Xatu: Boosting Existing DDoS Detection Systems Using Auxiliary Signals

Zhiying Xu
Harvard University

Sivaramakrishnan Ramanathan*
USC/Meta

Alexander Rush
Cornell Tech

Jelena Mirkovic
USC/ISI

Minlan Yu
Harvard University

ABSTRACT

Traditional DDoS attack detection monitors volumetric traffic features to detect attack onset. To reduce false positives, such detection is often conservative—raising an alert only after a sustained period of observed anomalous behavior. However, contemporary attacks tend to be short, which combined with a long detection delay means that most of the attack still reaches and impacts the victim. We propose Xatu, a system that utilizes auxiliary signals to improve the accuracy and timeliness of existing DDoS detection systems. We explore two types of auxiliary signals, attack preparation signals and the history of prior attacks. These signals can be easily mined from existing traffic monitoring systems in many ISP networks. To leverage these auxiliary signals for attack detection, we propose a multi-timescale LSTM model, which derives both long-term and short-term patterns from diverse auxiliary signals. We then leverage survival analysis to quickly detect attacks when they occur while minimizing false positives and thus scrubbing costs. We evaluate Xatu on traffic from a large ISP, using commercial defense alert data to label prevalent attack events. Xatu would help the commercial defense scrub up to 44.1% additional anomalous traffic and would reduce its median detection delay by 9.5 minutes.[1]

CCS CONCEPTS

• **Networks** → **Denial-of-service attacks**; • **Computing methodologies** → *Machine learning*.

KEYWORDS

DDoS attack detection; Machine learning

ACM Reference Format:
Zhiying Xu, Sivaramakrishnan Ramanathan, Alexander Rush, Jelena Mirkovic, and Minlan Yu. 2022. Xatu: Boosting Existing DDoS Detection Systems Using Auxiliary Signals. In *The 18th International Conference on emerging Networking EXperiments and Technologies (CoNEXT '22), December 6–9, 2022, Roma, Italy*. ACM, New York, NY, USA, 17 pages. https://doi.org/10.1145/3555050.3569121

*Work partially done while visiting Harvard.
[1]Code at https://github.com/harvard-cns/Xatu.

CoNEXT '22, December 6–9, 2022, Rome, Italy

ACM ISBN 978-1-4503-9508-3/22/12…$15.00
https://doi.org/10.1145/3555050.3569121

1 INTRODUCTION

Operational defenses against distributed denial-of-service (DDoS) attacks often detect the start of anomalous traffic based on volume-based or duration-based thresholds [21, 23, 76, 84]. Traffic is then sent to scrubbing centers, which typically attempt to identify and remove attack flows using threshold-based approaches. However, these approaches can be easily gamed by attackers, usually by launching short attacks [22, 38, 63]. For example, the work [63] reports that 63% of DDoS attacks are shorter than 5 minutes and 77% are less than 10 minutes. Thus, by the time a detection system triggers scrubbing, the attack has already ended.

DDoS detection could try to be more sensitive, to detect attacks sooner, but higher sensitivity will lead to false positives. Benign traffic can be bursty, and triggering mitigation too often can be costly. It is difficult to achieve accurate and timely DDoS mitigation just based on *incoming traffic volume*. We need *additional signals* to understand when traffic increase is due to an impending attack, and when it is simply due to fluctuating traffic patterns.

In this paper, we propose to *boost* existing, traffic-based DDoS attack detection with auxiliary signals, which capture attack preparation and attack history. In many cases, with auxiliary signals, it is possible to more accurately discriminate between benign traffic spikes and the attack, and trigger timely mitigation. Our system does not replace existing, traffic-based detection, but simply boosts it, increasing its accuracy.

Our key observation is that in practice attackers often repeat the same types of attacks on the same victims, they leverage spoofing and reuse previously compromised hosts in attacks. While it is certainly possible to generate attacks that do not exhibit these traits (e.g., fully random, always using new bots, and not spoofing), such a practice would be too costly for attackers and is thus not likely. In this work we leverage *auxiliary signals*, which reflect current attack practices, to boost DDoS detection. Attackers need many preparatory steps to launch large-scale, high-volume attacks. They must launch servers (e.g., purchasing or establishing botnets), explore vulnerabilities (e.g., experimenting with different types of attacks), test software (e.g., start with small traffic or attacks), and they often deploy the attack in phases (e.g., gradually increase attack rate or the number of attackers) [3, 7, 16, 19, 42, 73]. These behaviors can be captured by auxiliary signals, mined from sampled NetFlow data, which many service providers already collect.

We identify two types of auxiliary signals: *(1) Attack preparation signals*, which can signal an impending attack – an increase in incoming traffic to a customer from spoofed sources, from sources that appear on public or private blocklists, and from sources of previous attacks. Large network providers are in a unique position

to observe these attack preparation signals because they serve many customers (potential attack targets) and see all or majority of their customers' incoming traffic. *(2) Attack history*, which includes previous instances of different types of attacks, start time, and their victims. Because network providers often deploy a commercial or home-grown DDoS attack detection, their past detection signals can be leveraged to establish models of attack behavior for a given attack source or group of sources, and for a specific customer, or a group of related customers.

Our auxiliary signals are only weakly correlated with attacks, may happen long before attacks, and are sometimes noisy or incomplete (e.g., inaccurate blocklists). The research challenge lies in how to best leverage these signals to boost detection, without creating false positives. We propose Xatu, a framework that leverages machine learning over several time periods on rich auxiliary signals, along with volumetric signals, to boost DDoS detection. As an add-on boost to traffic-based DDoS detection, Xatu is easier to deploy and costs less than a complete replacement of current detection systems. In addition to the choice of features used to boost detection, there are two key challenges in the choice of the appropriate machine learning techniques:

Integrating long-term and short-term signals with multi-timescale LSTM. Diverse types of auxiliary signals can be observed in many days preceding an attack, and these auxiliary signals can be weak, intermittent, and variable. Xatu chooses a highly-parameterized LSTM model to learn complex time-series representations even from multiple types of weak auxiliary signals. To extend this model to work with many features from multiple days of data over tens of thousands of time steps, Xatu uses a multi-timescale LSTM [12] to mine information from auxiliary signals at multiple granularities from minutes to hours. This approach allows Xatu to learn to model complex properties of the input that occur both minutes and days before an attack.

Balancing early detection accuracy and cost with survival analysis. Our goal is to produce accurate estimates of an upcoming attack, at the right time, because a detection that is too early can increase scrubbing costs. Xatu employs a survival analysis model that describes the start time of an attack with a cumulative probability of the attack's occurrence. We use a loss function [89] that calibrates attack probability to balance early detection accuracy with detection time. The loss function enables early detection by maximizing the likelihood of detecting at any time before ground-truth detection, but minimizing the likelihood of making the detection throughout non-attack periods. This setup takes into account the process of developing attack events; a process that is ignored in pure classification systems.

We evaluate Xatu on a unique dataset from a large network provider, spanning 100 days and 18.5 TB of sampled Netflow data, accompanied by DDoS detection signals of 6 prevalent attack types from a state-of-the-art commercial DDoS defense – Arbor NetScout (See §2.1 for more details). Xatu detects attacks 9.5 minutes earlier than NetScout by median and only increases traffic sent to scrubbing by 0.1% over the ideal case. Early detection capabilities of Xatu improve the percentage of anomalous traffic sent to scrubbers, i.e. effectiveness, from 62.6% to 90.2% by the median. Xatu is robust to attackers that are capable of changing attack volume and rate to avoid NetScout's detection. Xatu also works well with different attack detection systems, such as FastNetMon[64].

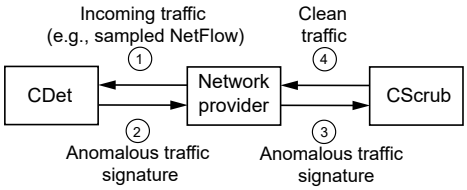


Figure 1: CDef consists of CDet that is used for attack detection and CScrub that is used for filtering.

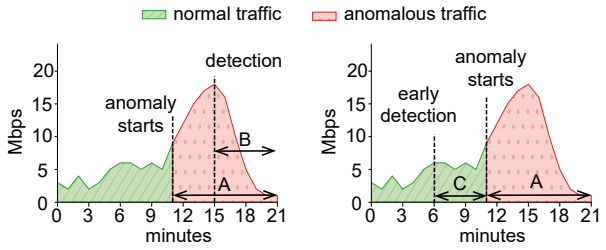
2 EARLY DDOS ATTACK DETECTION

2.1 Commercial DDoS Defense

Figure 1 shows a typical commercial attack defense infrastructure [26, 34, 38, 39, 56, 58, 81] (denoted as *CDef*) at network providers. The infrastructure consists of commercial attack detection appliances (denoted as *CDet*) and scrubbing centers (denoted as *CScrub*). CDef monitors either all or sampled incoming traffic (e.g., using NetFlow), and generates attack alerts and signatures to identify anomalous traffic (Step ① and Step ②). The attack signatures usually specify the traffic's destination address or prefix, transport protocol, and source and/or destination ports. For instance, Figure 2(a) shows an example of a UDP attack detected by CDet deployed in a large regional network provider. CDet detects an attack at 15 minutes and generates a signature identifying the victim's destination address, transport protocol as UDP, and source port number as 53. CDet uses such coarse-grained signatures to redirect anomalous traffic to CScrub for further inspection (Step ③ in Figure 1). CScrub identifies attack flows, using proprietary algorithms, and drops them, while it delivers clean (legitimate) traffic back to the network provider and its customers (Step ④ in Figure 1). Once the attack is mitigated, CScrub alerts customers to stop diverting their traffic for scrubbing. CScrub incurs costs based on the total amount of traffic it handles [32, 34, 56, 77]; this cost is passed on to the customers. A network provider may host CDet and CScrub in their network (e.g., buy a NetScout appliance), or outsource one or both of these functionalities to the cloud and/or a third-party defense provider (e.g., Cloudflare).

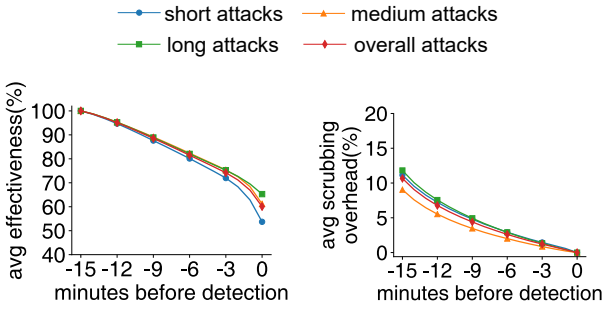
2.2 Our Dataset from a Large ISP

For an illustration of DDoS defense challenges in this Section and for evaluation in §6, we use a dataset from a large ISP with a global presence. The ISP serves more than 1,000 customer networks from all over the world, including North America, Asia, and Europe. These networks include telecom, healthcare, financial, online shopping sites, government, and education sectors, thus traffic that this ISP transit is very diverse and high volume (1 B packets per second). Our dataset includes sampled NetFlow records from all POPs in the ISP network, for a total of 18.5 TB of data, covering 100 days. The ISP also runs a popular commercial defense—NetScout [62]. Our dataset includes more than 10 K attack alerts from NetScout.



(a) Detection of attack by CDet. (b) Detection Boost of attack.

Figure 2: An example UDP attack describing the start of anomalous traffic and CDet detection. A: anomalous traffic; B: anomalous traffic sent to CScrub; C: extraneous traffic sent to CScrub.



(a) Low effectiveness of existing CDet. (b) Increase in scrubbing overhead due to early detection.

Figure 3: Challenges faced by existing CDet.

2.3 Problems of Late Detection

Today, both commercial [62, 64] and academic [20, 48, 70, 88] DDoS defenses are reactive in nature and can only detect an attack after it has established clear anomalous traffic patterns. For operational reasons, detection must be conservative to avoid sending too many alerts to operators and to reduce scrubbing costs. As a result, CDet can incur late detection and, in case of short-lived attacks, CDet may divert only a small portion of anomalous traffic[2] to CScrub.

We quantify the problem of late detection with *mitigation effectiveness* (effectiveness for short) measure, denoting the portion of *anomalous* traffic that is sent to CScrub by CDet. To define this metric, we leverage CDet's detection signals and anomaly (attack) signatures to go back in time and analyze all traffic matching the signature, prior to detection. We then identify the onset of the anomaly by looking for the sudden, sustained increase in the matching traffic, using CUSUM algorithm [10, 27] (see Appendix A). This is illustrated in Figure 2(a) for one attack in our dataset, which was a UDP flood. The attack was detected at 15 minutes after an hour. We analyze traffic prior to the attack and collect the volume of UDP traffic going to the attack's victim. There is a clear increase in UDP traffic at 9 minutes, and we denote this as "anomaly start" and Area A becomes our ground truth for attack duration. CDet detects the attack at 15 minutes when the traffic exceeds thresholds from CDet, and then all traffic is sent to CScrub (Area B). We define effectiveness as the percentage of A that is sent to CScrub, i.e., B/A.

CDet has low effectiveness. To motivate our work, we analyze all alerts generated by CDet in our dataset. Nearly 74% of attacks are shorter than 20 minutes, and nearly 75% of attacks have peak anomalous traffic lower than 21Mbps. Late detection becomes a critical issue as attacks become stealthier, with shorter duration and lower volumes, leading to late scrubbing. Late detection not only hurts the customer experience but can also cause SLA violations and revenue loss for provider networks [80]. Therefore, there is a clear need to boost early attack detection and divert more of the attack traffic to CScrub.

[2]We denote traffic matching the attack detection signature as *anomalous*. This may be a mix of legitimate and attack traffic.

2.4 Cost of Early Detection

One could detect attacks earlier, either by increasing the existing detection system's sensitivity or, as we propose, by *boosting* traffic-based detection with auxiliary signals.

Early detection could help redirect most or all anomalous traffic (area A in the Figure 2(b)) to CScrub, increasing defense effectiveness to its ideal 100%.

However, early detection could also occur much before an attack (before area A), or it could be erroneous – detecting an attack when it does not occur. In those cases, it would send *additional, extraneous* traffic to CScrub, exceeding the ideal case where traffic is only scrubbed during an actual attack. Additional scrubbing incurs additional, unnecessary cost[3], compared to an ideal case. We define the cost metric *scrubbing overhead* (overhead for short), as the ratio of extraneous traffic sent to CScrub (area C in Figure 2(b)) divided by anomalous traffic (area A), i.e., C/A.

We report cumulative overhead per customer (cu) of a network provider, over multiple attack instances (at), i.e., $\sum_{at} C_{cu}^{at} / \sum_{at} A_{cu}^{at}$.

Figure 3(a) and Figure 3(b)show the trade-off between overhead and the effectiveness of CDet. We assume a naïve, early detection where *all attacks* are uniformly detected N minutes before the actual CDet's alert. Since both effectiveness and overhead depend on the attack's duration, we show the results for short, medium, and long attacks separately. On average, uniformly alerting 15 minutes before CDet would achieve ideal effectiveness (100%) but at the cost of sending 8–12% more traffic to CScrub than in the ideal case. Conversely, if detection occurred 3 minutes before CDet, the effectiveness would drop to 74.8%, but the overhead would also drop to 1.2%. The improvement in effectiveness is largest for short attacks (< 5 min), which are currently mitigated only 50% of the time (x=0 point in Figure 3(a)) because CDet reacts slowly to their unexpected increase in anomalous traffic. The cost of early alerts is the largest for long attacks (green line in Figure 3(b)), because they also include significant traffic in the preparation phase (> 15 min). Obviously, there are advantages to early attack detection, but sophistication is needed to balance effectiveness with overhead.

[3]CScrub either charges customers based on peak traffic during an attack or based on the amount of legitimate traffic sent back to the customer. In both cases, running CScrub can be expensive [33, 34, 55].

2.5 Early Attack Detection Goal

Our goal is to boost attack detection early enough to maximize effectiveness while keeping the overhead small. To achieve this, we need to improve the accuracy and timeliness of attack detection using our proposed system – Xatu.

We improve timeliness by leveraging auxiliary signals (regularities in attack events and information about source past behaviors) in addition to the traffic volume, to train powerful, multi-timescale machine learning models. Such models help us detect attacks early, with high accuracy.

We keep overhead metrics small by using survival analysis to bound the overhead of early detection during training. This teaches our model to optimize our alert time while maintaining high detection accuracy.

2.6 Xatu Deployment

We envision that Xatu would be deployed in a network provider, which already has a DDoS detection system, and wants to boost its accuracy and timeliness. The provider would continuously feed their sampled NetFlow data, some publicly available data (e.g, public blocklists), and attack alerts from their existing DDoS detection system into Xatu in real time. Xatu is not in the line of traffic, it operates passively on data feeds.

When Xatu detects an attack early, it would provide new attack alerts, which precede those eventually generated by the existing detection system. These alerts would be used in an automated fashion by the network provider to trigger attack mitigation and direct traffic matching the alert signature to CScrub for filtering. CScrub monitors the anomalous traffic and stops Xatu's detection when an attack is fully mitigated. Our envisioned deployment is lightweight, and only requires data exchange between existing software and hardware. Thus Xatu is highly deployable.

3 AUXILIARY SIGNALS

To boost attack detection, we propose to leverage *auxiliary signals* to confirm or discount anomalies observed in traffic volume to a customer. Auxiliary signals are traffic signals that we can observe well before the attack (e.g., in the preceding 10 days). The presence of these signals can indicate attack preparation and early phases of attack launch, and it can increase confidence that observed increases in traffic volume to a customer indicate attacks and not benign fluctuations.

Intuition: which auxiliary signals and why. Similar to other software development processes, an attacker may need many preparations to identify the best resources and strategies to launch effective attacks. Attackers need to infect and coordinate bots, bootstrap scanning, and develop and test attack software. They may also launch attacks in phases to gradually increase the attack spread and volume. For example, a study on Mirai attacks [3] showed that the preparatory phase involved scanning to discover vulnerable IoT devices during the seven months preceding peak attacks launched by Mirai. Similar findings were also reported for other DDoS attacks [16, 19, 42].

Attackers often attack their targets repeatedly[7, 73] and may attack related targets in the same organization. These patterns occur due to the attackers' business model. They may attack a customer repeatedly to extort money, or to impact some service, which is particularly vulnerable at certain points in time (e.g., when its utilization is high). They may attack several customers simultaneously to bring down a collaborative service, such as a distributed Web application.

In this Section, we describe five types of auxiliary signals that Xatu uses, demonstrate their correlation with future attacks, and highlight the boosting benefits and challenges of using these signals for early attack detection. Our proposed auxiliary signals are easy to observe and inexpensive to maintain, as they either are publicly available or use simple statistics, which networks regularly collect.

Some attacks may not be preceded by auxiliary signals, as attackers may be familiar with our approach and may seek to evade it. We show in §6.4 that Xatu is robust to some evasion attacks. But ultimately, a determined attacker may make their attacks sudden and random to evade our detection. We discuss in §8 that this is possible, but unlikely, since it goes against the attackers' business model.

3.1 How to Leverage Auxiliary Signals?

The main challenge in leveraging auxiliary signals is that they are *weak and incomplete*. Signals are *weak*, because they may be present even when no attack is imminent. Thus the presence of auxiliary signals is not sufficient to detect DDoS attacks, we must also leverage volumetric traffic signals. Auxiliary signals simply help *boost* the volumetric signals, so attacks can be detected earlier before they have a large impact on a customer.

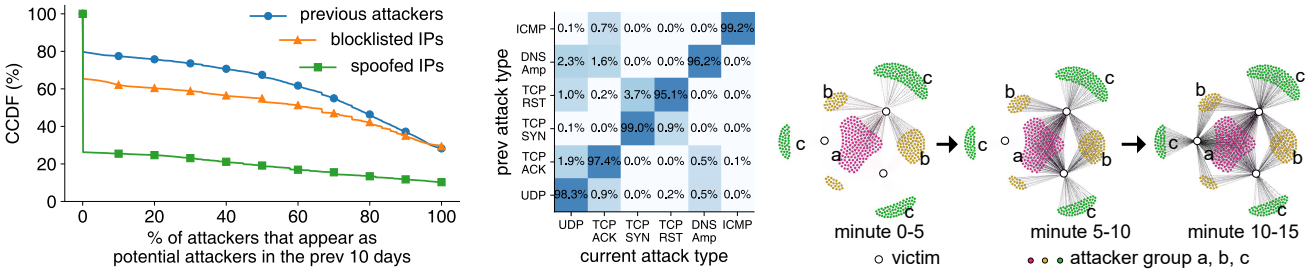
Auxiliary signals are *incomplete*, because they may not be accurately measured, nor comprehensively observed. For example, in many cases, it is difficult to establish if a given traffic source is spoofed or not. Similarly, many public blocklists are incomplete and may be inaccurate.

Because auxiliary signals are weak and incomplete, it is hard to design a deterministic approach to leverage them. Instead, we use machine learning to build models that capture auxiliary signals in both the long and short term, and capture the predictive strength of each signal with regard to a future DDoS attack. The intuition is that machine learning can be used to boost weak signals, and detect their correlations, which helps when looking for the needles in a haystack of high-dimensional datasets. The other known benefit of ML is that it can identify sophisticated temporal patterns across a long period of history (e.g., 10 days), as well as a few minutes before an attack.

3.2 Attack Preparation Signals

We define three types of easy-to-monitor auxiliary signals based on the activities that indicate the preparation of an attack. These signals include the activity of traffic sources that appear in public blocklists, the activity of sources of previous attacks on the same target (detected by deployed CDet), and the activity of spoofed traffic sources. We leverage not only the existence of these signals (e.g. blocklisted sources), but also their traffic patterns, e.g. increase in traffic volume from a specific type of source to a potential target.

(1) Blocklisted sources (A1). Many networks and security companies publish lists of past offenders as public blocklists [35], and many other networks leverage these lists to identify suspicious



(a) Percentage of attack, blocklist, and spoof IP addresses within the 10 days leading up to an attack

(b) Transition of attack types.

(c) Shared attackers toward 3 customers. During this time, attacker groups *a*, *b*, and *c* attack 3, 2 and 1 customers, respectively.

Figure 4: Attack preparation signals and attack history.

traffic. Tracking activities from blocklisted sources can alert us to new attacks in preparation.

(2) Sources of previous attacks (A2). Sources that participated in a previous attack on the same or related target may be involved in future attacks. We can view these past sources as a special blocklist [36], customized to a given customer.

(3) Spoofed sources (A3). IP spoofing is a common technique used in DDoS attacks to evade detection [6, 15] and is still widespread [49]. An increase in traffic carrying spoofed IP addresses [14, 15, 54, 59] can potentially indicate an incoming attack.

Xatu uses observations of traffic from blocklisted sources, previous attack sources, and spoofed sources in the given time period, prior to an attack, as auxiliary signals to boost future attack detection. These signals are easy to monitor and maintain. Public blocklists are frequently updated and accessible; previous attackers can be maintained by monitoring CDet alerts and obviously spoofed traffic (e.g., from private IP ranges) can be detected by monitoring incoming traffic. Besides, escaping auxiliary signals is expensive for attackers, as they would need to recruit new sources all the time and elaborately change their attack strategies. There are overlaps of addresses between three sources and we include an address in all the categories it belongs to. We detail our chosen features for each signal in §5, and evaluate the optimal length of the observation period in Appendix H.

Potential attack sources are active up to 10 days before the attacks. Figure 4(a) shows the percentage of attacks in our dataset, where sources have previously appeared on public blocklists, have previously attacked the same target, or are obviously spoofed. The x-axis shows the percentage of actual attackers that belong to blocklisted, previous attackers, or spoofed groups of sources, while the y-axis shows the percentage of attacks. There are blocklisted sources, previous attackers, and spoofed sources that get converted to actual attackers in 65.7%, 80%, and 26.3% of attacks. About 50% of attacks have more than 67.5% of attackers that have previously attacked the same customer. Similarly, 50% of the attacks have at least 54.9% of attackers that have appeared on blocklists, and 19.1% of attackers that are spoofed addresses. The percentage of spoofed traffic is relatively smaller because we only identify obviously spoofed traffic (see §5.2). Apart from the appearance of potential attack sources, their activity also increases closer to the attacks with more details in Appendix B.

Potential attack sources are weak auxiliary signals. Although the presence of potential attack sources is correlated with future attacks, their appearance does not guarantee future attacks. For example, in our dataset, on average in 95.5%, 86.3% and 82.8% of cases, traffic from blocklisted sources, previous attackers, or spoofed sources in the preceding 10 days did not result in a new attack on the same target. Further, potential attack source information is imperfect. Blocklists may miss some repeat offenders or list benign sources, and spoofing is notoriously hard to detect.

Takeaway: We need to carefully leverage the imperfect and weak information about the activity of potential attack sources to boost the detection of future attacks on a customer.

3.3 Attack History

Specific customers may be vulnerable to specific types of attacks, or some attacks may be easier to launch than others.

Previous attacks to the same customer (A4). Figure 4(b) shows how often a given attack type follows other attack types *on the same customer*. Most transitions occur between the same attack types. Among the 43.8K consecutive attack pairs in our dataset, 43.0K (or 97.9%) include attacks of the same type. This is especially true for UDP floods and TCP ACK floods. 98.3% of UDP floods are followed by another UDP flood, and 97.4% of TCP ACK attacks are followed by a TCP ACK attack. Such behavior makes sense – if the first attack is successful, the attackers may launch it again.

Attackers could systematically launch various types of DDoS attacks over time to explore vulnerabilities and estimate which attack would have the highest impact on a target. For instance, TCP SYN attacks are sometimes followed by TCP RST attacks (3.7%) in Figure 4(b). One potential reason is that both attacks target the same resource of TCP packet processing, and both packet types are frequent in regular operation and thus may bypass firewalls. Another example in Figure 4(b) is that 2.3% of DNS amplification attacks and 0.1% of ICMP attacks are followed by UDP flood attacks. This may be because these attacks target incoming network bandwidth. If these attacks are successful, attackers may add other volumetric traffic in future attempts.

Takeaway: We can leverage the knowledge of previous attack types and occurrences to boost the detection of the future type and time of the attack on a given customer.

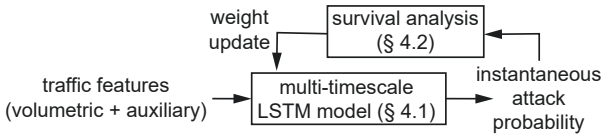


Figure 5: Xatu architecture.

Correlated attacks across customers (A5). The same set of attackers may attack different customers over time, because attackers may leverage the same bots [73]. For example, in Figure 4(c), the attacker group *a* starts by attacking one customer but later moves to two other customers over the next 15 minutes. The attacker group *b* also attacks additional customers over the 10-minute interval. Note that some attack groups target a subset of customers, and avoid others. Another Figure 16 in Appendix B also illustrates these correlated attack occurrences by showing the change in clustering coefficient [43].

Takeaway: We can leverage the knowledge of customers that were attacked in the past by overlapping attacker groups to boost the detection of future correlated attacks.

4 XATU'S MACHINE LEARNING

As shown in §3, auxiliary signals may indicate anomalous traffic as early as 10 days before an attack. For instance, A3 auxiliary signals are present at most ten days before many attacks. However, while auxiliary signals are useful, not all auxiliary signals will lead to anomalous traffic, and acting too early to reroute traffic can increase overhead (see §2).

The key design goals of Xatu are as follows: (1) derive both long-term and short-term benefits from diverse auxiliary signals and (2) enable early detection to improve effectiveness, while keeping the overhead low. As shown in Figure 5, we achieve goal (1) with a multi-timescale LSTM model (§4.1), and goal (2) with a survival analysis model (§4.2). The multi-timescale LSTM model takes in both long-term and short-term traffic features and generates an instantaneous attack probability. The survival analysis model then updates weights in the multi-timescale LSTM model based on the instantaneous attack probability, with a goal of early detection.

4.1 Model Different Time Scales

Xatu learns a time-series model for the probability of the start of an attack. It uses both standard volumetric signals, as well as auxiliary time-series signals. Table 1 shows the full list of 273 input features that Xatu uses, extracted from both volumetric and auxiliary signals (see §5 for more details). We use a machine learning model to integrate these multiple types of signals, as described in §3.1. Given that auxiliary signals may be present days before an attack (§3), and become strongest in the short period before an attack, we want a machine learning model that leverages both the long-term (e.g., 10-day) time series of auxiliary signals as well as recent auxiliary and volumetric signals.

Machine learning models such as random forests (RF) [46] and convolutional neural networks (CNN) [44] have been commonly used for the early detection of anomalous events. However, these models are less suitable for long-term time-series data. RFs regard each feature in the input equally and struggle to capture trends; while CNNs struggle at capturing features at varying timescales. Long Short-Term Memory (LSTM) [24] networks are specifically tailored to model time-series inputs. Compared to other RNNs, LSTMs are designed to retain historical trends over features with their special "memory cell," while maintaining a bias for recently observed features. However, as the length of the time-series data increases, the detection capabilities of LSTM can decrease [24]. For example, if Xatu were to monitor the signals discussed in §3 at 1-minute granularity, the length of time-series data would be 14K for a 10-day period. Such a very long sequence length may reduce LSTM's detection capabilities.

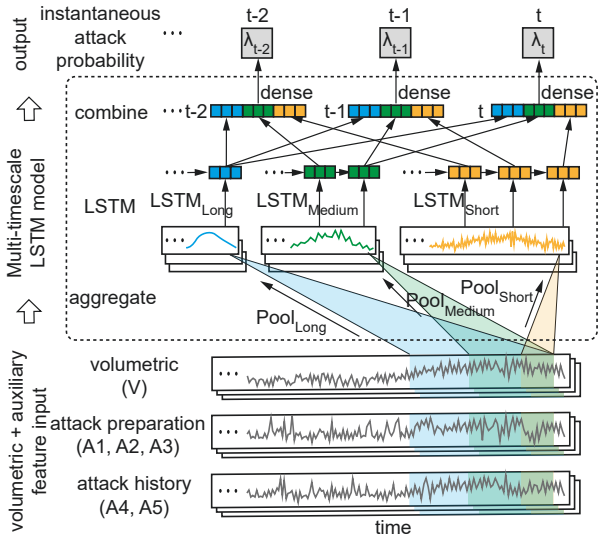


Figure 6: Multi-timescale LSTM model for processing auxiliary signals.

Our basic learning unit consists of three LSTM models as shown in Figure 6. We use a multi-timescale LSTM approach to aggregate the many features of interest at different time granularity [12], which provides both a fine-grained, zoomed-in view and a high-level view of each feature's dynamics. The former records the detailed recent trend of features towards the target customer and allows the model to make an accurate time prediction on the next attack. The latter provides an overview of long-term feature trends, including shifts during a day and a week, toward the target customer.

Xatu applies three different 1-dimensional aggregation (pooling) layers at different time granularity. We define the three aggregated time-series as TS_{Short} (1 minute), TS_{Medium} (10 minutes) and TS_{Long} (60 minutes). These are input to three different LSTM models—$LSTM_{Short}$, $LSTM_{Medium}$ and $LSTM_{Long}$. Xatu then combines the outputs from the three LSTM models using a dense layer. The final output is used to model the probability of an attack.

4.2 Survival Analysis for Early Detection

The goal of Xatu is to accurately boost the detection of anomalous traffic, while keeping the overhead low. Unlike anomaly detection approaches that alert after an event has occurred [18, 21, 23], we would like Xatu to boost detection based on *early* signals, to almost

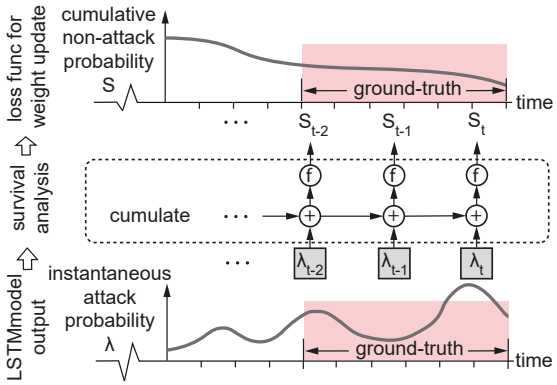


Figure 7: Survival analysis for boosted detection.

predict an attack. Therefore, we need to find the right weights inside the multi-timescale LSTM model for early detection. We would also like *consistent* detection, i.e. once an attack is detected for time t, for a given customer with a given signature, we should continue the same detection at all times after time t until the attack is mitigated by CScrub.

A naïve approach is to directly use the instantaneous attack probability from the output of the LSTM model to raise attack alert. However, this approach fails to meet either of our goals, as shown at the bottom of Figure 7. The instantaneous attack probability focuses too much on the detection for the current minute. Training with such an approach primarily raises the attack probability for a current minute and lowers it for preceding minutes. Therefore, the approach tends to detect attacks to mirror the ground truth in the training set, i.e. when the attack has already occurred. Moreover, the attack probability can be inconsistent over time. If a signal fluctuates, attack probability will fluctuate too, which is undesirable.

To achieve accurate early detection, Xatu employs survival analysis [41]. Survival analysis is commonly used to model the onset (or lack of onset) of an event in time. For Xatu, the event is the beginning (or the continued absence) of anomalous traffic. We introduce a cumulative probability S_t, which is the probability that an attack has not occurred by time t, i.e. $\Pr(A \geq t)$ where A is the attack. Compared to instantaneous attack probability, this cumulative probability depends on the start time t of an attack, and better fits our joint goals of early and consistent detection—predicting an onset of a future attack. Xatu estimates S_t within a detection window $t = 1, 2, \ldots, n$, and detects the attack if S_t falls below certain threshold.

We train Xatu to be aware of the early detection objective. We utilize the SAFE loss function [89], which is targeted for survival analysis, to train the weights in the LSTMs and the dense layer. This loss function seeks to minimize S_t for periods when there is an attack in training and maximize S_t for other periods. In particular, for time series with attack labels, we maximize the likelihood of detecting at any time before ground-truth detection; for time series with no attack labels, we maximize the likelihood of non-detection at each step. This loss function can be computed for each of the individual times S_t, and then propagated through the model. More details are in Appendix C. When using the model, our goal is to calibrate our detection to achieve low scrubbing overhead. Therefore, during validation, we select an overhead bound and search for a threshold on S_t that maximizes the effectiveness, while keeping overhead within the bound we have set. We use this threshold during our evaluation.

5 XATU PROTOTYPE

5.1 Datasets

Traffic. Our study uses NetFlow data from a large ISP, as detailed in §2.2. The provider has shared with us NetFlow data, collected for 100 days between April and August 2019. Overall there are 18.5 TBytes of sampled NetFlow data between the network customers and the Internet. This one-minute exportation delay is relatively small compared to the potential early detection boost (9.5 minutes in §6.1). The sampling rates range from 1:1 to 1:10,000 at various routers.

Alerts and Ground Truth. Xatu uses alerts from an existing DDoS defense – Arbor NetScout – to label ground-truth events during training and validation. We also use these alerts to label ground-truth events during evaluation, to understand Xatu's effectiveness and scrubbing overhead. We focus on the most prevalent six types of attacks: UDP flood, TCP ACK, TCP SYN, TCP RST, DNS Amplification, and ICMP flood (Table 2), which cover 97.2% of all alerts in our dataset. Among these attack types, the DNS Amplification attack is the only reflection amplification attack and takes up 7.2%.

Auxiliary signals. For A1 auxiliary signal (blocklisted sources), we select 11 categories of blocklists (containing a total of 151 public blocklists [69]) maintained by various providers. The selected blocklists collect sources of DDoS attacks, reflection attacks, VoIP attacks, and C&C servers and bots that are infected with specific malware families. We convert all the IP addresses and subnets in these blocklists to /24 subnets. This is a standard approach to improve the effectiveness of blocklists as used in previous work [8, 11, 69] due to dynamically managed IP address space. We collect the blocklists during the same 100-day period as our dataset. Note that blocklisted addresses may miss some offenders [69] and may contain legitimate addresses.

For A3 auxiliary signal, we determine spoofing by using three categories of IP addresses [47]: (a) Bogon source addresses defined in RFC1918 [71], RFC5738 [13], and RFC6598 [82] (b) unrouted source addresses not covered in BGP datasets [60, 65] (c) invalid source addresses not originated from the AS that announces the corresponding prefix or the AS' full cone with adjustments for multi-AS organizations. Note that this spoofed traffic measure is imperfect because identifying spoofed traffic is a challenging topic itself, especially with sampled NetFlow records [87]. We likely miss much-spoofed traffic.

Finally, for A2, A4, and A5 auxiliary signals, we determine previous attacker addresses by identifying all sources of traffic matching the alert signature for the time from the CDet's alert to the CDet's mitigation-end notice. This is an imperfect signal since a legitimate source could send matching traffic to the customer during an attack.

We also investigated other auxiliary features such as IP addresses involved in scanning, the correlation between attack targets, and

Type	Feature Description	#
V. volumetric	unique source nodes	1
	mean, max of traffic†	4
	UDP, TCP and ICMP traffic†	6
	traffic from popular source ports†	10
	traffic to popular destination ports†	10
	traffic with different TCP flags†	12
	traffic from 10 popular countries†	20
A1. blocklisted sources	volumetric features from blocklisted sources	63
A2. previous attack sources	volumetric features from previous attack sources	63
A3. spoofed sources	volumetric features from spoofed sources	63
A4. previous attack to same customer	attack severity (low, medium, high) for each attack type	18
A5. correlated attacks	three techniques (dot, min, max) to obtain clustering coefficient	3

Table 1: 273 features extracted from CDet alerts and NetFlow dataset (A = auxiliary). (†) are measured in bytes and packets.

other protocols and ports. We choose to leave them out, as they did not significantly improve performance.

5.2 Features

Table 1 shows a set of volumetric and auxiliary features extracted from the NetFlow and CDet alerts that describe the traffic and attack behavior towards each customer IP address. We extract 273 features and broadly group them into volumetric and auxiliary feature sets.

The *volumetric feature* sets capture traffic features, such as traffic rates towards a customer, both in total and disaggregated per transport protocol, popular port numbers, TCP flags that are associated with different TCP attacks, and countries of traffic sources (see Appendix D). The A1, A2, and A3 *auxiliary features* capture the same volumetric features for a subset of the flows that are sent by blocklisted addresses, previous attackers (for the same customer), and spoofed addresses. The A4 auxiliary signal captures the attack history on the same customer and the A5 auxiliary signal captures the attack correlation with other customers.

Some of our feature sets need attack detection signals, such as previous attacker (A2), attack history (A4), and attack correlations (A5). We use the existing NetScout alerts to extract these attack-dependent features during the training and validation period. However, for stabilization and testing periods, we rely on Xatu's detection to extract these features, to make Xatu independent of the existing defense.

5.3 Prototype Implementation

When deployed in a network, Xatu takes in sampled NetFlow and provides anomalous traffic signature in parallel with Cdet in Figure 1. The model can be trained offline with manually verified ground truth or detections from commercial DDoS detection systems, while the testing phase is real-time.

To form the training data, we select an equal number of attack and non-attack time series based on CDet alerts for training. For each time series, Xatu extracts time series of volumetric and auxiliary features in Table 1 in the past 10 days. For every minute of original NetFlow data (100 MB in our dataset), we can extract all the features within an average of 50 milliseconds per customer on a single thread of Intel Xeon CPU E5-2680 v3 machine.

We then use the feature data as input to our machine learning model, which includes two parts:

(1) Multi-timescale LSTM model. For each attack (or non-attack) time series, the model first downsamples the feature data into three time granularities — TS_{Short} as 1 minute, TS_{Med} as 10 minutes, and TS_{Long} as 1 hour. The three time series then feed into $LSTM_{Short}$, $LSTM_{Med}$, $LSTM_{Long}$ respectively and we set the number of hidden state values for each of the LSTMs to be 200.

(2) Survival analysis model. We predict the instantaneous attack probability λ_t at time t from the output of the dense layer after $LSTM_{Short}, LSTM_{Med}, LSTM_{Long}$. We use the probabilities $\lambda_1, \lambda_2, \ldots, \lambda_N$ (we use N=30) to calculate S_t, the probability of no attack by time t.

Xatu trains separate models for each attack type and evaluates them correspondingly. These models together cover 97.2% of all alerts in our dataset, and we can always train new models when new attack types become popular. The latter happens infrequently when a new vulnerability is detected [85]. During training, Xatu uses an Adam optimizer[40] with the SAFE loss function [89], a learning rate of 0.0001, and a batch size of 64. We perform an extensive sensitivity analysis on our ML parameters in Appendix H.

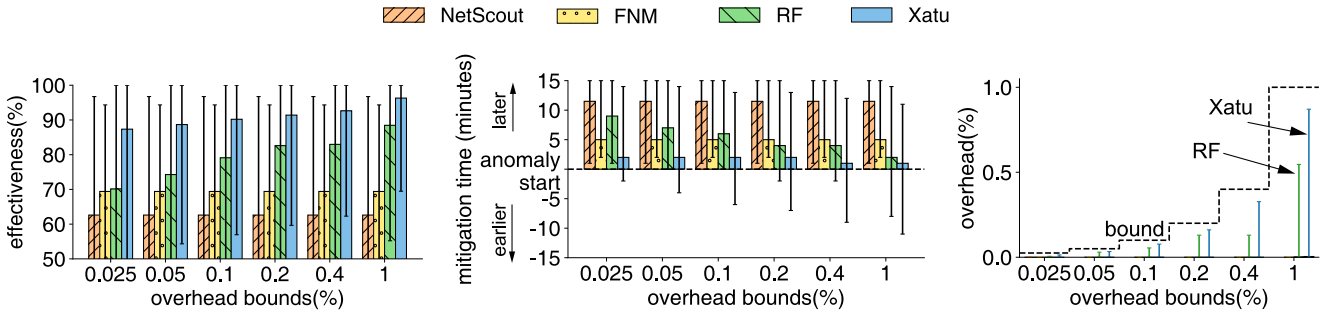
Xatu generates an attack alert when S_t falls below a threshold. After training, Xatu uses the validation phase to set appropriate thresholds. We identify the threshold in the validation data, which maximizes mitigation effectiveness, while keeping the scrubbing overhead for 75% of customers below a given bound. The scrubbing overhead bound is probabilistic, because while ground truth waits until an obvious anomaly, Xatu can detect the true start with auxiliary signals. This additional overhead is the right thing and shall not be limited by scrubbing overhead bound.

Finally, during operation (and during our evaluation), at each time step, Xatu generates and stores 140 MB of time-series feature data for the past 10 days for each customer. We use Xatu in an auto-regressive fashion [25], where the model takes into account its own previous early detection at each time step. Each detection boost model operates independently and generates attack alerts when the S_t is below the threshold selected during validation. Each detection runs within 10 ms. In deployment, Xatu can receive misdetection feedback from CDet and trigger retraining when many misdetections occur. Each retraining takes 4-8 hours but it is not necessary to retrain Xatu frequently, since the DDoS behavior evolves slowly [50].

6 EVALUATION

We evaluate Xatu using real-world NetFlow and attack detection alerts from a large regional network provider, as described in §5.2. We split the dataset chronologically into three parts (See Table 2 in Appendix F for details): (1) Training: 50 days from April 24th to June 12nd; (2) Validation: 20 days from June 13th to July 2nd; and (3) Testing: 30 days from July 3rd to August 1st, all in 2019.[4] During testing Xatu generates alerts that an attack is about to occur at a given time. This early detection is included as if it were ground truth to calculate A2, A4, and A5 groups of features for future time

[4]Experiments with different split ratios show the same good performance.



(a) Effectiveness (10^{th}, 50^{th}, and 90^{th} percentiles).

(b) Detection delay (10^{th}, 50^{th}, and 90^{th} percentiles).

(c) Scrubbing overhead (25^{th}, 50^{th}, and 75^{th} percentiles).

Figure 8: Effectiveness, detection delay, scrubbing overhead and ROC curve comparison of CDet, random forest (RF), and Xatu.

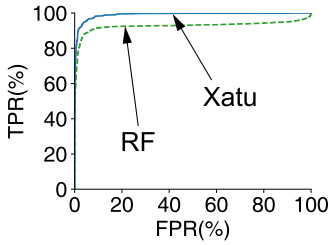
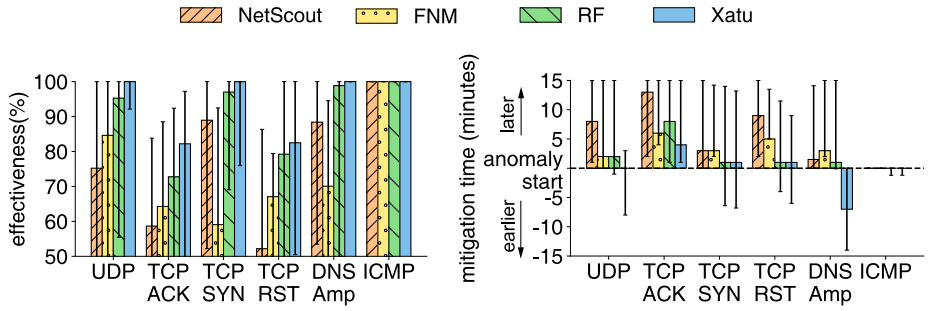


Figure 9: ROC curve.



(a) Effectiveness.

(b) Detection delay.

Figure 10: Xatu's performance across different attack types.

steps (§5.3). Due to this setup, we allocate the first 10 days of the testing period for our early detection accuracy to stabilize (similar to [25]). We report the results after the stabilization period.

Metrics. We adopt traditional detection metrics: false positive rate and false negative rate in our evaluation. Additionally, to capture the timeliness of detection and its impact on scrubbing, we use three metrics as defined in §2: **(1) Mitigation effectiveness.** The portion of anomaly traffic that is directed to CScrub from the start of the anomaly until the mitigation ends. Effectiveness measures the ability to remove anomalous traffic. **(2) Scrubbing overhead.** The ratio of traffic directed to CScrub prior to (or in the absence of) anomaly, divided by anomalous traffic. Overhead measures the extra cost of scrubbing legitimate traffic. **(3) Detection delay.** The time taken to detect an attack, after the start of anomalous traffic in the ground truth (a negative number means detection prior to the attack).

We show the median value with boxes, and we show high and low percentiles as error bars. For scrubbing overhead, recall that we measure cumulative overhead per customer, over many attacks. We show 25^{th} and 75^{th} percentiles for overhead, to show that our bound in training (75% of customers have low overhead in §5.3) is achieved by Xatu during evaluation. For mitigation effectiveness and detection delay, we show the 10^{th} and 90^{th} percentiles, to better capture head and tail distributions for those two measures. Ideally, our system should have high effectiveness, bounded overhead, and detection delay should be close to zero.

Alternative approaches. We compare Xatu with the following attack detection approaches: **(1) NetScout.** Arbor NetScout [62] is the CDet system used in our provider network, and has a 22% market share [61]. **(2) FNM.** FastNetMon (FNM for short) [64] is a threshold-based DDoS detection system using NetFlow that is open source and used both in business and in research [72, 74]. We configure FastNetMon with the best dynamic thresholds in production [84]. **(3) RF.** Random forest [28] is the standard supervised learning solution, which has been commonly used in DDoS detection [18, 30]. We trained RF as a binary classifier for each attack type using the same feature set from the same three timescales and applied an exhaustive grid search to identify the best hyper-parameters. **(4) Variants of Xatu.** We also investigate variants of Xatu with different auxiliary signals and time lengths in micro-benchmarks to verify the contribution of auxiliary signals and multi-LSTM.

6.1 Effectiveness and Overhead Metrics

Figure 8 shows various performance metrics as we vary scrubbing overhead bound, i.e., the operational cost of Xatu.

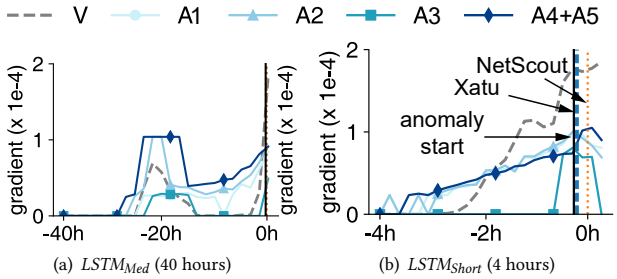
Xatu has higher mitigation effectiveness than alternative approaches because of early detection. Across different overhead bounds, Xatu's effectiveness is 39.6–53.8% higher than that of

NetScout and 25.9–38.8% higher than that of FNM (Figure 8(a)). This is because NetScout and FNM only rely on traditional volumetric signals and conservatively set such high thresholds [84] that they miss the ramp-up phases of the DDoS attacks. Meanwhile, Xatu detects anomalous traffic much earlier than NetScout or FNM with the help of auxiliary signals, which appear earlier than traditional volumetric signals. For example, in Figure 8(b), the median detection delay for Xatu is 1–2 minutes, while NetScout's or FNM's delay is 11.5 and 5 minutes. Xatu is always better in median effectiveness than RF by 8.7–24.7%. Although RF uses the same feature set and the same timescales, Xatu's LSTM captures the trends of auxiliary signals better. This is evident in Figure 8(b), where the median detection delay for Xatu is 1–2 minutes, compared to the RF's 2–9 minutes. RF even has a tail detection delay of 15 minutes among overhead bounds below 1%, which means that no detection occurs until the end of the time series.

Xatu bounds the scrubbing overhead while achieving high effectiveness. Early detection in Xatu may increase the scrubbing overhead due to false positives or too early detection. However, Xatu is able to bound the overhead for 75^{th} percentile of attacks, as expected (Figure 8(c)). Even when we bound overhead to 0.025%, which is very low, the median effectiveness increases by 39.6%. Customers deploying Xatu would gain a lot in effectiveness for this small additional cost. RF is also optimized for bounding the overhead. However, unlike Xatu, RF does not achieve large gains in effectiveness with the same overhead bound.

Xatu strikes a good balance between false positive rate and false negative rate. The ROC curves in Figure 9 show the trade-off between false positive rate and false negative rate. We view NetScout labels as ground truth and any Xatu's detection that does bir align with NetScout is counted as a false positive. For customers with at least one attack in our dataset, when the false positive rate is 4.8%, Xatu reaches a true positive rate as high as 95.4%, while RF only reaches 88.6. We manually inspect all false positive cases in Xatu, and 71% of false positive cases are likely to be missed attacks by NetScout with overwhelming suspicious traffic volume.

Xatu achieves high mitigation effectiveness across all attacks. Figure 10(a) shows the effectiveness of NetScout, FNM, RF, and Xatu across attack types under a scrubbing overhead bound of 0.1%. Xatu consistently achieves high effectiveness. For example, for UDP attacks, which are usually high-volume attacks, Xatu has 100% median effectiveness, while NetScout and FNM have only 75.2 and 84.6%. We also find Xatu to have high effectiveness for low volumetric attacks such as TCP ACK and TCP SYN. Xatu's median effectiveness is 82.2–100%, compared to both NetScout and FNM's 58.6–89%. This is also reflected in the median detection delay (Figure 10(b)), where Xatu achieves an average detection delay of 0, 4, and 1 minute for UDP, TCP ACK, and TCP SYN attacks respectively, compared to both NetScout and FNM's delays of 2–8, 6–13, and 3 minutes. For ICMP attacks, the median effectiveness of Xatu and both the CDets is 100%. ICMP attacks (constituting only 2% of all attacks) are usually short, and they tend to ramp up very quickly creating an easy-to-detect signal. RF is still inferior to Xatu, although it performs better than both CDets. RF has 1.5–14.9% lower median effectiveness, and up to 8 minutes higher median detection delay. For UDP and DNS Amplification attacks, RF alerts lag 7–14 minutes after Xatu for the 10^{th} percentile of time.



(a) $LSTM_{Med}$ (40 hours) (b) $LSTM_{Short}$ (4 hours)

Figure 11: Xatu's early detection uses aux. signals.

6.2 Why Does Xatu Work?

To illustrate why Xatu works, we take an example of a UDP attack and show the gradients observed in the LSTM model in Figure 11. The gradient of the input features represents the contribution of the features towards the final early detection—a higher gradient implies more contribution [66]. Particularly, we will focus on previous attack sources (A2 auxiliary signal), as they have the strongest signal for this attack.

The gradient for V (volumetric features) follows the UDP traffic towards the customer and CDet generates an alert when the UDP volume grows beyond the customer's threshold. This happens 14 minutes after the anomaly start time. Xatu, on the other hand, leverages auxiliary signals to detect attacks much earlier—one minute after anomaly start time.

Figure 11(a) shows that the A2 gradient in LSTM_{Med} is high 22 hours before the anomaly start time. This corresponds to the preparatory stage, where attackers generated 4.8 Kpps of traffic from 205 sources, which attacked the same customer 4 days prior. Figure 11(b) shows how $\text{LSTM}_{\text{Short}}$, detects A2 auxiliary signals 10 hours before the anomaly start time, even when there is no UDP traffic towards the customer (V gradient is zero). This illustrates how historical attack behavior contributes to Xatu's detection boost.

Even when A2 gradient is high, Xatu does not raise an alarm right away. Instead, it is able to generate the alarm close to the anomaly start time. This is because Xatu leverages the survival analysis model to balance mitigation effectiveness and scrubbing overhead.

6.3 Breakdown of Xatu's Contribution

Figure 12 shows the breakdown of Xatu signals' contribution to effectiveness under scrubbing overhead bound of 0.1% among different attack types. We train a separate model for each bar. Note that NetScout already has 52.2–100% median effectiveness, so the key challenge is how to improve the effectiveness for the lower half of the attacks.

Xatu's auxiliary signals Different auxiliary signals help detect and filter different subsets of attacks, and different attack types, and together they significantly improve the effectiveness of handling all attacks. For UDP attacks, each auxiliary signal in Xatu improves the 10^{th} effectiveness from 68.2% (no aux) to 71.2%—75.1% (w some

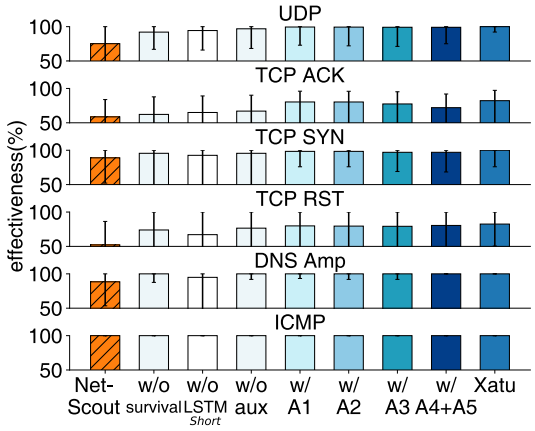
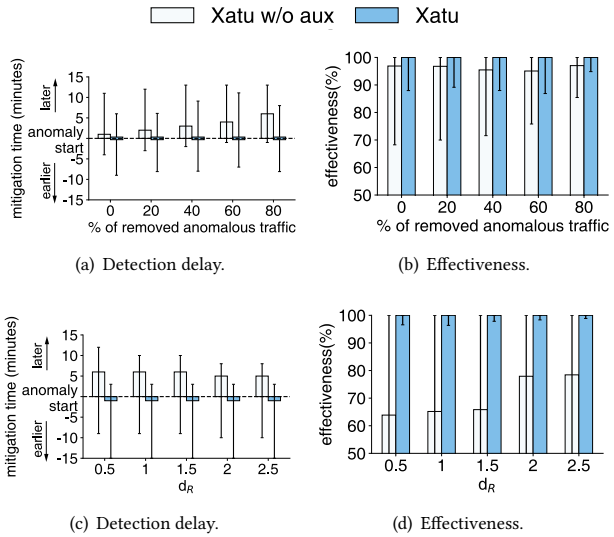


Figure 12: Contribution of aux. signals.



(a) Detection delay. (b) Effectiveness.

(c) Detection delay. (d) Effectiveness.

Figure 13: Xatu's robust against volume attackers (Fig a,b) and rate attackers (Fig c,d). (with 0.1% overhead bound)

of A1–A5) and median effectiveness from 96.9% to 99.1%—99.5%, indicating all auxiliary signals are effective attack indicators. The biggest increment in effectiveness for both UDP attacks and DNS amplification attacks comes from A4+A5 signals of attack history, due to their serial attack behavior. However, DNS amplification attacks benefit little from A1 or A3 signals, because the source of the reflection attack's traffic, DNS resolvers, is unlikely to appear in blocklists or be spoofed. Attacks related to TCP protocols—TCP ACK, TCP SYN, and TCP RST attacks—benefit from A1 and A2 signals most, because the same army of attackers seem to be continuously hired to conduct these attack types. The subset of attacks that benefit from a combination of these signals comprises only 27% of attacks, indicating that each signal contributes uniquely.

Xatu's ML design Xatu has two key ML designs: survival model and multi-timescale LSTM. Compared to Xatu w/o survival model, Xatu improves the median effectiveness by 4.6-32.2% among all attack types except ICMP attacks. Compared to simple LSTM, multi-timescale LSTM can help capture long-term historical trends such as periodic volume patterns. Compared to Xatu with only $LSTM_{Short}$, Xatu with multi-timescale LSTM improves the median effectiveness by 3.2-22.7% among all attack types except ICMP and DNS amplification attacks. The largest improvement from both survival model and multi-timescale LSTM are on attacks using TCP protocol, whose complicated time trends benefit from survival model and multi-timescale LSTM. More details are shown in Appendix H.

6.4 Robust to Smart Attackers

Attackers may change the way they ramp up the anomalous traffic to avoid attack detectors, which rely on traffic volume or ramp-up rates [21, 23, 76, 84]. Here we use UDP attacks[5] as an example and change the anomalous traffic volumes in the ramp-up period. We define the ramp-up time from the beginning of anomalous traffic to just before CDet detection time so that CDet performance is not affected. Xatu is robust to such volume-changing and rate-changing attackers.

[5]Other types of attacks show similar trends

Volume-changing attackers. We reduce the anomalous traffic sent by attackers just before CDet's detection time. When we remove these attackers, we also remove their corresponding auxiliary signals. Most DDoS detection systems rely on volumetric signals to detect attacks, thus volume-changing attackers can delay detection. Xatu's median and 90^{th} percentile effectiveness always remain at 100% (Figure 13(b)). On the other hand, as the attack volume changes, Xatu without auxiliary signals loses up to 6% of its effectiveness. Thus auxiliary signals help Xatu remain robust, as attackers adjust their rates.

Figure 13(a) shows the change in detection delay when attackers change their attack volume. The median detection delay of Xatu is always zero. When the auxiliary signals are removed, the median detection delay increases to 2–6 minutes. A similar effect can be noticed for 90^{th} percentile of detection delay. It is 6–11 minutes when Xatu leverages auxiliary signals and 11–13 minutes without auxiliary signals.

Rate-changing attackers Attackers could change the dynamics of their attack's onset to evade early detection, e.g., by ramping up slowly or suddenly. We evaluate the impact of these tactics on Xatu in Figure 13(d) and Figure 13(c). We change d_R the ramp-up rates of anomalous traffic (see the rate definition and visualization in Appendix G).

Figure 13(d) shows the change in effectiveness when the d_R increases. We find that Xatu and Xatu without auxiliary signals have consistent 90^{th} effectiveness of 100%, even when attackers are able to change their ramp-up rates. However, without using auxiliary signals, Xatu becomes more sensitive to rate change for the median effectiveness. The median effectiveness for Xatu without auxiliary signals for the lowest d_R is 63.9% and as d_R increases, the effectiveness improves to as much as 78.4%. This occurs because volumetric signals become more prominent with the increase in d_R. This is also reflected in median detection delay, which is 6–7

minutes earlier without auxiliary signals, and around zero with auxiliary signals, shown in Figure 13(c).

7 RELATED WORK

Various systems [20, 48, 70, 88, 90] have been developed for DDoS defense at the ISP to protect their customers. All these systems are reactive in nature and would benefit from working with Xatu to improve effectiveness. Once anomalous traffic is detected, the traffic is sent to the CScrub for further cleaning. Existing work employs different strategies of scrubbing [4, 26, 33, 55, 58]. Customers using these systems would benefit by using them along with Xatu, as Xatu reduces the cost of scrubbing, by bounding overhead.

Statistical analyses have been commonly used to detect anomalous traffic. One work [21] calculates entropy values and Chi-square values for features from packet headers to measure the deviation of the current traffic profile from the normal traffic for attack detection. Other works measure the ratio of inbound and outbound traffic volume [5], multivariate covariance of features in the TCP packet header [37], or time-based features such as periodicity and skewness of incoming traffic volume [23]. Previous works also adopt dynamic parameter configurations such as CUSUM [10] and EWMA [29]. An autoregressive fractional integrated moving average model (ARFIMA) in [2] is used as a combination of fractional differential noise and autoregressive moving average. However, these works only detect an attack when the attack has already started. Xatu is designed to *boost* detection at the start of anomalous traffic, which is before the attack, thereby increasing the effectiveness of defenses.

Attack detection approaches [1] based on machine learning collect a diverse set of traffic features and classify anomalous traffic automatically. IXP scrubber [83] tags each flow with association rule mining and then classifies per-target IP profiles aggregated from flows with machine learning classification. Kitsune [57] tracks features in network channels and feed them into an ensemble of auto-encoders for anomaly detection. Roudière et al. [72] produces an outlierness score with k nearest-neighbor (kNN) to determine traffic anomalies. Other works apply random partitioning binary trees [51], naive Bayes classifier [75], SVM [53], random forest[30], clustering [68], and neural networks [18] on various traffic features to classify anomalous and normal traffic. These are standalone ML solutions that detect an attack after anomalous traffic is established. Instead, Xatu is a boost to detection systems, which is easier to adopt.

Apart from DDoS attacks detection, machine learning has been applied to detect various attacks – random forests for BGP hijacking [78], natural language processing for phishing detection [67], LSTM classifier for Android Malware detection [86], or logistic regression to detect DNS tuneling [17], or adaptive deep forest for SQL injection attack [45]. Together with Xatu, machine learning provides opportunities for better attack detection.

8 DISCUSSION

Generality. Xatu builds a general model across all customers. If a customer is not attacked during training, Xatu can still protect him well, because Xatu transfers attack knowledge across customers. In our evaluation, 65.1% of customer nodes were not attacked during training. Xatu achieved a similar early detection of 9 minutes by median on those nodes.

Xatu relies on history traffic and auxiliary signal information to build up the model. Thus, similar to other statistical analyses and machine learning based detection solutions in §7, we need to retrain the model as we observe new attacks and low effectiveness. Xatu can train new models in 4–8 hours. New attack types occur rarely because they need the discovery of new vulnerabilities or reflection vectors [79].

Xatu may need retraining as traffic patterns and auxiliary signals change. Xatu does not require frequent retraining because its multi-timescale models capture well both short-term and long-term patterns, which evolve slowly. In §6, Xatu boosts detection of current attacks with the model trained from detection over two months preceding the attacks.

Limitations. Xatu is robust to some evasion attacks not preceded by auxiliary signals in §6.4. But ultimately, a determined attacker could also attempt to avoid Xatu's detection by minimizing auxiliary signals. To do so, the attacker would have to always use new attack sources and launch attacks at random times and of random types, on random customers. Such attacks are theoretically possible but they are unlikely. They would be very costly for the attacker, requiring new hosts and new attack software, and attacks on random customers at random times would serve no purpose.

Our dataset is collected at one large ISP (§2.2) which covers a variety of traffic patterns and attack patterns. We have also use Xatu to boost multiple detection solutions. Thus we believe Xatu can boost attack detection at many ISPs. Ideally, we would have liked to evaluate Xatu on more datasets, but large real traffic datasets from ISPs are not publicly available, due to provider's privacy concerns. A few publicly available, real DDoS datasets (e.g., Merit [31], CAIDA [9]), have only a handful of attacks, while machine learning requires large datasets. Also, none of the public DDoS datasets have accompanying detection signals from a commercial defense, which we need for labeling.

9 CONCLUSION

Xatu leverages auxiliary signals that occur before DDoS attacks to boost the effectiveness of existing CDet systems. Xatu uses a multi-timescale LSTM model to learn both long-term and short-term weak auxiliary signals, and survival analysis to achieve early detection with low scrubbing overhead. Experimental results show that these auxiliary signals bring huge gains in mitigation effectiveness (up to 44.1% over CDet) with low scrubbing overhead. This means that, with Xatu, customers would receive much faster DDoS mitigation, without paying much more for scrubbing.

ACKNOWLEDGMENTS

We thank our shepherd Oliver Hohlfeld and the anonymous reviewers for their insightful comments. We also thank Yang Zhou and Jiawei Zhou for their comments on various drafts of the paper. Zhiying Xu and Minlan Yu were supported in part by NSF grant CNS-1955422, CNS-1955487, and Tata Communications.

REFERENCES

[1] Ahamed Aljuhani. 2021. Machine Learning Approaches for Combating Distributed Denial of Service Attacks in Modern Networking Environments. *IEEE Access* (2021).

[2] Tomasz Andrysiak and Łukasz Saganowski. 2016. DDoS Attacks Detection by Means of Statistical Models. In *Proceedings of International Conference on Computer Recognition Systems CORES*. 797–806.

[3] Manos Antonakakis, Tim April, Michael Bailey, Matt Bernhard, Elie Bursztein, Jaime Cochran, Zakir Durumeric, J Alex Halderman, Luca Invernizzi, Michalis Kallitsis, et al. 2017. Understanding the Mirai Botnet. In *USENIX Security Symposium*. 1093–1110.

[4] Katerina J Argyraki and David R Cheriton. 2005. Active Internet Traffic Filtering: Real-Time Response to Denial-of-Service Attacks. In *USENIX ATC*, Vol. 38.

[5] Alexandru G Bardas, Loai Zomlot, Sathya Chandran Sundaramurthy, Xinming Ou, S Raj Rajagopalan, and Marc R Eisenbarth. 2012. Classification of UDP Traffic for DDoS Detection.. In *LEET*.

[6] Robert Beverly and Steven Bauer. 2005. The Spoofer Project: Inferring the Extent of Source Address Filtering on the Internet. In *Usenix Sruti*, Vol. 5. 53–59.

[7] Ryan Brunt, Prakhar Pandey, and Damon McCoy. 2017. Booted: An Analysis of a Payment Intervention on a DDoS-for-hire Service. In *Workshop on the Economics of Information Security*. 06–26.

[8] Xue Cai and John Heidemann. 2010. Understanding Block-level Address Usage in the Visible Internet. In *Proceedings of the ACM SIGCOMM*. 99–110.

[9] CAIDA. 2007. *The CAIDA UCSD "DDoS Attack 2007" Dataset.* https://www.caida.org/catalog/datasets/ddos-20070804_dataset

[10] Glenn Carl, George Kesidis, Richard R Brooks, and Suresh Rai. 2006. Denial-of-service Attack-detection Techniques. *IEEE Internet computing* 10, 1 (2006), 82–89.

[11] Wentao Chang, Aziz Mohaisen, An Wang, and Songqing Chen. 2015. Measuring Botnets in the Wild: Some New Trends. In *Proceedings of ACM Symposium on Information, Computer and Communications Security*. 645–650.

[12] Min Cheng, Qian Xu, Jianming Lv, Wenyin Liu, Qing Li, and Jianping Wang. 2016. MS-LSTM: A Multi-scale LSTM Model for BGP Anomaly Detection. In *IEEE ICNP*. 1–6.

[13] Michelle Cotton and Leo Vegoda. 2010. *Special Use IPv4 Addresses.* Technical Report.

[14] Jakub Czyz, Michael Kallitsis, Manaf Gharaibeh, Christos Papadopoulos, Michael Bailey, and Manish Karir. 2014. Taming the 800 Pound Gorilla: The Rise and Decline of NTP DDoS Attacks. In *Proceedings of IMC*.

[15] Alberto Dainotti, Karyn Benson, Alistair King, Bradley Huffaker, Eduard Glatz, Xenofontas Dimitropoulos, Philipp Richter, Alessandro Finamore, and Alex C Snoeren. 2016. Lost in Space: Improving Inference of IPv4 Address Space Utilization. *IEEE Journal on Selected Areas in Communications* 34, 6 (2016), 1862–1876.

[16] Michele De Donno, Nicola Dragoni, Alberto Giaretta, and Angelo Spognardi. 2018. DDoS-capable IoT Malwares: Comparative Analysis and Mirai Investigation. *Security and Communication Networks* 2018 (2018).

[17] Van Thuan Do, Paal Engelstad, Boning Feng, and Thanh van Do. 2017. Detection of DNS Tunneling in Mobile Networks Using Machine Learning. In *International Conference on Information Science and Applications*. 221–230.

[18] Rohan Doshi, Noah Apthorpe, and Nick Feamster. 2018. Machine learning DDoS Detection for Consumer Internet of Things Devices. In *IEEE Security and Privacy Workshops*. 29–35.

[19] Sam Edwards and Ioannis Profetis. 2016. Hajime: Analysis of a Decentralized Internet Worm for IoT Devices. *Rapidity Networks* 16 (2016).

[20] Seyed K Fayaz, Yoshiaki Tobioka, Vyas Sekar, and Michael Bailey. 2015. Bohatei: Flexible and Elastic DDoS Defense. In *USENIX Security Symposium*. 817–832.

[21] Laura Feinstein, Dan Schnackenberg, Ravindra Balupari, and Darrell Kindred. 2003. Statistical Approaches to DDoS Attack Detection and Response. In *Proceedings DARPA information survivability conference and exposition*, Vol. 1. 303–314.

[22] Massimo Ficco and Massimiliano Rak. 2014. Stealthy Denial of Service Strategy in Cloud Computing. *IEEE transactions on cloud computing* 3, 1 (2014), 80–94.

[23] Ramin Fadaei Fouladi, Cemil Eren Kayatas, and Emin Anarim. 2018. Statistical Measures: Promising Features for Time Series based DDoS Attack Detection. In *Multidisciplinary Digital Publishing Institute Proceedings*, Vol. 2. 96.

[24] Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. 1999. Learning to Forget: Continual Prediction with LSTM. (1999).

[25] Edit Gombay. 2008. Change Detection in Autoregressive Time Series. *Journal of Multivariate Analysis* 99, 3 (2008), 451–464.

[26] Google. 2022. *Project Shield.* https://projectshield.withgoogle.com/landing

[27] Olivia A Grigg, VT Farewell, and DJ Spiegelhalter. 2003. Use of risk-adjusted CUSUM and RSPRT charts for monitoring in medical contexts. *Statistical Methods in Medical Research* 12, 2 (2003), 147–170.

[28] Tin Kam Ho. 1995. Random Decision Forests. In *Proceedings of international conference on document analysis and recognition*, Vol. 1. 278–282.

[29] Rick Hofstede, Václav Bartoš, Anna Sperotto, and Aiko Pras. 2013. Towards Real-time Intrusion Detection for NetFlow and IPFIX. In *Proceedings of International Conference on Network and Service Management*. 227–234.

[30] Mohamed Idhammad, Karim Afdel, and Mustapha Belouch. 2018. Detection System of HTTP DDoS Attacks in a Cloud Environment based on Information Theoretic Entropy and Random Forest. *Security and Communication Networks* 2018 (2018).

[31] ImpactCyberTrust.org. 2016. *Merit Network "RADB DDoS 2016" Dataset.* https://www.impactcybertrust.org/dataset_view?idDataset=576

[32] Imperva. 2022. *How To Choose The Right Mitigation Service.* https://www.imperva.com/learn/ddos/ddos-mitigation-services

[33] Amazon Web Services Inc. 2022. *AWS Shield.* https://aws.amazon.com/shield

[34] Cloudflare Inc. 2022. *Cloudflare DDoS Protection.* https://www.cloudflare.com/ddos

[35] ISC. 2022. *DShield Reports and Database Summaries.* https://isc.sans.edu/reports.html

[36] ISC. 2022. *General Information On Submitting Logs To DShield.* https://isc.sans.edu/howto.html

[37] Shuyuan Jin and Daniel S Yeung. 2004. A Covariance Analysis Model for DDoS Attack Detection. In *IEEE International Conference on Communications*, Vol. 4. 1882–1886.

[38] Mattijs Jonker, Alistair King, Johannes Krupp, Christian Rossow, Anna Sperotto, and Alberto Dainotti. 2017. Millions of Targets under Attack: a Macroscopic Characterization of the DoS Ecosystem. In *Proceedings of Internet Measurement Conference*. 100–113.

[39] Mattijs Jonker, Anna Sperotto, Roland van Rijswijk-Deij, Ramin Sadre, and Aiko Pras. 2016. Measuring the Adoption of DDoS Protection Services. In *Proceedings of Internet Measurement Conference*. 279–285.

[40] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980* (2014).

[41] John P Klein and Melvin L Moeschberger. 2006. *Survival Analysis: Techniques for Censored and Truncated Data.*

[42] Constantinos Kolias, Georgios Kambourakis, Angelos Stavrou, and Jeffrey Voas. 2017. DDoS in the IoT: Mirai and Other Botnets. *Computer* 50, 7 (2017), 80–84.

[43] Matthieu Latapy, Clémence Magnien, and Nathalie Del Vecchio. 2008. Basic Notions for the Analysis of Large Two-mode Networks. *Social networks* 30, 1 (2008), 31–48.

[44] Steve Lawrence, C Lee Giles, Ah Chung Tsoi, and Andrew D Back. 1997. Face Recognition: A Convolutional Neural-network Approach. *IEEE transactions on neural networks* 8, 1 (1997), 98–113.

[45] Qi Li, Weishi Li, Junfeng Wang, and Mingyu Cheng. 2019. A SQL Injection Detection Method Based on Adaptive Deep Forest. *IEEE Access* 7 (2019), 145385–145394.

[46] Andy Liaw, Matthew Wiener, et al. 2002. Classification and Regression by Random Forest. *R news* 2, 3 (2002), 18–22.

[47] Franziska Lichtblau, Florian Streibelt, Thorben Krüger, Philipp Richter, and Anja Feldmann. 2017. Detection, Classification, and Analysis of Inter-domain Traffic with Spoofed Source IP Addresses. In *Proceedings of Internet Measurement Conference*. 86–99.

[48] Zaoxing Liu, Hun Namkung, Georgios Nikolaidis, Jeongkeun Lee, Changhoon Kim, Xin Jin, Vladimir Braverman, Minlan Yu, and Vyas Sekar. 2021. Jaqen: A High-Performance Switch-Native Approach for Detecting and Mitigating Volumetric DDoS Attacks with Programmable Switches. In *30th USENIX Security Symposium (USENIX Security 21)*. 3829–3846.

[49] Matthew Luckie, Robert Beverly, Ryan Koga, Ken Keys, Joshua A Kroll, and K Claffy. 2019. Network Hygiene, Incentives, and Regulation: Deployment of Source Address Validation in the Internet. In *Proceedings of ACM SIGSAC Conference on Computer and Communications Security*. 465–480.

[50] Steve Mansfield-Devine. 2014. The evolution of DDoS. *Computer Fraud & Security* 2014, 10 (2014), 15–20.

[51] Pierre-François Marteau. 2021. Random Partitioning Forest for Point-Wise and Collective Anomaly Detection—Application to Network Intrusion Detection. *IEEE Transactions on Information Forensics and Security* 16 (2021), 2157–2172.

[52] Mary Meeker and Liang Wu. 2013. Internet Trends. In *Proc D11 Conference*.

[53] Nisharani Meti, DG Narayan, and VP Baligar. 2017. Detection of Distributed Denial of Service Attacks Using Machine Learning Algorithms in Software-defined Networks. In *International conference on advances in computing, communications and informatics*. 1366–1371.

[54] Rui Miao, Rahul Potharaju, Minlan Yu, and Navendu Jain. 2015. The Dark Menace: Characterizing Network-based Attacks in the Cloud. In *Proceedings of IMC*.

[55] Microsoft. 2022. *Azure DDoS Protection Overview.* https://docs.microsoft.com/en-us/azure/ddos-protection/ddos-protection-overview

[56] Microsoft. 2022. *DDoS Protection and Mitigation Services.* https://azure.microsoft.com/en-us/services/ddos-protection

[57] Yisroel Mirsky, Tomer Doitshman, Yuval Elovici, and Asaf Shabtai. 2018. Kitsune: an Ensemble of Autoencoders for Online Network Intrusion Detection. *arXiv preprint arXiv:1802.09089* (2018).

[58] Giovane CM Moura, Cristian Hesselman, Gerald Schaapman, Nick Boerman, and Octavia de Weerdt. 2020. Into the DDoS Maelstrom: a Longitudinal Study of a Scrubbing Service. In *IEEE European Symposium on Security and Privacy Workshops*. 550–558.

[59] Giovane CM Moura, Ricardo de O Schmidt, John Heidemann, Wouter B de Vries, Moritz Muller, Lan Wei, and Cristian Hesselman. 2016. Anycast vs. DDoS: Evaluating the November 2015 root DNS event. In *Proceedings of IMC*.
[60] RIPE NCC. 2022. *RIPE Routing Information Service (RIS).* https://www.ripe.net/analyse/internet-measurements/routing-information-service-ris
[61] NETSCOUT. 2020. *NETSCOUT Maintains #1 Position in the DDoS Prevention Appliance Market.* https://www.netscout.com/report/omdia-ddos-prevention-appliances-excerpts
[62] NETSCOUT. 2022. *Arbor DDoS Attack Protection Solutions.* https://www.netscout.com/arbor-ddos
[63] Sean Newman. 2019. Under the Radar: the Danger of Stealthy DDoS Attacks. *Network Security* 2019, 2 (2019), 18–19.
[64] Pavel Odintsov. 2022. *FastNetMon.* https://github.com/pavel-odintsov/fastnetmon
[65] University of Oregon. 2022. *Route Views Project.* http://bgplay.routeviews.org
[66] Dhruva Patil, Bruce A Draper, and J Ross Beveridge. 2019. Looking under the Hood: Visualizing What LSTMs Learn. In *International Conference on Image Analysis and Recognition*. 67–80.
[67] Tianrui Peng, Ian Harris, and Yuki Sawa. 2018. Detecting Phishing Attacks Using Natural Language Processing and Machine Learning. In *IEEE international conference on semantic computing*. 300–301.
[68] Xi Qin, Tongge Xu, and Chao Wang. 2015. DDoS Attack Detection Using Flow Entropy and Clustering Technique. In *International Conference on Computational Intelligence and Security*. 412–415.
[69] Sivaramakrishnan Ramanathan, Jelena Mirkovic, and Minlan Yu. 2020. BLAG: Improving the Accuracy of Blacklists. In *Proceedings of NDSS*.
[70] Sivaramakrishnan Ramanathan, Jelena Mirkovic, Minlan Yu, and Ying Zhang. 2018. SENSS against volumetric DDoS Attacks. In *Proceedings of Annual Computer Security Applications Conference*. 266–277.
[71] Yakov Rekhter, Bob Moskowitz, Daniel Karrenberg, Geert Jan de Groot, and Eliot Lear. 1996. *Address Allocation for Private Internets.* Technical Report.
[72] Gilles Roudière and Philippe Owezarski. 2018. Evaluating the Impact of Traffic Sampling on AATAC's DDoS Detection. In *Proceedings of the Workshop on Traffic Measurements for Cybersecurity*. 27–32.
[73] José Jair Santanna, Roland van Rijswijk-Deij, Rick Hofstede, Anna Sperotto, Mark Wierbosch, Lisandro Zambenedetti Granville, and Aiko Pras. 2015. Booters—Analysis of DDoS-as-a-service Attacks. In *IEEE International Symposium on Integrated Network Management*. 243–251.
[74] Lumin Shi, Samuel Mergendahl, Devkishen Sisodia, and Jun Li. 2020. Bridging Missing Gaps in Evaluating DDoS Research. In *USENIX Workshop on Cyber Security Experimentation and Test*.
[75] Ningombam Anandshree Singh, Khundrakpam Johnson Singh, and Tanmay De. 2016. Distributed Denial of Service Attack Detection Using Naive Bayes Classifier Through info Gain Feature Selection. In *Proceedings of the International Conference on Informatics and Analytics*. 1–9.
[76] Robert Smith and Shawn Marck. 2019. *Identifying a Potential DDOS Attack Using Statistical Analysis.* https://patents.google.com/patent/US20160127406A1/en
[77] Akamai Technologies. 2022. *Large, Complex DDoS Attacks on the Rise in 2020 - The Akamai Blog.* https://blogs.akamai.com/2020/07/large-complex-ddos-attacks-on-the-rise-in-2020.html
[78] Cecilia Testart, Philipp Richter, Alistair King, Alberto Dainotti, and David Clark. 2019. Profiling BGP Serial Hijackers: Capturing Persistent Misbehavior in the Global Routing Table. In *Proceedings of the Internet Measurement Conference*. 420–434.
[79] Liam Tung. 2022. *DDoS Attackers Have Found This New Trick to Knock over Websites.* https://www.zdnet.com/article/attackers-now-hit-firewalls-to-knock-out-websites/
[80] Verizon. 2020. *DDoS Security - Service Level Agreement.* https://enterprise.verizon.com/service_guide/reg/cp_ddos_security_sla.pdf
[81] Daniel Wagner, Daniel Kopp, Matthias Wichtlhuber, Christoph Dietzel, Oliver Hohlfeld, Georgios Smaragdakis, and Anja Feldmann. 2021. United We Stand: Collaborative Detection and Mitigation of Amplification DDoS Attacks at Scale. In *Proceedings of Computer and Communications Security*. 970–987.
[82] Jason Weil, Victor Kuarsingh, Chris Donley, Christopher Liljenstolpe, and Marla Azinger. 2012. *IANA-reserved IPv4 Prefix for Shared Address Space.* Technical Report.
[83] Matthias Wichtlhuber, Eric Strehle, Daniel Kopp, Lars Prepens, Stefan Stegmueller, Alina Rubina, Christoph Dietzel, and Oliver Hohlfeld. 2022. IXP Scrubber: Learning from Blackholing Traffic for ML-Driven DDoS Detection at Scale. In *Proceedings of SIGCOMM*.
[84] James Edward Winquist, Joseph Welch, Tim Hoffman, and Olan Patrick Barnes. 2016. *Forced Alert Thresholds for Profiled Detection.* https://patents.google.com/patent/US9344440
[85] Wired. 2018. *A 1.3-Tbs DDoS Hit GitHub, the Largest Yet Recorded.* https://www.wired.com/story/github-ddos-memcached/
[86] Xi Xiao, Shaofeng Zhang, Francesco Mercaldo, Guangwu Hu, and Arun Kumar Sangaiah. 2019. Android Malware Detection based on System Call Sequences and LSTM. *Multimedia Tools and Applications* 78, 4 (2019), 3979–3999.
[87] Guang Yao, Jun Bi, and Peiyao Xiao. 2013. VASE: Filtering IP Spoofing Traffic with Agility. *Computer Networks* 57, 1 (2013), 243–257.
[88] Menghao Zhang, Guanyu Li, Shicheng Wang, Chang Liu, Ang Chen, Hongxin Hu, Guofei Gu, Qianqian Li, Mingwei Xu, and Jianping Wu. 2020. Poseidon: Mitigating Volumetric DDoS Attacks with Programmable Switches. In *Proceedings of NDSS*.
[89] Panpan Zheng, Shuhan Yuan, and Xintao Wu. 2019. Safe: A Neural Survival Analysis Model for Fraud Early Detection. In *Proceedings of the AAAI Conference*, Vol. 33. 1278–1285.
[90] Shengbao Zheng and Xiaowei Yang. 2019. Dynashield: Reducing the Cost of DDoS Defense Using Cloud Services. In *USENIX HotCloud 19*.

A CUSUM

The CUSUM [27] is a sequential analysis technique that involves the calculation of a cumulative sum and is used for anomaly detection. Given observations for a process $\{x_i\}_{i=1}^{N}$ with a mean of μ and a standard deviation of σ, the normalized observations $\{Z_i\}_{i=1}^{N}$ can be derived from $Z_i = \frac{1}{\sigma}(x_i - \mu - \text{NUMSTD} \cdot \sigma)$, where NUMSTD denotes how many standard deviations is the given value over what is expected for normal distribution. The CUSUM values S_n is calculated from the normalized observation with $S_n = \max(0, S_{n-1} + Z_n)$. When the value of S_n exceeds a certain threshold value, an anomaly in value has been found.

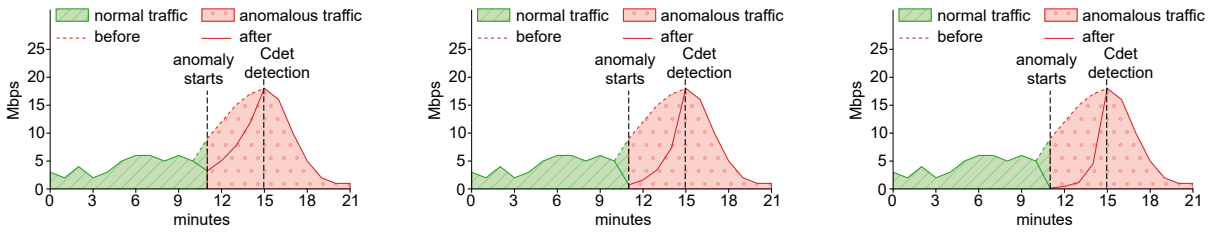
Instead, we use CUSUM in combination with attack labels to mark attack start[6]. The mean μ and a standard deviation σ are estimated by the bytes within the hour before the attack. Given a known attack, we select a more aggressive parameter in CUSUM to detect minor anomalies so that CUSUM could provide enough ground truth of anomalous traffic before attacks. In specific, we select NUMSTD = 1 for UDP, DNS Amplification attacks and NUMSTD = 0.5 for TCP ACK, TCP SYN, TCP RST, and ICMP attacks. However, the aggressive parameter is likely to cause high false positive cases if applied to normal time. Thus, CUSUM alone cannot be used as an early detection solution.

B AUXILIARY SIGNAL OBSERVATION

Activity of potential attack sources increases closer to the attacks. Auxiliary signals can bring different early detection benefits for different attacks. Figure 15 shows the 25^{th}, 50^{th}, and 75^{th} percentile of blocklisted sources, previous attackers, and spoofed sources that sent any traffic to customers in our dataset, in the ten days preceding the attack. As we approach the actual attack, the percentage of potential attack sources increases. For instance, from five days prior to one day prior to the attack, the median percentage of blocklisted sources increases from 66.7% to 92.9%, previous attackers increases from 64% to 86.7%, and spoofed sources increase from 2% to 92.6%. During the same period, the volume of traffic from these potential attack sources (not shown in the figure) increases to 16.2%, 27.1%, and 11.2% for blocklisted sources, previous attackers, and spoofed sources.

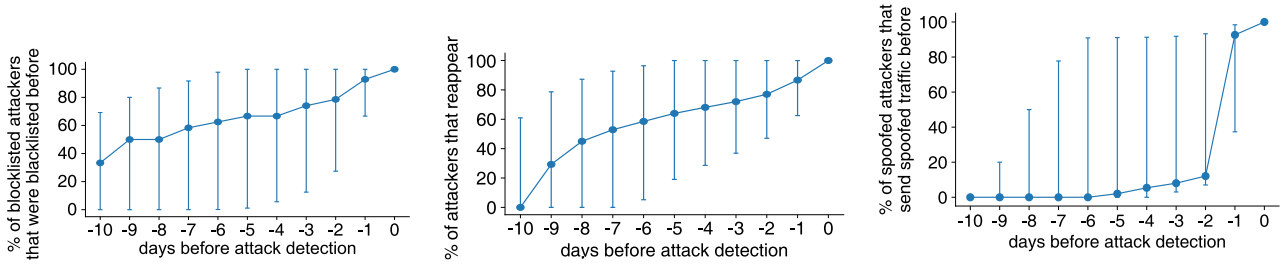
Correlated attack occurrences across customers increase closer to the attacks. Figure 16 illustrates these correlated attack occurrences, by showing the change in clustering coefficient [43], for a group of customers that has been attacked by the same set of attackers. The clustering coefficient ranges from 0 for customers being attacked by non-overlapping attacker groups, to 1, for customers

[6]This does not mean that if we run CUSUM we would have 100% effectiveness. This analysis was done in retrospect—that is CUSUM was applied on traffic before an attack was detected by a CDet.



(a) $d_R = 0.5$. (b) $d_R = 1.5$. (c) $d_R = 2.5$.

Figure 14: As the d_R value increases, the anomalous ramps up towards a customer quicker.



(a) Blocklists. (b) Previous attacks. (c) Spoof.

Figure 15: Increasing reappearance of attacker IP addresses over time (The error bar shows 25th, 50th, and 75th percentile of attacks based on the % attackers on the day -10.)

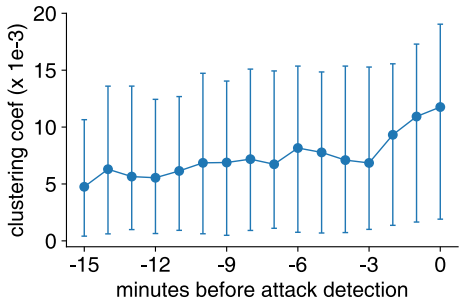


Figure 16: Clustering coefficient of attackers and customers.

being attacked by identical attackers. We only show customers that have some overlapping attacker groups (clustering coefficient larger than 0). The median clustering coefficient in our example shows an increase from 4.8×10^{-3} at 15 minutes before attack detection to 11.8×10^{-3} at the time of CDet's detection.

C LOSS FUNCTION

We adopt the loss function [89] based on survival analysis. The detailed mathematical formulation is as follows.

Let $\{(\boldsymbol{x}^i, c^i, t^i)\}_{i=1}^N$ denotes a set of time series, where $\boldsymbol{x}^i = (\boldsymbol{x}_1^t, \boldsymbol{x}_2^t, \ldots, x_n^t)$ indicates the ith time series of feature vectors; c^i indicates whether the ith of time series has an attack ($c^i = 1$) or not ($c^i = 0$); t^i denotes the time whether the ith time series is detected as an attack or the end of time series for a normal time series. N denotes the size of the dataset. In our scenario, survival time T is the length of time before the attack happens, so the survival probability $S_t = P\{T \geq t\}$ indicates the probability of no attack by time t. The goal of learning is to train a mapping function between a time series $(\boldsymbol{x}^i, c^i, t^i)$ and the probability of a target customer having no attack by time t, i.e. survival probability S_t.

Given dataset $\{(\boldsymbol{x}^i, c^i, t^i)\}_{i=1}^N$, a simple but non-trivial maximum likelihood model [89] to estimate survival probability S_t will be (1) If a time series i has the attack label ($c^i = 1$), the likelihood function seeks to make the detection earlier than when the anomaly starts, i.e., maximizing $P\{T < t^i\}$; (2) if a time series i does not have the attack label ($c^i = 0$), the likelihood function aims to make no detection till the end of the time series, i.e., maximizing $P\{T \geq t^i\}$. Thus, the joint likelihood function for a sample i is $P\{T < t^i\}^{c^i} \cdot P\{T \geq t^i\}^{1-c^i}$. For the convenience of mathematical formulation, we introduce the hazard rate $\lambda_t = P\{T = t | T \geq t\}$ that refers to the instantaneous rate of an attack at time t given that no attack observed before time t. In discrete observation time where $P\{T = t\} = S_{t-1} - S_t$, then $S_t = e^{-\sum_{k=1}^{t} \lambda_t}$. Therefore, the loss function $\mathcal{L}$ in the form of negative log-likelihood over the whole

data is

$$\begin{aligned}\mathcal{L} &= \sum_{i=1}^{N}[-\ln(P\{T < t^i\}^{c^i} \cdot P\{T \geq t^i\}^{1-c^i})] \\ &= \sum_{i=1}^{N}[-\ln((1-S_{t^i})^{c^i} \cdot {S_{t^i}}^{1-c^i})] \\ &= \sum_{i=1}^{N}[(\sum_{t=1}^{t^i}\lambda_t) - c^i \ln(e^{-\sum_{k=1}^{t^i}} - 1)].\end{aligned}$$

In order to minimize the loss function, the LSTM is adopted for understanding the time series of features $\boldsymbol{x}^i, i = 1, 2, \ldots, N$ to estimate the hazard rate λ_t and survival probability S_t.

D FEATURE SELECTION

We collect traffic features related to ports and countries of traffic source as below. These ports and countries in our NetFlow are prevalent in our NetFlow and take up over 95% of traffic respectively.

- Ports: 0, 53, 80, 123, 443
- Countries of traffic source: US, IN, SA, CN, GB, NL, FR, DE, BR, CA

E BREAKDOWN OF A1'S CONTRIBUTION

As our A1 auxiliary signals (blocklisted sources) includes 11 categories of blocklists, Figure 17 shows the breakdown of their contribution to effectiveness improvement for different attack types. The three most prevalent categories—DDoS source blocklist, bot blocklist, and scanner blocklist—improve the 10^{th} by 9%–237% the 50^{th} by 1%–9.4% among UDP, TCP ACK, TCP SYN, TCP RST attacks. The rest of the blocklists together also achieve a similar effectiveness improvement among these attack types. DNS Amplification and ICMP attacks benefit less from different blocklists with no improvement on the 10^{th} and the 50^{th} effectiveness. The former is because the source of attack traffic, DNS resolvers' IP addresses, are unlikely to be in blocklists; The latter is because the easy-to-detect signals, which achieve 100% effectiveness with only volumetric signals, allow no further room for effectiveness improvement with blocklists.

F DATA SPLIT

Table 2 in Appendix F shows the number of attacks in the training, validation, and testing datasets for each attack type.

	%	Train.	Val.	Test.
UDP	26.3%	1.3K	530	944
TCP ACK	62.0%	3.6K	1.1K	1.9k
TCP SYN	1.4%	81	24	44
TCP RST	1.1%	42	12	64
DNS Amp	7.2%	382	118	270
ICMP	2.0%	81	49	82
Total	100%	5.6K	1.8K	3.3K

Table 2: # of various types of attacks.

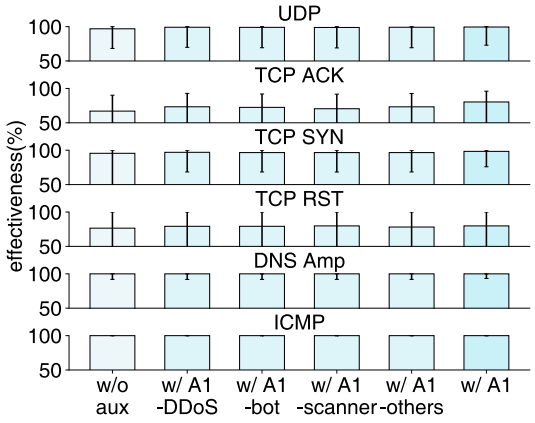


Figure 17: Contribution of A1 aux. signals.

G VISUALIZING d_R

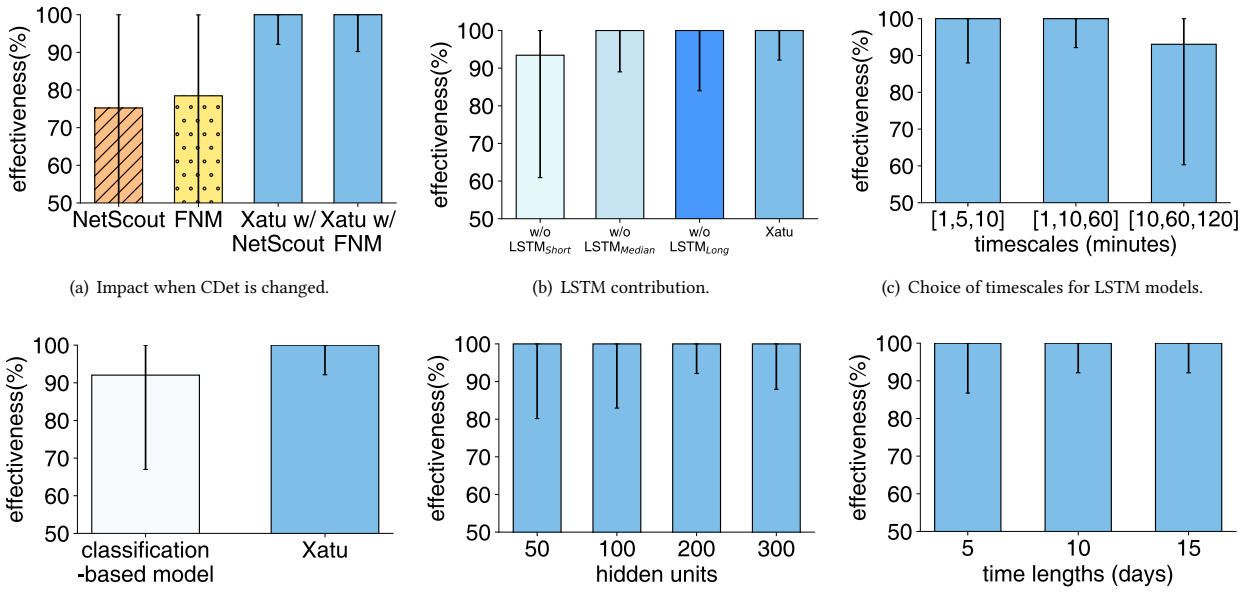
We vary the rate by varying $d_R = max\left|\frac{\mathrm{d}v_{out}}{\mathrm{d}t}\right|$—the speed of rate increase from one time period (1 minute) to the next [52]. For example, a $d_R = 1$ would mean that the attack rate doubles each minute, while a $d_R = 0.5$ means it increases by 50% each minute. We visualize how anomalous traffic is changed with three different d_R values in Figure 14.

H SENSITIVITY ANALYSIS

We conduct sensitivity analysis on different parameters used by Xatu, focusing on UDP attacks. Other attacks were tested but show similar trends.

Xatu is independent of CDet. We show that Xatu is independent of the underlying defense mechanism, by considering an open-source detection system FastNetMon [64] (denoted by FNM). We see in Figure 18(a) that the effectiveness of both NetScout and FNM are about the same (75.2%–78.5% median effectiveness). Xatu trained using labels generated by NetScout and FNM has comparable effectiveness (and detection delay, not shown). In both cases, Xatu offers significant improvements over the detection system alone. This shows that Xatu is only dependent on the attack detection system during the training/validation phase.

Xatu derives benefits from both short-term and long-term auxiliary signals. We perform analysis on the contributions of the three LSTM models, where we train Xatu without one of the LSTM models ($LSTM_{Short}$, $LSTM_{Med}$ and $LSTM_{Long}$), to understand their contribution. Figure 18(b) shows that $LSTM_{Short}$ has the highest contribution, as it improves the median effectiveness by 7.1%. This is also reflected in the detection delay (not shown in Figure), where the median delay decreases by 2 minutes. Further, short-term signals really help the 10^{th} percentile effectiveness, where the effectiveness improves by 51.2%. This is expected as we have seen in §3 that recent traffic behavior has a stronger signal for early attack detection. On the other hand, $LSTM_{Med}$ and $LSTM_{Long}$ contribute less, but still improve early detection of the tail-end of attacks. They improve the 10^{th} percentile of effectiveness by 3.5% and 9.6%.



(a) Impact when CDet is changed.

(b) LSTM contribution.

(c) Choice of timescales for LSTM models.

(d) Contribution of survival model.

(e) Different hidden units in LSTM.

(f) Choice of time length for LSTM models.

Figure 18: Sensitivity analysis on various components of Xatu (UDP attacks).

Aggregating at different timescales provides a good approximation. Earlier in §4, we described the need for three different timescales, to improve performance. To see the impact of different timescales,Figure 18(c) shows effectiveness for two additional configurations of timescales—one where (TS_{Short}, TS_{Med}, TS_{Long}) are tuned to the smaller timescale of interval (1, 5, 10) and the other with a larger timescale of interval (10, 60, 120). We find that having a smaller timescale has a lower 10^{th} percentile effectiveness (88%) when compared to the timescales chosen for Xatu (92.1%). On the other hand, we see a bigger drop in 10^{th} percentile effectiveness by using larger timescales for all attacks (60.3%), which is a 31.5% drop in effectiveness when compared with the timescales chosen for evaluating Xatu. This is expected, as the larger timescale aggregates the features more coarsely, and is less able to detect feature variations. We also conduct a grid search for different combination of TS_{Med} and TS_{Long}, and find our current selection achieves top effectiveness.

Survival analysis model assists Xatu to have high effectiveness. Xatu uses the survival analysis model as a loss function to balance effectiveness with overhead. To understand its contribution, we compare Xatu when it is trained using a binary cross-entropy loss function (classification-based approach). We see that Xatu with the survival model outperforms the classification model. The survival has higher median effectiveness by 8.7% , and the 10^{th} percentile effectiveness improves by 37.5%. This is also reflected in the 90^{th} percentile detection delay (not shown in the figure), where the survival model helps detect these attacks 3 minutes earlier than the classification model.

Hidden units. To estimate the optimal number of hidden units for the LSTM models, we start off with 50 hidden units, then 100 hidden units, followed by increments of 100 hidden units till we reach 300 hidden units, as shown in Figure 18(e). For all these configurations, the median and 90^{th} percentile effectiveness are high (nearly 100%). However, Xatu with a low number of hidden units affects the 10^{th} percentile effectiveness—where the effectiveness is just 80.2% for 50 hidden units. As we increase the number of hidden units, the 10^{th} percentile effectiveness improves. We find the best effectiveness of 88% with 200 hidden units, and we use this configuration in the evaluation. We also observe that by increasing the number of hidden units further, the 10^{th} percentile effectiveness does not change, thus 200 is sufficient. We further conduct a grid search for the number of hidden units from 50 to 1000 and find that any number between 150 and 700 will well satisfy our purpose.

Time length. As Figure 15 and Figure 16 show potential attack sources are active up to 10 days before the attacks, we further validate that observation on the choice of time length in LSTM models. Figure 18(f) depicts the effectiveness under different time lengths including 5 days, 10 days, and 15 days before the attacks. We see a 6% drop in the 10^{th} percentile effectiveness in a smaller time lengths. This is expected as the 5-day time lengths cannot well capture the complicated time trends of tail cases. On the other hand, we find little effectiveness improvement in 15-day time length. Thus, collecting a dataset with a longer time length is not necessary.