode and S. Muthukrishnan. An improved data stream summary: the count-min sketch and its applications. *Journal of Algorithms*, 55(1):58 – 75, 2005.

[11] R. Cramer and I. Damgård. Multiparty computation, an introduction. In *Contemporary Cryptology*, Advanced Courses in Mathematics - CRM Barcelona, pages 41–87. Birkhsuser Basel, 2005.

[12] A. Dainotti, R. Amman, E. Aben, and K. C. Claffy. Extracting benefit from harm: using malware pollution to analyze the impact of political and geophysical events on the internet. *SIGCOMM CCR*, 42(1):31–39, 2012.

[13] I. Damgård, M. Geisler, M. Krøigaard, and J. Nielsen. Asynchronous multiparty computation: Theory and implementation. In S. Jarecki and G. Tsudik, editors, *Public Key Cryptography – PKC 2009*, volume 5443 of *Lecture Notes in Computer Science*, pages 160–179. Springer Berlin Heidelberg, 2009.

[14] M. Djatmiko, D. Schatzmann, A. Friedman, X. Dimitropoulos, and R. Boreli. Collaborative network outage troubleshooting with secure multiparty computation. *IEEE Communications Magazine*, November 2013.

[15] Downdetector outage detection service. `http://www.downdetector.com`.

[16] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In S. Halevi and T. Rabin, editors, *Theory of Cryptography*, volume 3876 of *Lecture Notes in Computer Science*, pages 265–284. Springer Berlin Heidelberg, 2006.

[17] E. Glatz and X. Dimitropoulos. Classifying internet one-way traffic. In *Proceedings of the 2012 ACM conference on Internet measurement conference*, IMC '12, pages 37–50, New York, NY, USA, 2012. ACM.

[18] D. Gupta, A. Segal, A. Panda, G. Segev, M. Schapira, J. Feigenbaum, J. Rexford, and S. Shenker. A new approach to interdomain routing based on secure multi-party computation. In *Proceedings of the 11th ACM Workshop on Hot Topics in Networks*, HotNets-XI, pages 37–42, New York, NY, USA, 2012. ACM.

[19] A. Hanemann, J. W. Boote, E. L. Boyd, J. Durand, L. Kudarimoti, R. Lapacz, D. M. Swany, S. Trocha, and J. Zurawski. Perfsonar: a service oriented architecture for multi-domain network monitoring. In *Proceedings of the Third international conference on Service-Oriented Computing*, ICSOC'05, 2005.

[20] W. Henecka and M. Roughan. Strip: Privacy-preserving vector-based routing. In *IEEE International Conference on Network Protocols*, 2013.

[21] M. Hirt, C. Lucas, U. Maurer, and D. Raub. Passive corruption in statistical multi-party computation. In *Proc. ICITS'12*. Springer-Verlag, 2012.

[22] E. Katz-Bassett, H. V. Madhyastha, J. P. John, A. Krishnamurthy, D. Wetherall, and T. Anderson. Studying black holes in the internet with hubble. In *Proceedings of the 5th USENIX NSDI Symposium*, pages 247–262, Berkeley, CA, USA, 2008. USENIX Association.

[23] E. Katz-Bassett, C. Scott, D. R. Choffnes, I. Cunha, V. Valancius, N. Feamster, H. V. Madhyastha, T. Anderson, and A. Krishnamurthy. Lifeguard: practical repair of persistent route failures. In *Proceedings of the ACM SIGCOMM 2012 conference on Applications, technologies, architectures, and protocols for computer communication*, SIGCOMM '12, pages 395–406, New York, NY, USA, 2012. ACM.

[24] D. Many, M. Burkhart, and X. Dimitropoulos. Fast private set operations with sepia. Technical Report 345, ETHZ, March 2012.

[25] NANOG. `http://www.nanog.org/`.

[26] outages@outages.org. The outages mailinglist. `http://puck.nether.net/mailman/listinfo/outages`.

[27] RIPE Routing Information Service. `http://www.ripe.net/data-tools/stats/ris/routing-information-service`.

[28] M. Roughan and Y. Zhang. Privacy-preserving performance measurements. In *Proceedings of the 2006 SIGCOMM workshop on Mining network data*, MineNet '06, pages 329–334, New York, NY, USA, 2006. ACM.

[29] M. Roughan and Y. Zhang. Secure distributed data-mining and its application to large-scale network measurements. *SIGCOMM Comput. Commun. Rev.*, 36(1):7–14, Jan. 2006.

[30] Route Views Page. `http://www.routeviews.org`.

[31] D. Schatzmann, S. Leinen, J. Kögel, and W. Mühlbauer. FACT: flow-based approach for connectivity tracking. In *Proc. PAM'11*, pages 214–223, Berlin, Heidelberg, 2011. Springer-Verlag.

[32] A. Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, Nov. 1979.

[33] X. Shi, Y. Xiang, Z. Wang, X. Yin, and J. Wu. Detecting prefix hijackings in the internet with argus. In *Proceedings of the 2012 ACM conference on Internet measurement conference*, IMC '12, pages 15–28, New York, NY, USA, 2012. ACM.

[34] The Swiss Education and Research Network (SWITCH). `http://www.switch.ch`.

[35] B. Tierney, J. Metzger, J. Boote, E. Boyd, A. Brown, R. Carlson, M. Zekauskas, J. Zurawski, M. Swany, and M. Grigoriev. perfsonar: Instantiating a global network measurement framework. In *4th Workshop on Real Overlays and Distributed Systems*, 2009.

[36] M. Zhang, C. Zhang, V. Pai, L. Peterson, and R. Wang. Planetseer: internet path failure monitoring and characterization in wide-area services. In *Proceedings of the 6th OSDI Conference*, pages 12–12, Berkeley, CA, USA, 2004. USENIX Association.

[37] C. Zheng, L. Ji, D. Pei, J. Wang, and P. Francis. A light-weight distributed scheme for detecting ip prefix hijacks in real-time. In *Proceedings of the ACM SIGCOMM 2007 Conference*, pages 277–288, New York, NY, USA, 2007. ACM.