Flowzilla: A Methodology for Detecting Data Transfer Anomalies in Research Networks

Anna Giannakou
Data Science and Technology Dept.
Lawrence Berkeley National Lab
California, USA
agiannakou@lbl.gov

Daniel Gunter
Data Science and Technology Dept.
Lawrence Berkeley National Lab
California, USA
dkgunter@lbl.gov

Sean Peisert
Data Science and Technology Dept.
Lawrence Berkeley National Lab
California, USA
sppeisert@lbl.gov

***Abstract*—Research networks are designed to support high volume scientific data transfers that span multiple network links. Like any other network, research networks experience anomalies. Anomalies are deviations from profiles of normality in a research network's traffic levels. Diagnosing anomalies is critical both for network operators and users (e.g., scientists). In this paper we present Flowzilla, a general framework for detecting and quantifying anomalies on scientific data transfers of arbitrary size. Flowzilla incorporates *Random Forest Regression*(RFR) for predicting the size of data transfers and utilizes an adaptive threshold mechanism for detecting outliers. Our results demonstrate that our framework achieves up to 92.5% detection accuracy. Furthermore, we are able to predict data transfer sizes up to 10 weeks after training with accuracy above 90%.**

***Index Terms*—network security, network performance measurement, anomaly detection**

I. Introduction

Modern science depends on high-performance research networks, just as it depends on high-performance computing and storage, to make progress. For example, some of the experiments at the Large Hadron Collider in Switzerland and France require high-speed transfers of petabytes of data across multiple sites located in different parts of the world. The LSST [1] in the Chilean mountains has near-real-time requirements on the order of seconds to transfer and analyze data halfway around the world.

The research networks used for science (henceforth, just "research networks") use similar technologies to the ones present in commodity Internet; however, their workload and requirements are very different. They service far fewer flows in a day but, for some portion of those flows, must provide orders of magnitude lower packet loss and better throughput [2].

Just as with any network, research networks experience variations in traffic patterns. While some variations can be ascribed to known causes, others should be classified as anomalies. These anomalies can be associated with a variety of factors, from malicious activities (e.g., data exfiltration attempts, malicious uploads) to simple user errors, to network device (e.g., router) misconfigurations.

Regardless of their source, real time detection and diagnosis of these anomalies remains a challenge for network engineers and is crucial from an operational standpoint. Undetected anomalies can lead to network congestion and over-utilization of network devices. Detecting anomalies is also important for identification of components of the software and network stack that are experiencing "soft failures" that degrade performance, but do not generate explicit alerts.

There are two distinct approaches in detecting network anomalies. The first is *signature-based* detection [3], which focuses on discovering pre-defined "signatures" (i.e. operational patterns) of previously-identified malicious events in network traffic. The second approach, *anomaly-based* detection [3], identifies events that deviate from what is considered as "normal" behavior of the monitored system and thus has the potential to detect novel attacks.

Despite a large literature on *anomaly-based* detection that incorporates machine learning techniques in order to detect outliers in traffic datasets, effectively detecting most network anomalies in most environments remains challenging [4]. These challenges can be attributed to numerous factors, including: 1) the scarcity of training datasets that contain sufficient amounts of flow data for generating accurate profiles of both *normal behavior* and *anomalous behavior*; 2) most network traffic inherently has high degrees of variability, causing the signal of many anomalies to be lost in the noise of legitimate traffic; 3) statistical techniques, such as machine learning, are typically designed to discover similarities rather than detect outliers [4]. Due to these factors, network operators and scientists typically become aware of an anomalous event long after the fact and often without actionable details about the anomaly.

In this work, we focus on overcoming these limitations, while leveraging the distinctive properties of research networks. In particular, our framework focuses on *volume anomalies*, i.e., *any positive or negative change in the size (volume) of the data exchanged between two source and destination endpoints*. Since not all network anomalies are attributed to malicious intent, a volume anomaly might well be related to a simple erroneous user activity. For example, in the context of a scientific network, a larger than usual download could be related to a misconfigured automated script that starts downloading large amounts of data from the host where it was executed to the user's local machine. On the other hand, a large and fast download could indicate an attacker trying to exfiltrate a large amount of scientific data. Regardless of their underlying cause,volume anomalies are critical to understand in research networks, as they can cause performance problems to applications trying to use the same network components.

The goal of this paper is to describe methods for detecting volume anomalies in large science flows for improving security and reliability for the science that uses research networks. By operating on research network flow data, our approach addresses a fundamental impediment of machine-learning intrusion detection: enormous variability of input traffic data [4], [5]. Our work breaks anomaly detection into three steps: First, we build a profile of normal traffic behavior based on past traffic measurements representing scientific data transfers of arbitrary size. Then, based on the profile, we detect individual flows that deviate significantly in terms of total volume of data being transferred. The deviation refers to either positive or negative volume changes. These flows are labeled as *anomalous*. Finally, we quantify the detected anomalies by their size.

The contributions of the paper are:

1) A general framework, called *Flowzilla*, to detect volume anomalies in large science flows. We build a profile of normal traffic behavior based on past flow measurements. Once the normal traffic profile has been constructed *Flowzilla* uses an adaptive threshold mechanism to detect outlier flows that deviate significantly from the normal profile.
2) We evaluate *Flowzilla* in terms of detection capability and temporal stability. In our evaluation process we systematically inject anomalous network flows in order to calculate the detection rate of our framework for different anomaly sizes. Our results show that *Flowzilla* is able to achieve a detection rate up to 92.5% for different anomaly sizes.

The paper is organized as follows: In Section II we provide the problem description. In Section III we discuss related work on anomaly detection using different machine learning techniques, while Section IV presents our proposed solution and highlights key elements of our approach. Details of our results and evaluation are described in Section V-A. Finally, Section VI concludes the paper with key observations, recommendations, and suggestions for future work.

II. Background

In this section we advocate for the need of an adaptive threshold mechanism for detecting volume anomalies. Furthermore, we describe our data sources and the data collection process.

A. Adaptive threshold

An intuitive approach to *volume anomaly* detection would be to build a *normal* profile for network flow sizes and then flag any data transfer that deviates from that profile (i.e., the size of the transfer exceeds a certain threshold) as anomalous. Setting a constant volume threshold requires careful consideration since setting the threshold value at a relatively low number could lead to an increased number of false positives while setting it too high could have the opposite effect. Even if the anomaly detection system was able to assign a reasonable threshold value automatically, this approach has serious drawbacks. Scientific traffic has high seasonal variability, e.g., large data transfers tend to occur near major conference deadlines. This means that any static threshold approach will regularly produce too many false positives or negatives to be practically useful.

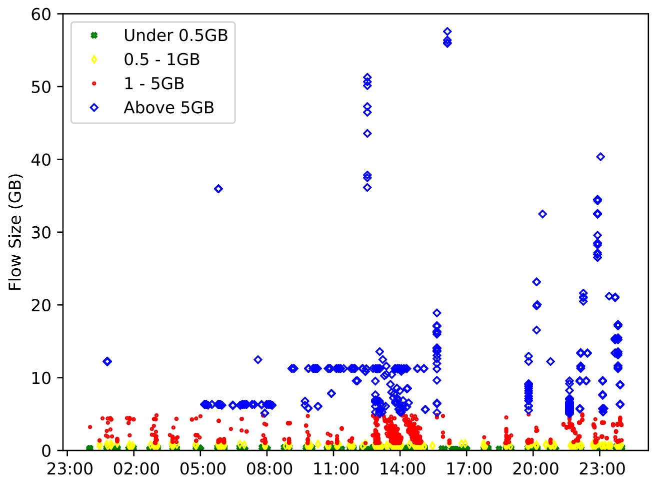

Fig. 1: Flow sizes towards a Data Transfer Node (DTN) at NERSC, on May 29th 2018. Majority of flow sizes are under 20GB.

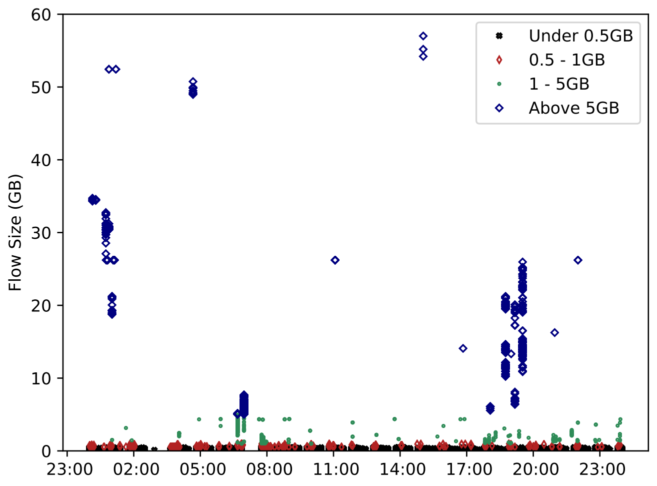

Fig. 2: Flow sizes towards a Data Transfer Node (DTN) at NERSC, on June 5th 2018. Majority of flow sizes exceed 20GB.

Figures 1 and 2 represent the total volume of data transfers towards the NERSC DTNs used for our experiments (see next section for details), over a 24 hour period on two occasions that are one week apart from each other. In the 05/06/2018 dataset, there is a much larger number of transfers exceeding 20GB (5-8 pm time window). As a result, a threshold value that provides an acceptable detection rate in the 05/29/2018 dataset would result in an increased number of false positives in the 05/06/2018 dataset.

To address this issue, we have designed an anomaly detection framework that periodically recomputes and calibrates the threshold value in order to maintain an acceptable detection rate.

B. Network Data

Our anomaly detection model operates on network traffic data collected from 10 hosts referred to as *Data Transfer Nodes (DTNs)* [2] located at the *National Energy Research Scientific Computing Center (NERSC)* [6]. DTNs are explicitly dedicated and fine tuned for performing large data transfers between the NERSC scientific facility and the external scientific community. They are configured with low-latency, high-bandwidth network interface cards (NICs) and equally high-bandwidth disk I/O subsystems designed to limit disk access as a bottleneck. A variety of tools such as Globus online [7], Aspera [8], and GridFTP [9] are typically used in order to automate transfer of large datasets.

Data from each DTN is collected with the "tstat" network monitoring tool [10]. Tstat is in many ways similar to the more well-known NetFlow [11], but is, first and foremost, not sampled. In addition, it is able to group packets into flows and derive detailed statistics and performance metrics for each flow. Grouping packets into flows is particularly useful for optimizing efficiency in processing large amounts of network data. In addition, tstat records include 52 features per flow, including some not collected by NetFlow, such as a count of retransmits. The full list of the computed performance metrics can be found here [10]. Furthermore, as opposed to NetFlow solutions, where several analyzers need to be placed at strategic monitoring points (such as exchange points), tstat is a much less invasive approach that analyzes flow traces at end hosts focusing at end-to-end flow statistics.

III. Related Work

A variety of statistical anomaly detection methods have been previously investigated in order to detect network anomalies that span across multiple links. Gu et al. [12] used a behavior-based anomaly detection method based on maximum entropy for establishing a baseline distribution for benign, "normal" traffic. Packets are divided into different classes based on characteristics like port number, protocol, etc. The baseline distribution is then compared with the relative entropy of new network traces. The dataset analyzed includes one week of flow data collected from a UMASS gateway. Although the authors are able to identify specific packet classes that are more prone to cause anomalies, they did not validate their approach on multiple datasets. Furthermore, the traces analyzed exclude small sized flows.

Lakhina et al. [13] were able to identify different anomaly categories, from flash crowds to denial of service (DoS) attacks and massive data transfers, by aggregating traffic flows at an origin-destination level and applying subspace methods on sampled traffic data from the Abilene Network [14]. Although the analyzed data represent flows from a research network, the authors use a significantly smaller data set (4 weeks of sampled only data) for training. Furthermore, their analysis was performed on NetFlow-type data derived from routers and not end hosts. Lu et al. [15] used wavelet analysis in order to detect outliers in network flows. The authors used DARPA intrusion detection dataset [16] in order to evaluate their solution. The obtained results show a relatively high detection rate (up to 75% in identifying attack classes) but their solution has only been tested on significantly narrower datasets, containing up to a single day's worth of training data that do not focus on scientific data transfers.

Hu et al. [17] use robust support vector machines (SVM) in order to perform anomaly detection over noisy and potentially tainted data. The authors use the 1998 DARPA intrusion detection dataset [16] both for training and evaluation of their solution. Their solution is able to achieve satisfactory detection rates but has only been tested in a relatively outdated dataset with single host data as opposed to multiple end-host data from a research network. Khan et al. [18] describe a scalable SVM-based solution focusing on hierarchical clustering for reducing SVM's training time. Their solution was evaluated on generic internet traffic datasets that did not include high volume scientific data transfers.

Network traffic has high-dimensionality because of the plethora of available feature measurements along with the different distribution and dispersion of each measurement. Lakhina et al. [19] use Principal Component Analysis (PCA) for reducing network traffic dimensionality and for successfully detecting volume-related anomalies. Furthermore, they were able to identify the actual flow(s) that are involved in the anomalous event. The approach used three datasets, each containing a weeks worth of origin-destination (OD) NetFlow data from the European backbone of Sprint network [20] and Abilene network [14]. Although the proposed framework is able to quantify the anomalous events in terms of traffic volume, it is as sensitive to the presence of outliers in the analyzed traffic data as most of the PCA-based methods [21]. Furthermore, the proposed solution is applied to sampled NetFlow-like data, as opposed to complete flow records.

Chhabra et al. [22] investigate both unsupervised and supervised machine learning techniques for classification of scientific flows based on their size (elephant and mice flows). Although this work constitutes the first step towards predicting flow size in research data transfers, it only focuses on flow size prediction rather than identification of potentially anomalous flows. Furthermore, the training dataset contains NetFlow data from exchange points as opposed to our solution which trains on end-host non-sampled data.

Self learning techniques about the system's normal behavior have been the basis for many machine-learning based anomaly detection solutions. However, as Sommer and Paxson report [4], network traffic fundamentally exhibits considerable variability making it hard to establish a notion of normality. In addition, the base-rate fallacy [23] affects the ability of IDS to detect truly malicious activity. Our work leverages the distinct properties of research networks for overcoming the normality establishment limitation, and the relative frequency of volume anomalies (as we define them) mediates the difficulties introduced by the base-rate fallacy.

We have described different machine learning and statistical approaches for anomaly detection on different types of network data. To our knowledge our work distinguishes itself from other solutions in the following manner: we perform anomaly detection using machine learning on non-sampled science traffic data from a research network and our datasets include data transfers of arbitrary size.

IV. Architecture

Figure 3 shows the architecture of Flowzilla. Our framework consists of four main components: the Feature Extraction Filter, the Model, the Threshold Calculator and the Detector. The tstat tool is responsible for collecting data from the DTNs and placing them in a database. The **Feature Extraction filter** securely connects to the tstat database and extracts the desired subset of features. This subset of features is sent, as training data, to the **Model**. Once the model is built, it is stored for later use in the threshold calculation and detection phases. Flowzilla can periodically retrain the **Model** depending on changes on the overall network load. For example, during periods with high frequency large data transfers network operators can opt for retraining in order to increase flow size prediction accuracy. The **Threshold Calculator** is responsible for calculating the detection threshold value based on previous flow data (see Subsection IV-B for a detailed explanation of the threshold calculation mechanism). The threshold value is periodically recalculated to reflect changes in the volume of scientific data transfers. The **Detector** detects anomalous flows based on the **Model's** predicted flow sizes and the threshold value. The **Detector** stores details of anomalous flows and their individual features for further analysis.

There are two phases in our approach: training and detection. During the training phase, we use the training dataset in order to build the model for flow size prediction and calculate the threshold value (steps 1,2 and 3 in Figure 3). Then, the threshold is communicated to the detector (step 4). During the detection phase (steps 5 and 6) we use the *Feature Extraction Filter* to extract the necessary subset of features from new, previously unseen data (step 5). Then, the new data is forwarded to the detector for testing and detection of anomalous events.

Our approach is based on using history based flow size prediction as a tool for detecting volume-related anomalies in research networks. The task of flow size prediction can be formulated as a regression problem, i.e., predicting a real valued number (in our case the flow size) based on multiple real valued input features.

A. Feature Selection

A complete tstat flow entry contains 52 features, each of which has varied distribution across datasets. In order to derive meaningful results when predicting the desired feature, we carefully investigate the relationship between the size of the flow and different combinations of tstat flow entries. Furthermore, by selecting an approximately-best subset of features, we reduce the training time of our model while simultaneously maintaining sufficient information per flow.

Input feature	**Abbr.**	**Tstat field**
Network throughput	T	*bits_per_second, throughput_Mbps*
Flow duration	D	*duration*
Source IP	S_{IP}	*src_geoip*
Destination IP	D_{IP}	*dst_geoip*

TABLE I: Input features and their tstat representation

The subset of features can be found in Table I. All features are weighted equally. Note that we do not use port numbers in our model since we are targeting an application-agnostic prediction. In our model each flow (F) is represented as a set of the features $F = \{T, D, S_{IP}, D_{IP}\}$ at time t. Given F we want to predict y where y is the amount of data (i.e. volume) being transferred on F. This is achieved by training a regression function f and applying f to F. The function f is trained using training data i.e., collected data from previous flows with known feature sets and the corresponding measured volume.

To predict the size of the data transfers we use *Random Forest Regression* (RFR) [24] an established machine learning tool for multi-feature regression. We select RFR for the following reasons: 1) It can accept multiple features and show their importance when generating a prediction regarding the size of the data transfers [25]. 2) The computational cost as well as the training time for RFR remains low for datasets containing hundreds of thousands of scientific network flows.

B. Threshold definition

Before describing our adaptive threshold mechanism, we denote the actual volume of a flow i as V^i_{real} and the predicted volume as V^i_{pred}. Our approach relies on whether the difference between the real volume and the predicted volume, over a given time interval, exceeds a certain threshold. Namely,

$$|V^i_{real} - V^i_{pred}| > T \tag{1}$$

In order to account for seasonal (weekly and daily) trends the value of T is set adaptively based on the mean value of $|V_{real} - V_{pred}|$ which is computed over the full set of flows $1, ..., m$ from recent (e.g. previous week) flow records.

If μ is the mean value of the differences between the real and the predicted volume at times less than t we set T as:

$$T = \mu + x \tag{2}$$

For a flow i occurring at time t if Equation 1 holds then flow i is considered to be anomalous. In Equation 2, x denotes the distance from the mean value μ which we consider to be

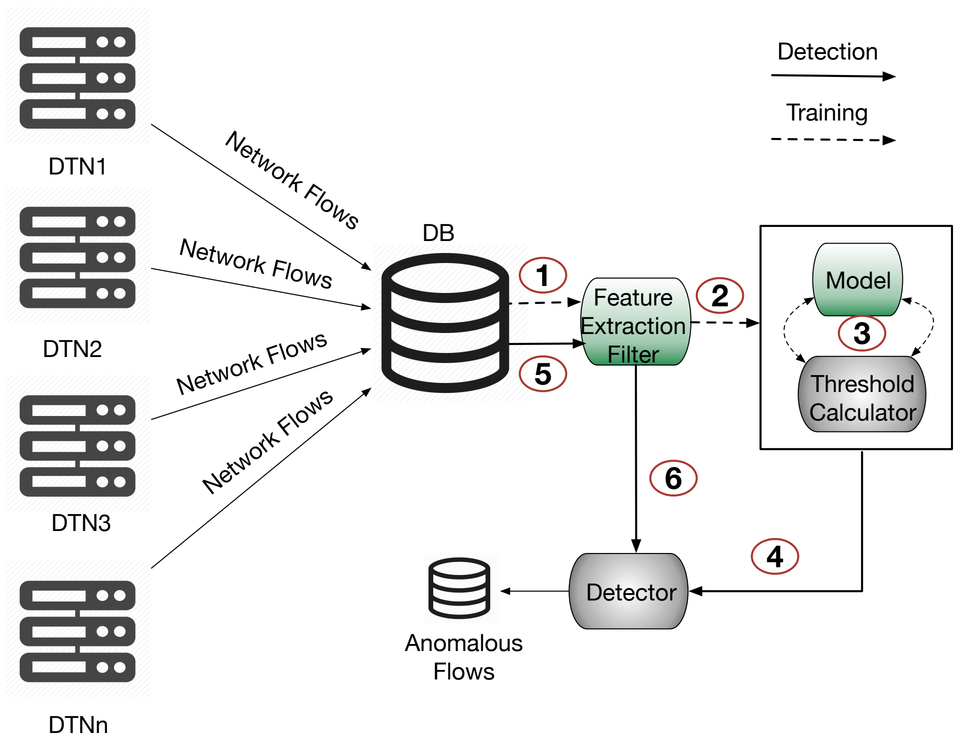

Fig. 3: *Flowzilla* architecture and detection process. Training data from tstat is extracted (1), used to build a model (2), and to calculate threshold values (3) for the Detector (4). Then, data from tstat (5) is sent to the Detector (6) to detect anomalous flows.

an indicator of an anomaly. Allowing the user to arbitrarily set x would significantly affect the detection rate of our approach. Setting a high value would yield some anomalies to remain undetected while setting x at a low percentage would increase the number of false positives.

We formulate the problem in Equation 3 where $f(x, \mu, i) = 0$ implies the presence of an anomaly where as $f(x, \mu, i) = 1$ corresponds to a legitimate flow.

$$f(x, \mu, i) = \begin{cases} 1, & \text{if } |V^i_{real} - V^i_{pred}| \leq \mu + x \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

To address the issue of increased false positives, we set x such that the sum of f over all flows is 90% of the total number of flows.

Once an anomalous flow is detected, the whole set of features $F = \{T, D, S_{IP}, D_{IP}\}$ is extracted from the record and the actual volume of the flow (i.e. V_{real}) is recorded.

Aside from adapting the threshold based on seasonal trends, our framework offers the possibility of re-calibrating the threshold value based on the detection rate over a past time window. For example, if a particular value of x resulted in an increased number of flows that were erroneously considered anomalous (i.e. false positives) then Flowzilla increases x while in the opposite case (i.e. increase in false positives) x is decreased.

V. Evaluation

In this section we describe our evaluation methodology, our experimental setup and the composition of our evaluation datasets. Finally, we discuss obtained results.

A. Methodology

Our evaluation approach is centered around answering four questions: How well does our method perform in detecting volume anomalies in scientific data transfers?, Does Flowzilla detect anomalous flows regardless of their size and time of occurrence? Furthermore, does the direction of the anomalous flow (upload or download) affect our detector's capabilities? Is the quality of our model's predictions in terms of size of data being transferred after a certain period of time stable or does it degrade? This question is crucial for selecting an optimal strategy for retraining frequency.

We answer the first three questions presented by using the following methodology: Since it is impossible to obtain ground truth about which of the previous data transfers were anomalous (remember that we consider all flows belonging to the training set — Dataset1 — as clean), we perform our own anomalous data transfers, from now on referred to as *synthetic anomalies*, and evaluate the ability of our framework to detect and quantify them. For the fourth question, we evaluate the temporal stability and the quality of predictions of our model by testing it on new data obtained weeks after training (Datasets 2 to 5) and we record its accuracy in terms of data transfer size prediction.

Quantifying the ability of a system to successfully identify anomalous events (i.e. events that significantly deviate from a "normal" profile) is an inherently challenging problem [26]. For the scope of this work we use detection rate for our framework as presented in equation 4.

$$\text{Detection Rate} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \tag{4}$$

B. Synthetic Anomalies

Since we investigate volume-related anomalies, we focus on injecting anomalous events that demonstrate variation only in the size of the data being transferred. For injecting synthetic anomalies in normal traffic we first select two injection sizes *large* and *average*. The *large* anomalies include flow sizes of 10GB while the *average* anomalies include flow sizes between 1 and 5GB. Each synthetic anomaly is represented as a file transfer between an external source and one of the ten NERSC DTNs. The file transfer can either be an upload or a download. We opt for bi-directional anomalies in order to cover semantically both aspects of misuse: either a malicious user which would be attempting to exfiltrate sensitive scientific data or a simple user error and/or a network device misconfiguration.

C. Experimental Setup

We used 8 nodes from Grid5000 experimental testbed [27] acting as the sources/destinations of the file transfers. Two sets of experiments were performed: In the first experiment, each Grid5000 node performs five transfers of 10GB each (*large* anomaly) towards or from one of the ten NERSC DTNs. The waiting time between transfers is set to one hour interval. In the second experiment, each Grid5000 node performs five transfers of size between 1 and 5GB (*average* anomaly) towards or from one of the ten NERSC DTNs. The waiting time between transfers in the second experiment is randomly selected between one minute and one hour intervals. In both experiments the destination is selected randomly among a list containing the ten NERSC DTNs. The total number of transfers per experiment (i.e. anomalous flows) is 40.

D. Datasets

The datasets used for our experiments with their collection period and number of flows in each dataset are shown in Table II. Dataset 1 was used only for training our model. Datasets 2 to 5 were used to test the accuracy of our model in predicting flow sizes on previously unseen data. Datasets 6 and 7 include the injected anomalous flows and were used to test the detection capabilities of Flowzilla.

Dataset	**Number of Flows**	**Duration**	**Year**
Dataset1	368,041	Oct 1 – Nov 30	2017
Dataset2	86,814	Jan 1 – Jan 15	2018
Dataset3	103,590	Jan 15 – Jan 31	2018
Dataset4		Feb 1 – Feb 15	2018
Dataset5		Feb 16 – Feb 28	2018
Dataset6	12,810	Mar 30 – Mar 31	2018
Dataset7	30,595	June 5	2018

TABLE II: Dataset composition

Since our training datasets only include scientific data tranfers rather than generic internet traffic, the variability in the input data is significantly lower [5]. As a result, anomalous events exhibit stronger signals that are more easily distinguishable.

E. Injected Anomalies Detection Results

Results for the synthetic anomalies experiments (datasets 6 and 7) are shown in Table III.

Experiment	**Total Anomalies**	**Anomaly Size**	**True Positives**	**False Negatives**
Large	40	10 GB	37	3
Average	40	1–5 GB	34	6

TABLE III: Results for synthetic anomalies

Our results show that for both experiments our framework is able to achieve a detection rate above 80% (92.5% and 85% for large and average anomalies respectively). However, in both experiments, our system erroneously identified a number of legitimate flows as anomalous (false positives).

Refining the threshold calculation mechanism by assigning weights on in μ (see Equation 2) could reduce the number of false positives and achieve higher detection rate.

Figure 4 shows the estimated (i.e., our model's prediction for the size of the flow) and actual size for the *large* anomalous flows that were injected in the first experiment. Obtained results show that our model is able to predict reasonably well the size of the anomalous transfers. However, the difference between the real and the predicted size of the anomalous flow needs to be bigger than the threshold value in order for the anomalies to be detected.

F. Cross Dataset Accuracy Results

Our model is designed to predict the size of scientific data transfers (i.e., flows) over a period of time. For evaluating the quality of our predictions, we test our model on new data obtained weeks and months after training and record the accuracy of the predictions. The datasets used for testing are Dataset2 to Dataset5. Results over a tenfold validation are shown in Figure 5.

We observe that the accuracy for three of the four datasets is maintained above 90% while for the dataset that was collected 6 weeks after training (Dataset3 in Table II) time the accuracy

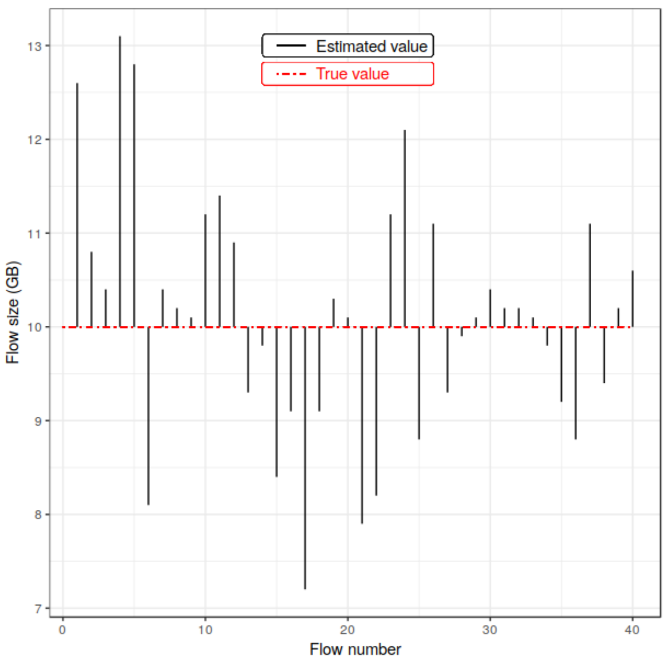

Fig. 4: Comparison of estimated (predicted by model) and actual sizes for 40 anomalous flows injected in the first experiment. Flowzilla is able to predict anomalous flow sizes reasonably well.

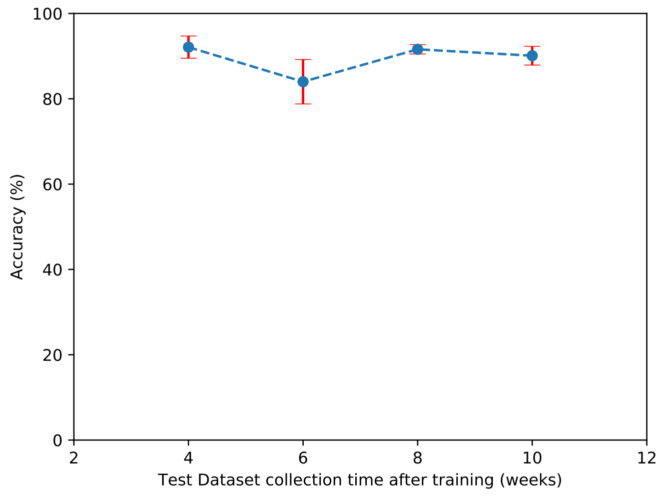

Fig. 5: Flow size prediction accuracy in datasets obtained up to 10 weeks after training. Flowzilla's accuracy remains above 85%.

is 84%. This is due to the fact that this particular dataset contains traffic collected between 15th to the 30th of January 2018 where the data volume towards the NERSC DTNs was significantly lower. However even in the case of unusual data transfer behavior our model's predictions are still more than 80% accurate. Overall, our results demonstrate that our model is able to provide accurate predictions over a 10 week period without requiring any retraining or dynamic update.

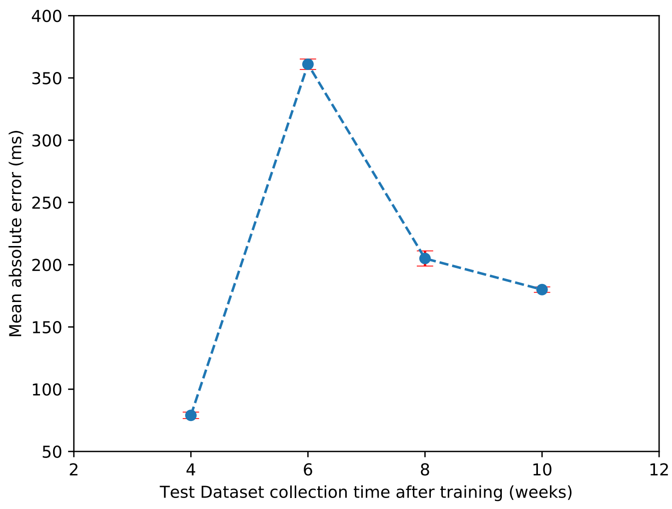

Fig. 6: Mean absolute error (mae) in datasets obtained up to 10 weeks after training. Mae is maximized 6 weeks after training where accuracy (see Figure 5) is lower.

The mean absolute error for the different datasets is shown in Figure 6. Again, for the dataset collected 6 weeks after training (Dataset2 in Table II), the average distance between our prediction and the test traffic is maximized (356 MB).

VI. Summary and Future Work

In this paper, we have addressed the problem of detecting volume-related anomalies in scientific data transfers. Our solution, *Flowzilla*, uses past data transfers in order to build a normal profile for data transfer volume. Then, using an adaptive threshold mechanism it flags flows that deviate significantly from the normal profile as anomalous and stores them in store/destination pairs. *Flowzilla* is able to provide an estimate for the amount of data being transferred in each anomalous flow.

We have evaluated the accuracy of our framework on network flows that represent real scientific data transfers. We injected anomalous flows of two different size classes and our results show that *Flowzilla* can detect volume related anomalies with a 92.5% detection rate. Our results show that a high percentage of detection rate is achievable in the context of scientific data transfers that exhibit lower variability than generic Internet traffic. We have demonstrated that anomaly detection is possible in the specific domain of scientific traffic. *Flowzilla* stores detected anomalous events and facilitates further analysis from network operators (e.g. NERSC).

Furthermore, *Flowzilla* can provide a reasonably accurate estimation of the size of the anomalous flows. Lastly, by testing the accuracy of our predictions on different datasets obtained up to 10 weeks after training, we have demonstrated that our solution is temporally stable.

Our ongoing work is centered around expanding our framework to cases which a specific anomaly involves multiple flows with different volume sizes. We plan to expand our evaluation process by calculating the detection rate of Flowzilla on datasets covering several weeks of scientific data transfers. Furthermore we would like to expand our solution in order to detect different types of anomalies in scientific data transfers by incorporating additional metrics in our model. For example, looking at the number of SYN packets packets over a specific flow would allow us to detect SYN-flooding attacks.We plan to evaluate the robustness of our solution in terms of detection accuracy with more detailed simulated anomalies, such as scraping a database or intentionally adding a misconfiguration in the path. Finally, we would like to experiment with different strategies regarding retraining frequency, including confidence intervals between the predicted and the actual data transfer volume.

Acknowledgements

The authors would like to thank numberous people for making this work possible: Josh Boverhof and Ravi Cheema for assisting with assisting with data access at NERSC; and Jon Dugan, Mariam Kiran, Eric Pouyoul, and Brian Tierney from ESnet for their helpful suggestions on the approach, data analysis, and interpretation of results.

This manuscript was authored by authors at Lawrence Berkeley National Laboratory under Contract No. DE-AC02-05CH11231 with the U.S. Department of Energy. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsors of this work.

This research also used resources of the National Energy Research Scientific Computing Center (NERSC), a U.S. Department of Energy Office of Science User Facility operated under Contract No. DE-AC02-05CH11231.

Appendix

Submission and reviewing guidelines, and methodology: *http://submissions.supercomputing.org/reproducibility*

A. Abstract

Research networks are designed to support high volume scientific data transfers that span multiple network links. Like any other network, research networks experience anomalies. Anomalies are deviations from profiles of normality in a science network's traffic levels. Diagnosing anomalies is critical both for network operators and scientists. In this paper we present Flowzilla, a general framework for detecting and quantifying anomalies on scientific data transfers of arbitrary size. Flowzilla incorporates *Random Forest Regression* (RFR) for predicting the size of data transfers and utilizes an adaptive threshold mechanism for detecting outliers. Our results demonstrate that our framework achieves up to 92.5% detection accuracy. Furthermore, we are able to predict data transfer sizes up to 10 weeks after training with accuracy above 90%.

B. Description

1) Check-list (artifact meta information):

- **Algorithm:** Anomaly detection leveraging the Random Forest Regression (RFR) algorithm from scikit-learn
- **Program:** Python
- **Compilation:** None needed
- **Data set:** Dataset description in Section V-D
- **Experiment customization:** None
- **Publicly available?:** Code can be made available upon request by contacting the authors of this paper, subject to export control review by Lawrence Berkeley National Lab (pending). The tstat data used was collected and provided by the NERSC computing facility at LBNL. The tstat data contains source and destination IP addresses, and so is not publicly available for privacy reasons. However, NERSC periodically makes data available to qualified researchers. Inquiries should be directed to *security@nersc.gov.*

2) How software can be obtained (if available): Code can be made available upon request

3) Hardware dependencies: None

4) Software dependencies: Required python packages: requests, sockets, elasticsearch, json, sys, os, re, datetile, ipaddress, numpy, sklearn.metrics, pandas, sklearn.ensemble. sklearn,model_selection, pandas

5) Datasets: See section V-D for detailed description

C. Installation

No installation for python script

D. Experiment workflow

See sections V-B, V-C

E. Evaluation and expected result

See Section V-E

F. Experiment customization

None needed

G. Notes

n/a

References

[1] P. A. Abell, J. Allison, S. F. Anderson, J. R. Andrew, J. R. P. Angel, L. Armus, D. Arnett, S. J. Asztalos, T. S. Axelrod, S. Bailey *et al.*, "LSST Science Book, Version 2.0," *arXiv*, 2009.

[2] E. Dart, L. Rotman, B. Tierney, M. Hester, and J. Zurawski, "The Science DMZ: A Network Design Pattern for Data-intensive Science," in *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, ser. SC '13. ACM, 2013, pp. 1–10.

[3] S. Axelsson, "Intrusion Detection Systems : A Survey and Taxonomy," Chalmers University of Technology, Tech. Rep., 2000.

[4] R. Sommer and V. Paxson, "Outside the Closed World: On Using Machine Learning for Network Intrusion Detection," in *Proceedings of the 31st IEEE Symposium on Security and Privacy*, May 2010.

[5] A. Marnerides, D. Pezaros, and D. Hutchison, "Internet traffic characterisation: Third-order statistics & higher-order spectra for precise traffic modelling," *Computer Networks*, vol. 134, pp. 183 – 201, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389128618300677

[6] "National Energy Research Scientific Computing Center," http://www.nersc.gov, accessed: 2018.

[7] I. Foster, "Globus Online: Accelerating and Democratizing Science Through Cloud-Based Services," *IEEE Internet Computing*, vol. 15, no. 3, pp. 70–73, May 2011.

[8] "Aspera High Speed Transfer Software," https://asperasoft.com, accessed: 2018.

[9] W. Allcock, "GridFTP: Protocol Extensions to FTP for the Grid," 07 2018.

[10] "Tstat- A statistic and analysis tool," http://tstat.polito.it/measure.shtml, accessed: 2018.

[11] "Cisco NetFlow," https://www.cisco.com/c/en/us/products/ios-nx-os-software/ios-netflow/index.html, accessed: 2018.

[12] Y. Gu, A. McCallum, and D. Towsley, "Detecting Anomalies in Network Traffic Using Maximum Entropy Estimation," in *Proceedings of the 5th ACM SIGCOMM Conference on Internet Measurement*, ser. IMC '05. USENIX Association, 2005, pp. 32–32.

[13] A. Lakhina, M. Crovella, and C. Diot, "Characterization of Network-wide Anomalies in Traffic Flows," in *Proceedings of the 4th ACM SIGCOMM Conference on Internet Measurement*, ser. IMC '04. ACM, 2004, pp. 201–206.

[14] "The Abilene Network," http://ots.utsystem.edu/pubs/internet2/3.html, accessed: 2018.

[15] W. Lu and A. A. Ghorbani, "Network Anomaly Detection Based on Wavelet Analysis," *EURASIP J. Adv. Signal Process*, vol. 2009, pp. 4:1–4:16, Jan. 2009.

[16] R. P. Lippmann, D. J. Fried, I. Graf, J. W. Haines, K. R. Kendall, D. McClung, D. Weber, S. E. Webster, D. Wyschogrod, R. K. Cunningham, and M. A. Zissman, "Evaluating intrusion detection systems: the 1998 DARPA off-line intrusion detection evaluation," in *Proceedings DARPA Information Survivability Conference and Exposition. DISCEX'00*, vol. 2, Jan 2000, pp. 12–26 vol.2.

[17] W. Hu, "Robust Support Vector Machines for Anomaly Detection," in *In Proc. 2003 International Conference on Machine Learning and Applications (ICMLA03)*, 2003, pp. 23–24.

[18] L. Khan, M. Awad, and B. Thuraisingham, "A New Intrusion Detection System Using Support Vector Machines and Hierarchical Clustering," *The VLDB Journal*, vol. 16, no. 4, pp. 507–521, Oct. 2007.

[19] A. Lakhina, M. Crovella, and C. Diot, "Diagnosing Network-wide Traffic Anomalies," in *Proceedings of the 2004 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, ser. SIGCOMM '04. ACM, 2004, pp. 219–230.

[20] "Sprint Network," https://www.sprint.com/en/landings/network.html, accessed: 2018.

[21] E. J. Candès, X. Li, Y. Ma, and J. Wright, "Robust Principal Component Analysis?" *J. ACM*, vol. 58, no. 3, pp. 11:1–11:37, Jun. 2011.

[22] A. Chhabra and M. Kiran, "Classifying Elephant and Mice Flows in High-Speed Scientific Networks," in *Proceedings of INDIS 2017*, 2017.

[23] S. Axelsson, "The Base-Rate Fallacy and the Difficulty of Intrusion Detection," *ACM Transactions on Information and System Security (TISSEC)*, vol. 3, no. 3, pp. 186–205, Aug. 2000.

[24] A. Liaw and M. Wiener, "Classification and Regression by randomForest," *R News*, vol. 2, no. 3, pp. 18–22, 2002. [Online]. Available: http://CRAN.R-project.org/doc/Rnews/

[25] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[26] S. Axelsson, "The Base-rate Fallacy and the Difficulty of Intrusion Detection," *ACM Trans. Inf. Syst. Secur.*, vol. 3, no. 3, pp. 186–205, Aug. 2000.

[27] D. Balouek, A. Carpen Amarie, G. Charrier, F. Desprez, E. Jeannot, E. Jeanvoine, A. Lèbre, D. Margery, N. Niclausse, L. Nussbaum, O. Richard, C. Pérez, F. Quesnel, C. Rohr, and L. Sarzyniec, "Adding Virtualization Capabilities to the Grid'5000 Testbed," in *Cloud Computing and Services Science*, ser. Communications in Computer and Information Science, I. I. Ivanov, M. van Sinderen, F. Leymann, and T. Shan, Eds. Springer International Publishing, 2013, vol. 367, pp. 3–20.