# A SURVEY OF TRAFFIC SHAPING APPLICATION IN PRIVACY PROTECTION

TECHNICAL REPORT

**Lili Tang**[*]
Department of Information Security
University of Science and Technology of China
Hefei
tl2000@mail.ustc.edu.cn

**Zhengyuan Jiang**[*]
Department of Information Security
University of Science and Technology of China
Hefei
jzy2018@mail.ustc.edu.cn

July 9, 2021

## ABSTRACT

Due to the explosive growth information uploaded in the Internet, it is particularly important to protect personal information from leakage. However, even if you are using VPN and strong encryption while surfing the Internet, some potential passive network observer may also obtain your private information through traffic analysis without knowing any encrypted content in the traffic packet. Worse, for example, the privacy of IoT net is more vulnerable with the singleness of equipment function and its sparse(usually) and fixed traffic pattern. Thus, passive network observers can simply apply some machine learning algorithms to cluster all the devices and different traffic model and then train a Hidden Markov Model (HMM) to infer the user activities. In this paper, our group introduces two fundamental algorithms: Leaky Bucket and Token Bucket. We implement a simple demo for each of them. Moreover, we demonstrate the vulnerability of privacy through traffic analysis in IoT net, and enumerate some specific attacks to that. We summarize the traditional methods against traffic analysis and two specific traffic shaping algorithms: independent link padding (ILP) and stochastic traffic padding (STP) applied in IOT devices. We further point out the pros and cons and their specific application situation. Finally, we evaluate its feasibility in real network scenario and propose our methods to optimize its implementation and deployment. Our new algorithm is random burst padding (RBP) that can deal with burst traffic and simulate real traffic which might be more effective in real IoT net. Also we suggest that there may be a good match between convert channel and traffic shaping to defense both external and internal adversaries.

***Keywords*** Cyber Security · IoT · Traffic Shaping · Traffic Analysis attack

## 1 Introduction

On one hand, traffic shaping is a well-known technique to guarantee or improve end-to-end delays and avoid buffer overflow, thereby increasing the Quality of Service in networks. This is achieved by a traffic profile defined in network nodes. Traffic shapers are components which receive a stream of packets and potentially delay them, such that their outgoing stream contains fewer bursts and a guaranteed number of packets per time interval. This can decrease network contention and limit the buffer size in network nodes required to temporarily store incoming data.This kind of traffic shaping is often used in real-time system to provide a low-delay and stable traffic flow service[6]. IEEE P802.1 Qcr ATS was proposed as a method to smooth traffic patterns by reshaping streams per-hop and is independent of network-wide global time synchronization. This traffic shaping strategy is usually implemented in the link layer since in the link layer, a widely used protocol is Ethernet which does not provide sufficient QoS mechanisms to service packets in real-time applications. In some industries such as automotive, this real-time service in the link layer is pivotal.

[*] Information about author (webpage,research institute, alternative address, e-mail)

On the other hand, traffic shaping is also applied to provide privacy protection and anti-traffic analysis services. In traditional Internet, our packets can be protected by encryption, firewall and virtual private network(VPN). However, all the potential adversary gets is traffic information such as traffic peaks and flow timing diagrams (when the user surfs the Internet). Low level of privacy leakage is acceptable due to the diversity of activities in our PC, pad or phone.However, with the rapid development of the Internet of Things network, such leakage of traffic information may lead to severely serious privacy issues, which are challenges must to be faced when widely implementing the IoT network. Almost all IoT devices have only one single function, and the functions of a certain IoT manufacturer's IoT devices are fixed, which provides an effective method (MAC address) for identifying IoT devices. Most importantly, the traffic of IoT devices has obvious timing peak characteristics,which can be inferred by machine learning algorithms. For example, have you ever imaged that in a smart home,your neighbourhood or someone faraway can easily know whether you are in or out and what your are doing simply through the status of your IoT devices? To avoid such privacy attacks, it is necessary to apply traffic shaping in IoT networks together with the traditional protection: encryption ,VPN, etc.

In this paper, we focus on cyber security and elaborate how traffic shaping is realized to protect user privacy.

# 2 The Algorithm in Traffic Shaping [2]

Traffic shaping is a measure to actively regulate the data output and it is used to optimize or guarantee performance, improve latency, or increase usable bandwidth for some kinds of packets by delaying other kinds. Traffic shaping is commonly applied at the network edges to control traffic entering the network, but can also be applied by the traffic source (for example, computer or network card) or by an element in the network. The specific technique used in traffic shaping is GTS(Generic Traffic Shaping). And there are two fundamental algorithms in traffic shaping. The first one is Leak Bucket Algorithm, and the second one is Token Bucket Algorithm, which we will explore in the following passage.

## 2.1 Leak Bucket Algorithm

Leak Bucket Algorithm(LBA) is a frequently used algorithm in traffic shaping or rate limiting. It is an algorithm based on an analogy of how a bucket with a constant leak will overflow if either the average rate at which water is poured in exceeds the rate at which the bucket leaks or if more water than the capacity of the bucket is poured in all at once. Its main purpose is to control the rate of data injection into the network and smooth the burst traffic on the network. Leaky bucket provides a mechanism by which burst traffic can be shaped to provide a stable traffic for the network.

[2] We focus on the basic algorithms for further privacy protection and don't explain too much of specific examples of traffic scheduling or the capability to guarantee or improve end-to-end delay.For more detailed information, see reference [6] and [11]

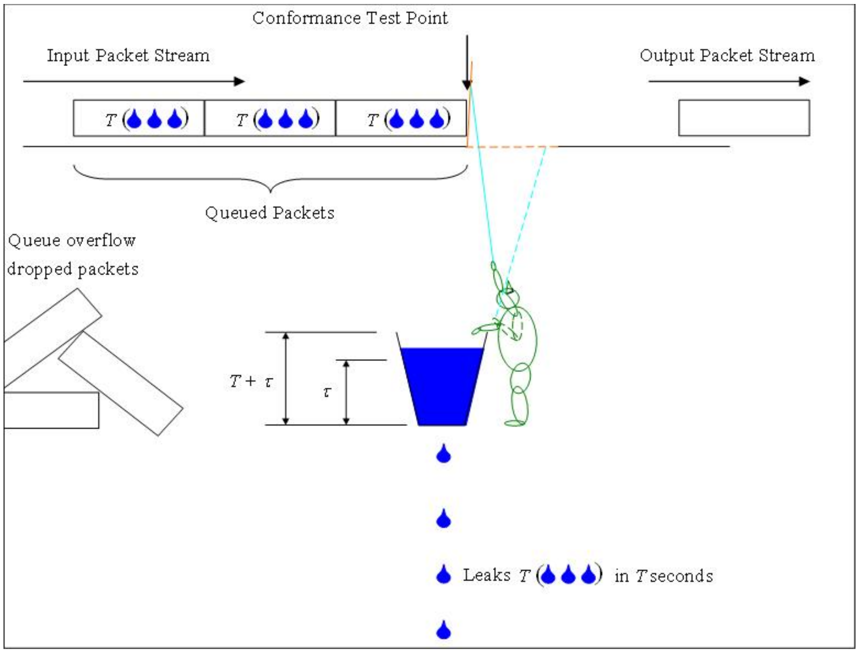


Figure 1: Leak Bucket

A description of the concept of operation of the Leaky Bucket Algorithm as a meter that can be used in either traffic policing or traffic shaping, may be stated as follows:

1) A fixed capacity bucket, associated with each virtual connection or user, leaks at a fixed rate.

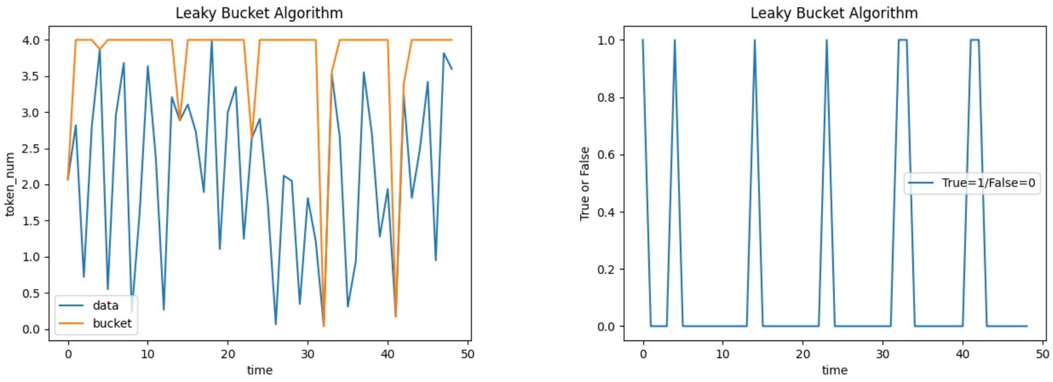
2) If the bucket is empty, it stops leaking.

3) For a packet to conform, it has to be possible to add a specific amount of water to the bucket: The specific amount added by a conforming packet can be the same for all packets, or can be proportional to the length of the packet.

4) If this amount of water would cause the bucket to exceed its capacity then the packet does not conform and the water in the bucket is left unchanged.

We implement LBA(Leak Bucket Algorithm) using python. The code used in this paper can be found at HERE.

In the experiment, we observe how Leak Bucket Algorithm works when data continuously comes in. We initialize the capacity of the bucket, the rate of water leaks and addnum of data. To simulate the real situation, we generating random amount of data, which is easy in Python. We show the performance of Leak Bucket Algorithm in Figure 2. And results are as below:



(a) data in/out rate

(b) flow situation

Figure 2: Result: Leak Bucket

From two graphs above, we get to know the performance of Leak Bucket. In Fig 2.(a), the orange line shows how much data is in bucket. When it reaches the peek(i.e. 4), it means that the bucket is full. When data comes in is huge and lasts for a long period of time, this situation will possibly occurs. In Fig 2.(b), the points at peaks means all data flows in can be transmitted in that cycle, and the points at bottom means data cannot flow out completely, which means some of them will be detained or dropped.

### 2.2 Token Bucket Algorithm

The token bucket algorithm is based on an analogy of a fixed capacity bucket into which tokens, normally representing a unit of bytes or a single packet of predetermined size, are added at a fixed rate. When a packet is to be checked for conformance to the defined limits, the bucket is inspected to see if it contains sufficient tokens at that time. If so, the appropriate number of tokens, e.g. equivalent to the length of the packet in bytes, are removed , and the packet is passed, e.g., for transmission. The packet does not conform if there are insufficient tokens in the bucket, and the contents of the bucket are not changed.

Packets can be treated in various ways:

1. They may be dropped.

2. They may be enqueued for subsequent transmission when sufficient tokens have accumulated in the bucket.

3. They may be transmitted, but marked as being non-conformant, possibly to be dropped subsequently if the network is overloaded.

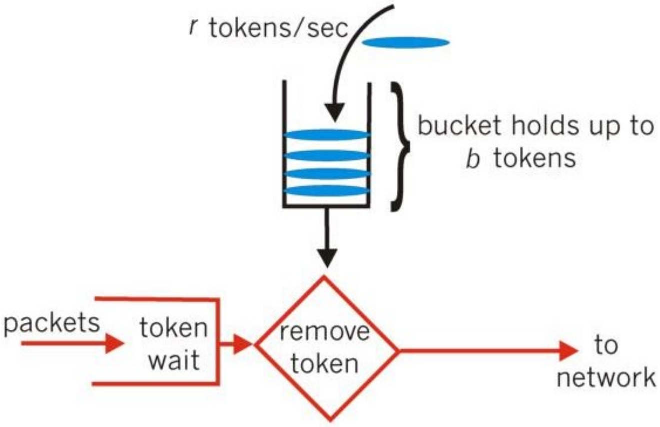
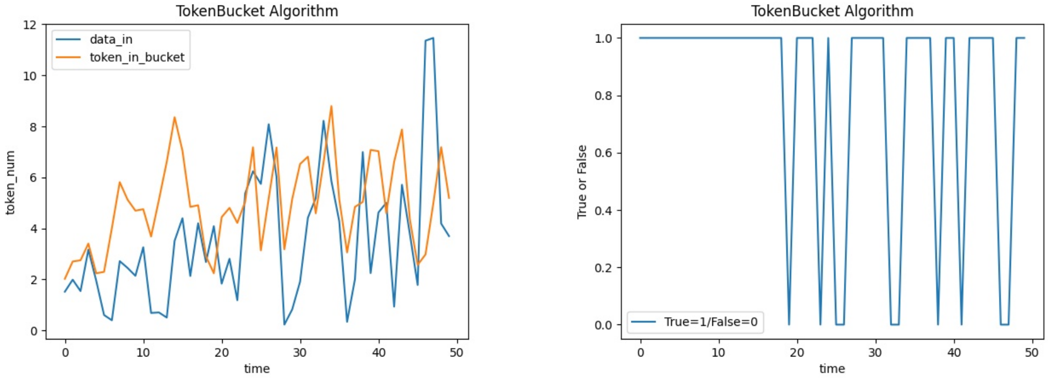


Figure 3: Token Bucket

A description of the concept of operation of the Token Bucket Algorithm is as follow:

1) A token is added to the bucket every 1/r seconds.
2) The bucket can hold at the most b tokens. If a token arrives when the bucket is full, it is discarded.
3) When a packet (network layer PDU) of n bytes arrives: if at least n tokens are in the bucket, n tokens are removed from the bucket, and the packet is sent to the network; if fewer than n tokens are available, no tokens are removed from the bucket, and the packet is considered to be non-conformant.

Similarly, we observe Token Bucket Algorithm's performance. We initialize the capacity of the bucket, the rate of token supplement and addnum of data. To simulate the real situation, we generating random amount of data with a increasing trend to learn about different conditions. We show the performance of Leak Bucket Algorithm in Figure 4. And results are as below:



(a) data in/out rate (b) flow situation

Figure 4: Result: Token Bucket

In Fig 4.(a), the orange line shows how many tokens are in bucket. Due to our setting, the data flow are increasingly larger as time goes by. We find out that although packets are becoming larger, the tokens number maintains in a stable range. In Fig 4.(b), the points at peaks means all data flows in can be transmitted in that cycle. Since packets are small at the beginning, most data can be transmitted successfully. The points at bottom means data cannot flow out completely, which means some of them will be delayed or dropped.

### 2.3 Conclusion

Compared Token Bucket Algorithm with Leak Bucket Algorithm, we discover some difference between the two:

1) The Leak Bucket Algorithm guarantees the certain stable flow rate in the network. As a result, the water in bucket is more likely to reach the peak, especially when large packet arrives.

2) The Token Bucket Algorithm also help smooth the data flows in the network. Nonetheless, this Algorithm is more flexible. And TBA allow burst-transmission, which is an pro when sometimes large packet arrives.

Because the leakage rate of the leak bucket is a fixed parameter, it cannot use network resources effectively compared to token bucket. Owing to the ability of burst-transmission, token bucket is less likely to be full by huge data flow and has higher chance to transmit the whole packet successfully. As a result, we conclude that Token Bucket Algorithm outperforms Leak Bucket Algorithm under certain circumstance.

Both LBA and TBA are very classic algorithm for traffic shaping. And they are basis for our further analysis. Noted that other algorithms and improved versions are proposed nowadays. We will try to explore them in the future after we acquire more knowledge. If you are interested in our work, you can get access to our code from OUR GITHUB, and try it yourself!

## 3 Traffic Shaping for privacy protection

As IoT devices usually are single-purpose devices, the capabilities of individual smart home devices are relatively limited, comprising only a few states or actions. For example, a smart lock can assume only one of two states, locked or unlocked **Figure**(5). Given a certain situation between IoT devices and server, the contents of the exchanged messages or commands are hidden. However, the encryption only hides the payload.Related meta-data (e.g., packet lengths, traffic rate) of the network traffic still leaks some information about the messages exchanged. Attackers can get the source IP, destination IP, sending time, traffic peak, DNS queries, and traffic flow. To protect all these meta-data from leakage, we use VPN to hide its source IP, destination IP and DNS queries, use traffic shaping to hiding time, length, traffic flow and spikes. Although these methods have a great effect on decreasing adversary's confidence, they also cause additional bandwidth and time delay. Thus We need to make a direct trade-off between privacy protection and additional bandwidth consumption. ILP and STP are two typical algorithms showing this trade-off.

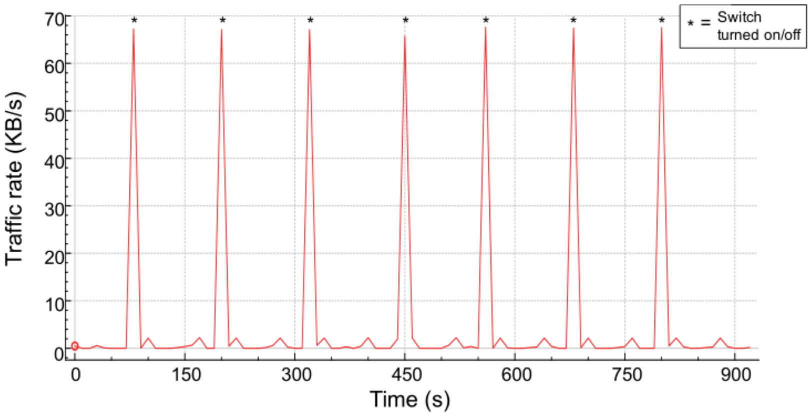


Figure 5: Smart Lock

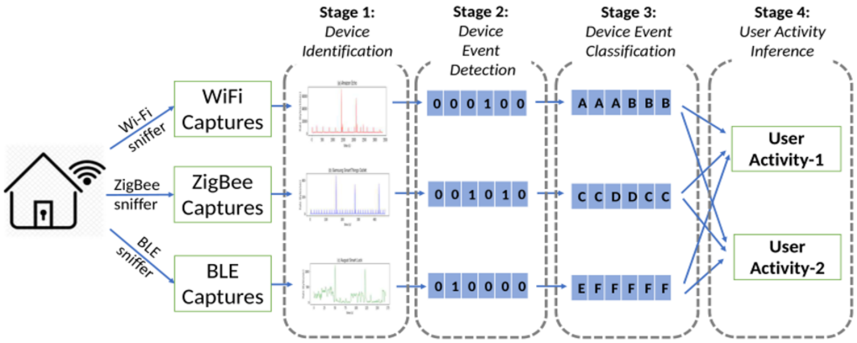


Figure 6: Muti-stage Privacy Attack

### 3.1 The Potential Attack in IoT Network

In this section, we simply introduce the work in [2]. In this paper, the author introduce a novel multi-stage privacy attack on smart home network.The four stages of this attacks can be described in **figure**6.

**Stage 1** Here, in an IoT network, the DNS queries and MAC address can be a fingerprint to identify a device, for example the DNS queries of Samsung ST Outlet may only relate to certain Samsung cloud server. By knowing the MAC address, we can ensure its manufacture which implies the function of the IoT device.

**Stage 2** After discovering the types of individual devices the attacker's goal is to infer the state of individual devices, which is a 0,1 sequence shown in **figure** 6.

**Stage 3** After detecting transitions between device states, the attacker splits the network trace of a device into segments corresponding to different device states (e.g., ON, OFF). Identifying these states is then reduced to a multi-class classification problems, where classes represent possible device states.

**Stage 4** Any user activity in a smart home can be predicted by observing the predicted states of devices and sensors and using a Hidden Markov Model to predict the corresponding user activity. This is rather appalling imaging that someone is watching your every step in a so-called "smart" home!

To sum up, this attack has the following characteristics:

1) **Passive attack**: A cascading style and passive attack which means that the user may not be aware of his privacy leakage.
2) **High accuracy**: Effective on multiple protocols(WIFI,BLE,ZigBee) and various IoT devices.(90+ percent)
3) **Automatic detection and recognition**: Use machine learning algorithms to train a hidden Markov model to automatically identify device types and infer user activities from device status and even predict what the user is going to do!

### 3.2 The traditional privacy protection methods

To defend traffic analysis attack, there are two main traditional ways: firewall and virtual private network(VPN). Although these methods are widely feasible in common networks for hiding users' private information, they are not strong enough to ensure a smart home's safety.We make a simple introduction of how these two strategies work.

**Firewall** This strategy simply block all the sensitive IoT packets from flowing out of the LAN to prevent an adversary from collecting smart home network traffic. We need to recognize the IoT traffic and set up a couple of rules in the router of the smart home LAN. Obviously, this won't stop your neighbourhood (an internal adversary) from sniffing your traffic. More importantly, some IoT devices need cloud service to support their basic capability. Thus, we additionally need to filter the sensitive information, which is hard to implement in practical use.

**VPN** Virtual private network provides authentication and encryption in communication,and encapsulates IP packets to redirect them to the specified server. Therefore, all the packets coming out of the LAN router have the same IP source and destination address.In most cases, the traffic flow is fairly safe with VPN tunnel,but VPN doesn't perform well if the traffic in the LAN that is sparse or when there exists a dominating device in the network. Unfortunately, the IoT traffic in a smart home is sparse most of the time, which allows for the adversary to cluster the traffic model. This attack may be feasible seeing the following **figure** 7 from the work in [3]. In addition, VPN can't deal with an internal adversary either.

Due to the reliance on cloud services, we can't block the necessary IoT traffic through firewall. While tunneling via a VPN does make the attack somewhat more difficult, it does not provide any guarantee of privacy; a dedicated adversary could still use the available traffic rate information from the VPN in machine learning. Thus, we apply traffic shaping to defend both internal and external adversaries from traffic analysis attack in the following sections.

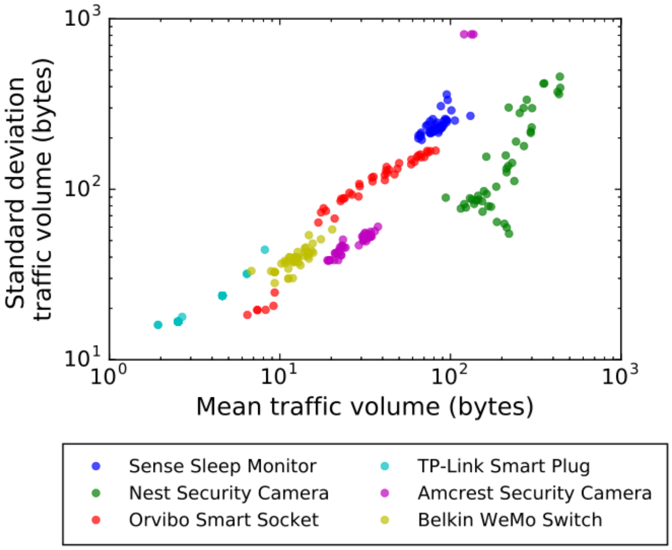


Figure 7: Traffic cluster of IoT devices

## 3.3 Traffic shaping

### 3.3.1 Independent Link Padding (ILP)

One approach to traffic shaping that can guarantee smart home privacy involves sending fixed-size packets at a constant rate independent of the underlying device traffic. This is called the independent link padding (ILP). It's the easiest traffic shaping strategy and also the safest algorithm which completely obscure traffic information. ILP shaper is realized by token bucket algorithm. We generate an extra cover traffic in our LAN router or IoT device[3].A shaping sample is showed in **Figure 8(b)**

Firstly, We treat IoT traffic differently from ordinary traffic since we only shape IoT traffic and do no processing on ordinary traffic. [4]

Next, we create a new PRIO queuing discipline (qdisc). The PRIO qdisc has three priority classes for packets. If packets are available in a high-priority queue, the qdisc will send those packets before sending packets from a lower-priority queue.Therefore, we set a higher priority on real IoT traffic and a lower priority on the cover traffic.

Finally,We use a hierarchical token bucket (HTB) to generate tokens at a constant rate *V*. There is only one token in the token bucket to ensure a constant traffic speed. The generation rate of the cover traffic is faster than the token generation rate which is our shaping traffic constant rate. We set the HTB as the parent of the PRIO qdisc. When a packet is ready to transmit from the PRIO qdisc, either from the high or low priority queue, the interface attempts to

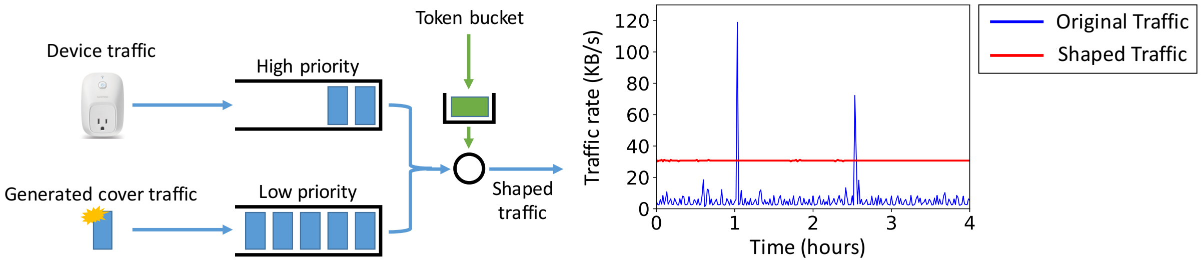
[3] Note: in the following elaboration, we assume the traffic shaper is realized in the LAN router.

[4] If the shaper is implemented in the IoT device, it promises a high security against both internal and external adversary.

**Algorithm 1:** ILP.

```
Input: An HTB token bucket $H$;
A high priority queue $Q_{real}$ containing real packets;
A low priority queue $Q_{fake}$ containing fake packets;
A fixed time interval $t$ for generating tokens(shaped rate);
Function ILP($H$, $Q_{fake}$, $Q_{real}$, $t$):
    while true do
        Sys.sleep(t)
        $H$.generate_token()
        $Q_{fake}$.add(fake_packet)
        p=Get-packet()
        if Traffic-type(p)!=IoT then
            Continue
        $Q_{real}$.add(p)
```

grab a token from the HTB. If a token is available, the packet is sent.Otherwise, the interface must wait for a token to become available.(HTB is only used for the IoT packet, seeing this **Figure** 8(a) from [3])



(a) HTB for ILP (b) flow situation

Figure 8: Token Bucket and shaping traffic

### 3.3.2 Stochastic Traffic Padding (STP)

STP provides an easily tunable trade-off between adversary confidence and bandwidth overhead. Users can choose their own privacy protection level by applying different parameters for STP. STP also imposes no additional network latency and can achieve low adversary confidence for relatively little bandwidth overhead. All in all, it's a classical trade-off between bandwidth overhead and user privacy.

**Concepts** Although ILP provides an absolutely privacy protection for users, it may not be feasible if there exists time-sensitive devices or high traffic-consumption devices. STP is put forward to deal with such situations. The following algorithm is referenced from [2].

**STP** STP divides time into discrete periods of length $T$. All user activity traffic (detected by traffic rate threshold or machine learning methods) is padded to match the pre-selected traffic pattern, preventing an adversary from differentiating activity types based on traffic rate metadata. When there is no user traffic, we set a probability $q$ to generate a fake traffic at the beginning of the period T and the fake traffic will be discarded at the receiving end point. We can set a high enough traffic rate $R$ such that all normal traffic is not delayed and create one traffic generator for each device.Thus the shaped traffic is something in this Figure 9.If the generating rate $q$ tends to 1, STP is equivalent to ILP. The relationship between adversary's confidence $c$ (probability of getting private information with traffic analysis) and extra bandwidth overhead $b$ is :

$$\frac{c}{b} = O(q^{-2}) \qquad (3.3.2)$$

**Algorithm 2:** Stochastic Traffic Padding (STP)

```
padStart ← $0$
padEnd ← $0$
Function STP($t$, $q$, $T$, $R$):
    /* Arguments: current time $t$, non-activity padding probability $q$, time period length $T$,
       padding rate $R$ */
    if $t \bmod T = 0$ and decisionFn($q$, ...) then
        /* decisionFn() draws a random Boolean from a model parameterized on $q$ and (optionally)
           previous user activity. */
        padOffset ← uniformRandom($0, T$)
        if $t$ + padOffset > padEnd then
            padStart ← $t$ + padOffset
            padEnd ← padStart + $T$
        else
            padEnd ← padEnd + $T$
    if padStart $\leq t \leq$ padEnd then
        padTraffic($R$)
    else if userActivityOccurring($t$) then
        padStart ← $t$
        padEnd ← $t + T$
        padTraffic($R$)
```

In paper [2], there is a detailed derivation of this formula 3.3.2.

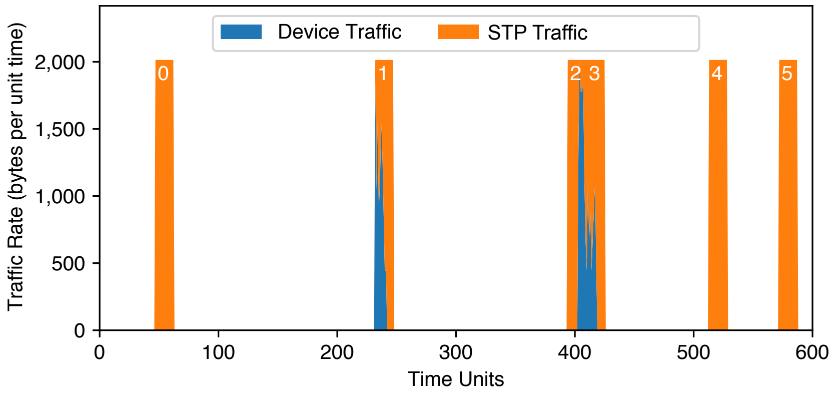


Figure 9: Shaped Traffic of STP

### 3.4 A summary for ILP and STP

In this section, we summarize the characteristics of the above two algorithms and their applicable scenarios. It's obvious that ILP protects most metadata and all traffic information from leakage and provides a high privacy protection level for users. STP provides a optional privacy protection mechanism for users and guarantees quality of service(QoS). Considering the various situations of practical applications, we have summarized the applicability of these two algorithms as follows.

**ILP** ILP always generates cover traffic to ensure user privacy which makes it impossible to set a high shaped rate for the smart home LAN. Applicable scenarios:

- Devices with relatively constant traffic rates.
- Devices that can tolerate long network latencies.
- No high traffic-consumption devices

**STP** STP's performance is closely related to parameter $q$ and $T$.In practical deployment, it's flexible and suitable for most smart home environments.

- Suitable for IoT devices with discrete traffic peaks and no long-term continuous traffic.
- Suitable for time-sensitive devices.
- Flexibility and adjustable privacy requirements

## 4 Random Burst Padding (Our new algorithm RBP)

Although the above two algorithms have a good performance in practical applications, they're both not able to deal with a burst traffic in real application because they have a fixed and constant shaping rate. In addition, from **Figure** 9, we can see that their shaped traffic diagram is square and different from normal traffic(more smooth or with sharp peaks). Therefore, the adversary can easily know whether the traffic is shaped or not.Based on the above points and the core idea of ILP and STP, we propose a new shaping algorithm that can deal with burst traffic and simulate real traffic(which might be more effective). Here we give the pseudo codes of the algorithm random burst padding **Algorithm** 3.

**Algorithm 3:** RBP

**Input:**

$t$: the current time;
$T$: the interval between two judgments;
$x_0$: the threshold to allow a burst transition;
$p$: the probability of shaping when the real traffic needs bursting transition;
$q$: the probability of generating fake traffic;
$Q_{real}$: A high priority queue containing real packets;
$Q_{fake}$: A low priority queue containing fake packets;
$HTB$: The hierarchical token bucket;
$v_0$: the always existing cover traffic;
$v_{max}$: the max allowed burst traffic;

```
Function RBP($t$,$T$,$x_0$,$v_0$,$v_{max}$,$p$,$q$,$Q_{real}$,$Q_{fake}$):
    if $length(Q_{real}) > x_0$ then
        /* decisionFn() draws a random Boolean from a model parameterized on $q$ and (optionally)
           previous user activity.  */
        if decisionFn($p$, ...) then
            /* shaping the burst traffic */
            /* we assume that the transition speed $v$ is equivalent to the size of bucket ,or we
               can find a linear transition, we also allow the shaped traffic speed is lower than
               the original burst speed $\Delta$ */
            burst ← $expRandom(v(length(Q_{real})) - \Delta, v_{max})$;
            HTB.bucketsize ← burst;
        else
            /* no shaping */
            HTB.bucketsize ← $v(length(Q_{real}))$;
    else if $t \bmod T = 0$ and decisionFn($q$, ...) then
        padOffset ← uniformRandom($0, T$);
        if $t$ + padOffset > padEnd then
            padStart ← $t$ + padOffset;
            padEnd ← padStart + $T$;
        else
            padEnd ← padEnd + $T$;
        burst ← $expRandom(0, v_{max})$;
        HTB.bucketsize ← $v_0 + burst$;
        $Q_{fake}$.pad_rate ← uniformRandom($v_0$,$v_0 + burst$);
    else if $t \bmod T = 0$ then
        HTB.bucketsize ← $v_0$;

    /* fullfill the bucket */
    HTB.full();
```

The main idea of RBP can be broken into the following points:

- Variable bucket size: This allows different traffic peaks and makes the output stream more randomized which makes the adversary hard to identify the real traffic. Possibly, the adversary even cannot figure out whether the traffic is shaped or not!
- Cover traffic: Similar to ILP, we use this cover traffic to hide the low-level traffic information (such as connection or basic interactions).This traffic speed is usually less than 3 KB/s and in most LANs, it won't affect the user's surfing experience in the Internet.
- Shaping the real traffic (non-essential): this is an optional item for users. When the burst traffic comes, we can shape our real traffic with probability $p$. Also, we allow some kind of delay without affecting the normal capabilities.

The results of the shaped traffic seems like **Figure** 10

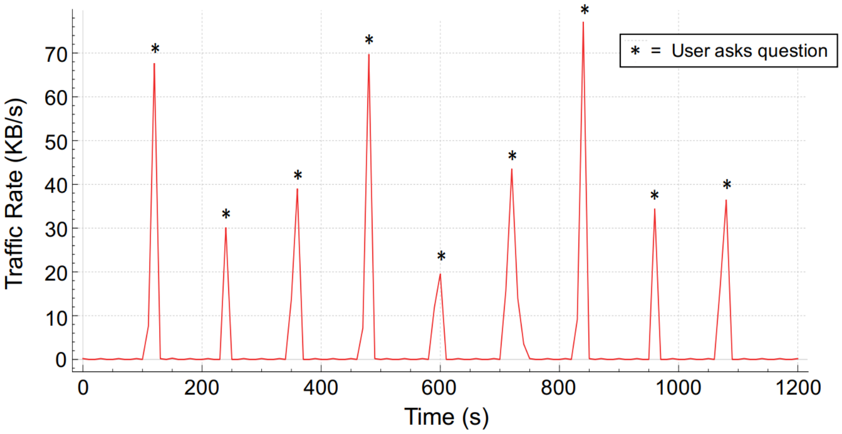


Figure 10: Shaped Traffic of RBP

From the adversary's point of view, all the traffic might be fake and even the peak rate may be lower or higher than the real traffic rate! The **algorithm** 3 is a simple demo for our designed RBP model which might be more complicated while implementing RBP to real smart home LANs. We need to do extra experiments to determine what parameters are more suitable for an actual system. A feasible way is that we can auto-train these parameters while devices are running for a period of time. From the perspective of the token bucket controlling burst traffic transmission, this algorithm is certainly feasible and might be a better trade-off between privacy and extra bandwidth overhead than the previous algorithms.

At last, although all these three traffic shaping is effective in theory, we have not demonstrate its feasibility.In paper [3], the authors give the following reasons of its feasibility:

- First, smart home devices, especially non-A/V devices, use very little network bandwidth compared to smartphones and personal computers.
- Second, smart homes are likely to have Internet speeds and data caps much higher than typically used by consumers. For example, Comcast states that more than 99 percent of consumers do not use their limit of 1TB of data per month.
- Third, traffic shaping overheads do not scale linearly with the number of devices. Adding more devices to a smart home does not necessarily increase the amount of shaping bandwidth needed.

## 5 Conclusion and Future work

In this paper, we first explain and implement(by a Python demo) two classic traffic shaping algorithms: Leaky Bucket and Token Bucket. Then we discuss the widely used algorithms in real scenario—ILP and STP. We verify the feasibility and effectiveness of applying traffic shaping in privacy protection. It becomes harder for bad guys to obtain owner's private information from traffic analysis attack. Nevertheless, the existing methods also have their shortcomings. Based on that, our group propose a new algorithm RBP to allow burst transmission as well as make traffic after shaping more realistic. Finally, we predict the result of our method in **Figure** 10.

We wish our algorithm RBP be a good trade-off between privacy-level and performance. Hopefully more related work about RBP will be done in the future.The future work of RBP may focus on the following points:

- The shaping effect when we also shape the real bursting traffic.
- Implement the RBP algorithm in router or IoT devices.Although implementing the shaper in IoT devices is ideal, it needs the support of IoT manufactures, such as VPN and basic interface which is costly.
- The distribution of randomness. We can test different probability distributions in RBP while choosing the extra burst size. We can use Gauss, exponential or uniform distribution.
- How can we choose a set of proper parameters in a given IoT net? Manual adjustment or smart settings? Anyway, users should be able to choose their privacy level applied in different devices.

What's more, notice that all the traffic shaper we talked above is implemented in the LAN router and therefore we can only defence the external adversary from traffic analysis attack. However, what if an attacker is just someone nearby the smart home who can sniff most of the Wifi or ZigBee traffic? Since our traffic shaper is in the LAN router, a potential internal adversary(your neighborhood) can still perform traffic analysis attacks.Here, we propose two available ways to solve this problem.

1 Implementing the traffic shaper in the IoT device, which demands that the IoT manufactures' support. However, It's expensive and hard to implement.

2 We can set up a convert channel to protect the communication between the IoT devices and the LAN router. This channel is based on characteristics of the device sensor such as gyroscope and is a one-to-one channel which only allows the target device to receive the traffic flow. In the paper [7], this could be possible through a speaker-to-gyroscope channel using the resonant frequencies of gyroscope. Base on this work, we look forward to designing a secret communication protocol to keep it a safe and one-to-one channel or we can take advantage of IoT fingerprints to redesign the channel. Anyway, this channel is worth researching and might be a good match with the traffic shaper.

With the advent of the 5G era, the security of the Internet of Things is particularly important and privacy protection is imperative. Never underestimate potential adversaries on the net!

## References

[1] A. Acar, H. Fereidooni, T. Abera, A. K. Sikder, M. Miettinen, H. Aksu, M. Conti, A. Sadeghi, and A. Uluagac. Peek-a-boo: i see your smart home activities, even encrypted! *Proceedings of the 13th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, 2020.

[2] N. J. Apthorpe, D. Huang, D. Reisman, A. Narayanan, and N. Feamster. Keeping the smart home private with smart(er) iot traffic shaping. *Proceedings on Privacy Enhancing Technologies*, 2019:128 – 148, 2019.

[3] N. J. Apthorpe, D. Reisman, S. Sundaresan, A. Narayanan, and N. Feamster. Spying on the smart home: Privacy attacks and defenses on encrypted iot traffic. *ArXiv*, abs/1708.05044, 2017.

[4] M. Butto, E. Cavallero, and A. Tonietti. Effectiveness of the "leaky bucket" policing mechanism in atm networks. *IEEE J. Sel. Areas Commun.*, 9:335–342, 1991.

[5] K. P. Dyer, S. E. Coull, T. Ristenpart, and T. Shrimpton. Peek-a-boo, i still see you: Why efficient traffic analysis countermeasures fail. *2012 IEEE Symposium on Security and Privacy*, pages 332–346, 2012.

[6] B. Fang, Q. Li, Z. Gong, and H. Xiong. Simulative assessments of credit-based shaping and asynchronous traffic shaping in time-sensitive networking. *2020 12th International Conference on Advanced Infocomm Technology (ICAIT)*, pages 111–118, 2020.

[7] M. Gao, F. Lin, W. Xu, M. Nuermaimaiti, J. Han, W. Xu, and K. Ren. Deaf-aid: mobile iot communication exploiting stealthy speaker-to-gyroscope channel. *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*, 2020.

[8] Guo-LiangWu and J. W.Mark. Discrete time analysis of leaky-bucket congestion control. *Computer Networks and ISDN Systems, Vol.26, Issue.1*, pages 79–94, 1993.

[9] E. Mohammadpour, E. Stai, M. M. Mohiuddin, and J. Boudec. Latency and backlog bounds in time-sensitive networking with credit based shapers and asynchronous traffic shaping. *2018 30th International Teletraffic Congress (ITC 30)*, 02:1–6, 2018.

[10] Y. ning Dong, Z. Wu, and Y. Peng. Traffic shaping based queue management for delay sensitive multimedia applications. *2018 Eleventh International Conference on Mobile Computing and Ubiquitous Network*, 2018.

[11] D. Oehlert, S. Saidi, and H. Falk. Code-inherent traffic shaping for hard real-time systems. *ACM Transactions on Embedded Computing Systems (TECS)*, 18:1 – 21, 2019.

[12] Y. Qian, X. Li, S. Ihara, L. Zeng, J. Kaiser, T. Süß, and A. Brinkmann. A configurable rule based classful token bucket filter network request scheduler for the lustre file system. *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, page 1–12, 2017.

[13] A. Saeed, N. Dukkipati, V. Valancius, V. Lam, C. Contavalli, and A. Vahdat. Carousel: Scalable traffic shaping at end hosts. *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, 2017.

[14] J. Valenzuela, A. Monleon, I. S. Esteban, M. Portoles, and O. Sallent. A hierarchical token bucket algorithm to enhance qos in ieee 802.11: proposal, implementation and evaluation. *IEEE 60th Vehicular Technology Conference, VTC2004-Fall*, 2004.

[15] J. van den Hooff, D. Lazar, M. Zaharia, and N. Zeldovich. Vuvuzela: scalable private messaging resistant to traffic analysis. *Proceedings of the 25th Symposium on Operating Systems Principles*, 2015.

[16] S. Wildhagen, M. A. Müller, and F. Allgöwer. Predictive control over a dynamical token bucket network. *IEEE Control Systems Letters ( Volume.3, Issue.4*, pages 859 – 864, 2019.

[17] Yin, Nanying, Hluchyj, and M. G. Analysis of the leaky bucket algorithm for on-off data sources. *Journal of High Speed Networks, vol.2, no.1*, pages 81–98, 1993.