TADOOP: Mining Network Traffic Anomalies with Hadoop

Geng Tian[1,3], Zhiliang Wang[2,3(✉)], Xia Yin[1,3], Zimu Li[2,3], Xingang Shi[2,3], Ziyi Lu[4], Chao Zhou[4], Yang Yu[1,3], and Dan Wu[1,3]

[1] Department of Computer Science and Technology, Tsinghua University, Beijing, China
[2] Institute for Network Sciences and Cyberspace, Tsinghua University, Beijing, China
wzl@cernet.edu.cn
[3] Tsinghua National Laboratory for Information Science and Technology (TNList), Beijing, China
[4] Cisco Systems, Inc., Shanghai, China

Abstract. Today, various anomalies and large number of flows in a network make traffic anomaly detection a big challenge. In this paper, we propose **DTE-FP** (**D**ual q **T**sallis **E**ntropy for flow **F**eature with **P**roperties), a more efficient method for traffic anomaly detection. To handle huge amount of traffic, based on Hadoop, we implement a network traffic anomaly detection system named TADOOP, which supports semi-automatic training and both offline and online traffic anomaly detection. TADOOP with a cluster of five servers has been deployed in Tsinghua University Campus Network. Furthermore, we compare DTE-FP with Tsallis entropy, and the experimental results show that DTE-FP has much better detection capability than Tsallis entropy.

Keywords: Tsallis entropy · Traffic anomaly detection · Hadoop · Big data · MapReduce

1 Introduction

Today the explosive growth of network size, users and applications generates huge amount of traffic in the Internet. The obvious network traffic fluctuation also reduces the efficiency in traffic anomaly detection. Besides, it is very difficult to use one way to detect all network anomalies, including both known and unknown ones. All of the above make traffic anomaly detection in a network still be a big challenge.

Entropy has been proved to be an effective metric on network traffic anomaly detection [1], [2], [3], and entropy-based methods can detect both known and unknown traffic anomalies. A typical method of entropy-based traffic anomaly detection is to split the traffic into several time bins and compute the entropy value of each time bin for anomaly detection. In recent years, Tellenbach et al. [4] have presented a Traffic Entropy Spectrum (TES) to reveal traffic anomalies.

B. Thuraisingham et al. (Eds.): SecureComm 2015, LNICST 164, pp. 175–192, 2015.
DOI: 10.1007/978-3-319-28865-9_10

The basic idea is using several different Tsallis entropy values corresponding to different parameters to form TES. Berezinski et al. [5] have shown that Tsallis entropy has better performance than Renyi entropy and Shannon entropy. In order to find an easy and efficient method to detect traffic anomalies, we analyze the characteristics of Tsallis entropy for flow feature distributions, and propose **DTE-FP** (**D**ual q **T**sallis **E**ntropy for flow **F**eature with **P**roperties), a new method for anomaly detection. The basic insight is to use the two most efficient q values for highlighting high and low probability feature distributions respectively, which usually imply anomalies in network traffic. DTE-FP contains two parts: DTE and FP. On one hand, we introduce DTE to reveal the high and low probability events in a network. On the other hand, we calculate entropy value for each flow feature with properties (FP). In this way, we can obtain both more concise detection results and more details of the anomalies.

In order to process huge amount of flow data, big data analytics has been widely used to process large scale data set in recent years. An increasing number of people have leveraged MapReduce [6] and Hadoop [7] to mine network traffic anomalies [8], [9], [10]. Zhang et al. [9] have implemented a Shannon entropy based system with Mapreduce. Hodge et al. [10] have proposed a Hadoop based framework for parallel and distributed feature selection. In this paper, we have implemented **TADOOP**, a network **T**raffic **A**nomaly **D**etection system based on had**OOP**, to detect flow-level traffic anomalies. Finally, We have deployed TADOOP with a cluster of five servers in Tsinghua University Campus Network. The experimental results show that our system has strong capability in traffic anomaly detection.

The key contributions of this paper are described as follows:

- First, we analyze the characteristics of Tsallis entropy for flow feature distributions, and present a new traffic anomaly detection method DTE-FP.
- Second, we implement TADOOP, which supports semi-automatic training, offline detection and online detection, deploy our system with a cluster of five servers in Tsinghua University Campus Network.
- Third, we compare DTE-FP with Tsallis entropy, and the results show that DTE-FP performs much better than Tsallis entropy in traffic anomaly detection.

The paper is organized as follows. Section 2 introduces the related work. In Section 3, we analyze the characteristics of Tsallis entropy for network traffic anomaly detection, and describe the details of DTE-FP. In Section 4, we describe the implementation of TADOOP. Then the experimental results are presented in Section 5. In Section 6, we emphasis on which kinds of anomalies can be detected. Finally, Section 7 concludes this paper.

2 Related Work

Nowadays, network traffic anomaly detection [11] is still a big challenge for the explosive growth of network traffic, so that big data analytics is very necessary

for network traffic analysis because of their online and offline detection capability. There are several studies on network traffic analysis based on big data analytics, e.g. [8], [12]. Till now, MapReduce [6] and its open source implementation Hadoop [7] are still the most popular big data programming model and platform used in network traffic analysis. As a representative work of network traffic analysis with Hadoop, Lee et al. [8] presented a Hadoop-based traffic monitoring system that can perform network traffic analysis of both packet-level and flow-level. However, this work was limited to simple IP packet statistics of the traffic with Hadoop.

Shannon entropy has been proved as a good metric in network traffic anomaly detection [1], [13], [4], [3], [14], and has shown stronger anomaly detection capability than volume-based methods [1]. Zhang et al. [9] implemented a Shannon entropy based system with MapReduce. Besides Shannon entropy, Tsallis proposed and analyzed Tsallis entropy in their works [15], [16], [17]. However, the first work for using Tsallis entropy in anomaly detection was introduced in 2007 [13]. After that, Tellenbach et al. [4] proposed a Traffic Entropy Spectrum (TES) method, in which different entropy values corresponding to different parameters are used to form TES to reveal anomalies. Berezinski et al. [5] presented that Tsallis entropy had better performance than Renyi entropy and Shannon entropy.

Entropy based detection method usually splits time into several time bins, and calculates entropy values for flow feature distributions in each time bin [1]. Lakhina et al. used entropy values of source IP address/port and destination IP/port for traffic anomaly detection [1]. Besides above feature distributions, Nychis et al. employed Shannon entropy for in-degree, out-degree and flow size distribution (FSD) to mine more anomalies [3]. In this paper, besides source IP/port and destination IP/port, we not only introduce flow byte for traffic anomaly detection, but also use flow feature with properties instead of one single feature. For example, we use source IP as the main flow feature, and use flow direction, protocol and TCP control bit as its properties.

3 DTE-FP

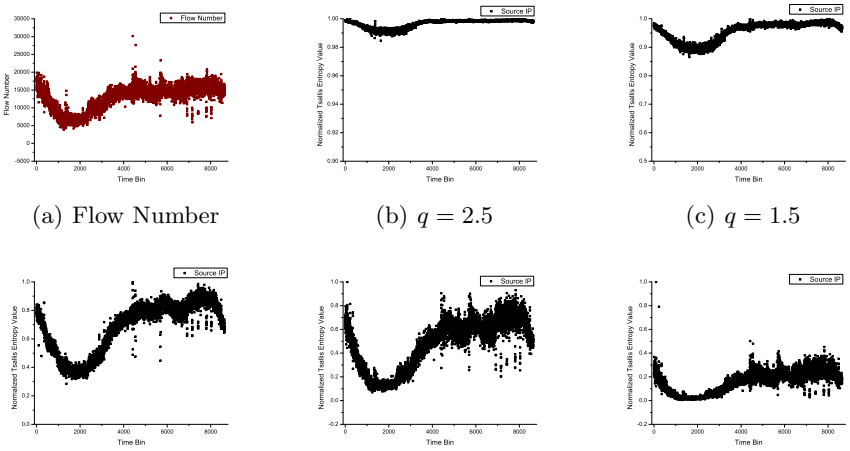
Entropy has been proved to be an efficient matric for traffic anomaly detection, and widely used in anomaly detection systems. A typical mode of entropy-based traffic anomaly detection is to split the traffic into several time bins and compute the entropy values of flow feature distributions for all time bins. Finally, we can find out the anomalous time bins, in which their entropy values deviate much from the normal ones. In this section, we propose a new Tsallis entropy based method for traffic anomaly detection.

3.1 Tsallis Entropy Characteristics for Anomaly Detection

Tsallis entropy $S_{Ts} = \frac{k}{q-1}(1 - \sum_{i=1}^{n} p_i^q)$ performs well in traffic anomaly detection. The parameter q means different sensitivity for different probability events, which makes Tsallis entropy flexible and efficient for traffic anomaly detection. We will illustrate its characteristics from two aspects: stability for normal flows and sensitivity for anomalous flows.

Stability for Normal Flows. The flow numbers in a network change all the time, especially between day and night, which makes Tsallis entropy values change with the flow numbers. In order to know the effect of flow numbers in a whole day, we analyze the flow data with few anomalies for the period 0:00 to 24:00. We divide the whole time into 8640 time bins by using 10s as the time interval. We then obtain Tsallis entropy values with different q values, such as $q = 2.5, 1.5, 0.5, -0.5, -1.5$, because the work of Tellenbach et al. finds that the selection $q = 2, 1.75, ..., -1.75, -2$ gives sufficient information to detect network anomalies [4]. We normalize Tsallis entropy values by dividing by the max entropy value for each feature. As shown in Fig. 1, we can find that the flow numbers obviously decrease in the night while increase during the daytime. The corresponding Tsallis entropy values for source IP address decrease when there are few flows in the night, and increase when there are many flows in the day. Furthermore, Fig. 1 also shows that Tsallis entropy with a bigger q value is more stable and less effected by flow numbers.

Sensitivity for Anomalous Flows. In order to test the sensitivity of Tsallis entropy for different q values, we randomly select a normal data with 60 time tins. We inject a DDoS attack of 20k flows into time bin #30. In this attack, a large number of source IP addresses and ports launch a SYN flood to a same destination IP and port. As shown in Fig. 2, we can find that a small q value results in an obvious fluctuation of entropy values, while a bigger q means a more steady entropy value. We can also find that the Tsallis entropy value of destination IP address will decrease sharply when $q > 1$, while the entropy value for source IP address has no obvious increase. We thus observe that Tsallis entropy is sensitive to high probability elements but insensitive to low probability elements when $q > 1$. Furthermore,



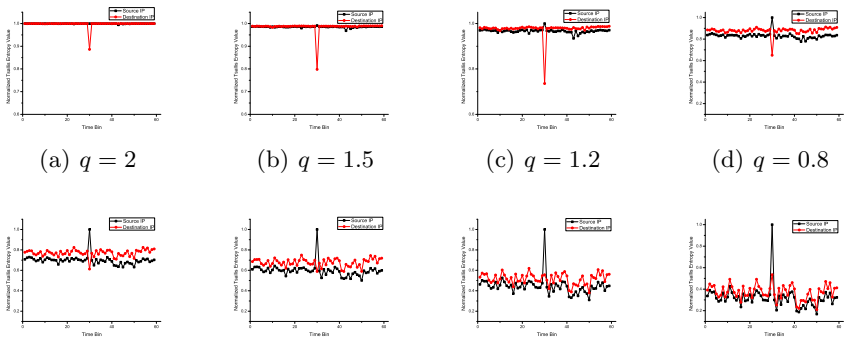
(a) Flow Number (b) $q = 2.5$ (c) $q = 1.5$

(d) $q = 0.5$ (e) $q = -0.5$ (f) $q = -1.5$

Fig. 1. Stability for Normal Flows

the Tsallis entropy value for source IP address increases sharply, while the entropy value for destination IP address decreases smaller, even increases when $q < 1$. The situation above means Tsallis entropy is sensitive to low probability elements but insensitive to high probability elements when $q < 1$.

Therefore, the characteristic of Tsallis entropy can be summarized into following points: (1) A bigger q value is less effected by total normal flow numbers when $q > 1$. (2) Tsallis entropy is sensitive to high probability elements when $q > 1$, and sensitive to low probability elements when $q < 1$.



(a) $q = 2$ (b) $q = 1.5$ (c) $q = 1.2$ (d) $q = 0.8$

(e) $q = 0.5$ (f) $q = 0.2$ (g) $q = -0.5$ (h) $q = -1.5$

Fig. 2. Sensitivity of Tsallis Entropy for The DDoS Attack

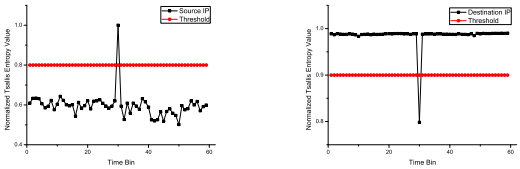
3.2 DTE-FP

According to the characteristic of Tsallis entropy for the normal flows and anomalous flows, we propose DTE-FP, a new method for traffic anomaly detection. The basic insight is to use the two most efficient q values for highlighting high and low probability feature distributions respectively, which usually imply anomalies in network traffic. DTE-FP contains two aspects: DTE and FP. For one thing, we present dual q Tsallis entropy (DTE), whose definition is shown in Definition 1. For another, we calculate entropy for each flow feature with properties (FP).

Definition 1.

$$S_{DTE} =< S_L, S_H >, where$$
$$\begin{cases} S_L = \frac{k}{q_l - 1}(1 - \sum_{i=1}^{n} p_i^{q_l}), (q_l < 1) \\ S_H = \frac{k}{q_h - 1}(1 - \sum_{i=1}^{n} p_i^{q_h}), (q_h > 1) \end{cases} \quad (1)$$

DTE for Detection. In DTE, a pair of q value $< q_h, q_l >$ is employed for different anomalies. We use q_h and q_l to detect the anomalies with high and low probability feature distribution respectively. As shown in Fig. 3, if a DDoS attack happens in time bin #30, we can find the entropy value for source IP address exceeds its



(a) q < 1 (b) q > 1

Fig. 3. DTE for DDoS Attack

upper threshold, and the entropy value for destination is below its lower threshold. Therefore, a suitable q could find more anomalies. Note that we should guarantee that $q_h > 1$ and $q_l < 1$. Normally, we can select q_h and q_l by training.

FP. Usually, the flows of an attack have the similar pattern. For example, the flows of a DDoS attack have the same destination IP, destination port, protocol number and TCP control bit. However, we cannot use each traffic feature to compute entropy values, such as protocol numbers, because the entropy values for protocol number distributions have little information, and they can hardly help us to detect anomalies. But if we select some flow features as main features and other features as their properties, we will obtain more precise results. As shown in Fig. 4, we choose source IP/port, destination IP/port, and flow byte as the main features, and use time bin of flow, flow direction, protocol number and TCP control bit as the properties of main features. We use the time bin and flow direction to divide the traffic into different feature distributions. Protocol number and TCP control bit help to compute entropy value for each feature distribution. FP will not only help to obtain more concise results, but also provide more details of the anomalies for more detailed classification.

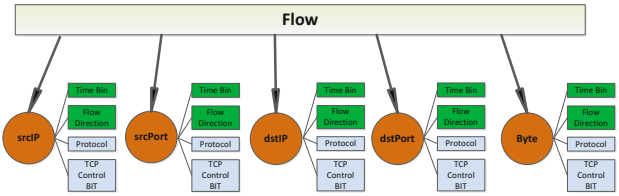
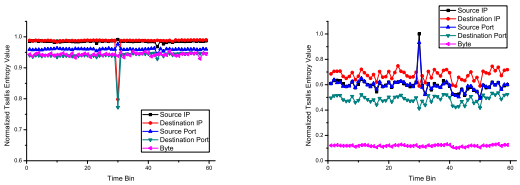


Fig. 4. Flow Feature with Properties

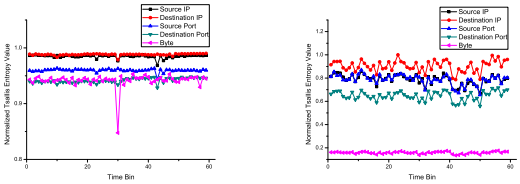
3.3 Detection for Common Attacks

DDoS. As shown in Fig. 5, in time bin #30, we inject a DDoS attack of 20k flows, in which a large number of source IP addresses and ports launch a SYN



(a) q = 1.5 (b) q = 0.2

Fig. 5. DDoS Attack



(a) q = 1.5 (b) q = 0.2

Fig. 6. Spam

Table 1. Relationships between DTE-FP and Typical Traffic Anomalies

Anomaly	$q < 1$				$q > 1$					Protocol	TCPctrlBit
	sIp	sPt	dIp	dPt	sIp	sPt	dIp	dPt	byte		
DoS: SYN flood					↓		↓	↓		6	2
DoS: ACK flood					↓		↓	↓		6	18
DoS: UDP flood					↓		↓	↓		17	0
DoS: ICMP flood					↓		↓	↓		1	0
DDoS: SYN flood	↑						↓	↓		6	2
DDoS: ACK flood	↑						↓	↓		6	18
DDoS: UDP flood	↑						↓	↓		17	0
DDoS: ICMP flood	↑						↓	↓		1	0
DRDoS			↑	↑	↓	↓				6	2
PortScan1			↑		↓			↓		1/6/17	0/2/0
PortScan2				↑	↓		↓			1/6/17	0/2/0
Spam									↓	6	-
Worm								↓	↓	6/17	-

flood to a same destination IP and port. Then we can find the entropy value for source IP address in this time bin sharply increases when $q < 1$, and entropy value for destination IP address and port obviously decreases when $q > 1$.

Spam. As shown in Fig. 6, if a spam of 2k flows happens in time bin #30, the Tsallis entropy value for the flow byte feature will decrease sharply.

Other Attacks. We can use DTE-FP to detect the anomalies which deviate from the normal situation. Table 1 introduces the relationships between entropy and the typical traffic anomalies.

4 Implementation of TADOOP

In this section, we describe the architecture and implementation of TADOOP. As shown in Fig. 7, our system consists of a traffic collector, a entropy calculation module, a training module, a detection module and a web-based interface.

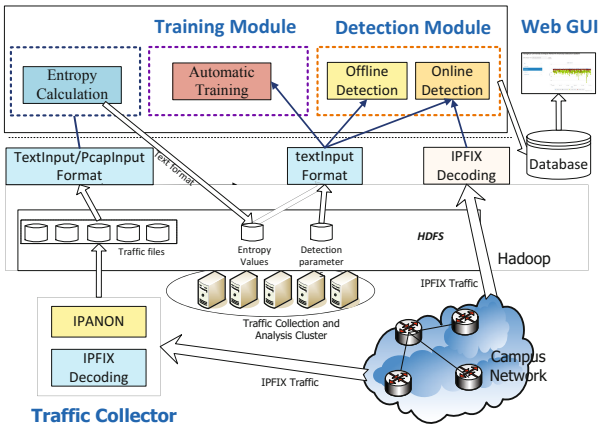


Fig. 7. Architecture of TADOOP

4.1 Traffic Collector

The traffic collector receives NetFlow packets from the edge routers of an AS or edge network, and supports NetFlow v5 and IPFIX format flow data. We leverage "libipfix" [18] to decode IPFIX format data and use "p3" [8] to decode NetFlow v5 format data. Besides, we anonymize all IP addresses by "IPANON".

4.2 Entropy Calculation Module

Entropy calculation module aims at computing Tsallis entropy value pairs for each flow feature distribution. We implement this module in MapReduce framework. In our system, we use one-round MapReduce to achieve above function. Algorithm *Tsallis.map* aims to extract and transform flow information. First, it extracts flow features from each flow (line 2), and obtains flow direction and time bin value (line 3-4), then outputs the final flow features with properties (line 5-20). We divide all flows into outside flows, incoming flows, outgoing flows

Algorithm 1. Tsallis.map

Input: The set of flow records decoded from NetFlow file (FR), the length of each time bin (L), the set of owner As numbers (AS)
Output: The set of new $< key, value >$ pairs (MS)

1 **foreach** *flow* $f \in FR$ **do**
2 $Extract(sIp, sPt, dIp, dPt, Bt, srcAs, dstAs, pro, bit, endTime)$ from flow f;
3 $fD \leftarrow flowDirection(srcAs, dstAs, AS)$;
4 $tNum \leftarrow endTime/L$;
5 %form new flow feature with property $newSrcIp \leftarrow fD + tNum$;
6 $newSrcIpVal \leftarrow$ "sIp" $+ sIp + pro + bit + 1$;
7 $newSrcPt \leftarrow fD + tNum$;
8 $newSrcPtVal \leftarrow$ "sPt" $+ sPt + pro + bit + 1$;
9 $newDstIp \leftarrow fD + tNum$;
10 $newDstIpVal \leftarrow$ "dIp" $+ dIp + pro + bit + 1$;
11 $newDstPt \leftarrow fD + tNum$;
12 $newDstPtVal \leftarrow$ "dPt" $+ dPt + pro + bit + 1$;
13 **if** $bit == 19 || bit == 27 || bit == 31$ **then**
14 $newByte \leftarrow fD + tNum$;
15 $newByteVal \leftarrow$ "Bt" $+ Bt + pro + bit + 1$;
16 $MS \leftarrow MS \cup < newByte, newByteVal >$;
17 $MS \leftarrow MS \cup < newSrcIp, newSrcIpVal >$;
18 $MS \leftarrow MS \cup < newSrcPt, newSrcPtVal >$;
19 $MS \leftarrow MS \cup < newDstIp, newDstIpVal >$;
20 $MS \leftarrow MS \cup < newDstPt, newDstPtVal >$;

and inner flows. For example, if both the source and destination AS number belongs to the network, the flow is inner flow.

Algorithm $Tsallis.reduce$ is in charge of obtaining the final Tsallis entropy value pairs for each flow feature distribution. First, it classifies all flow features into five hash maps for source IP address, source port, destination IP address, destination port and flow byte. It then employs function *update_hm* to compute the occurrence number of the same flow features and combine them into one $< key, value >$ pair (line 2-14). Second, it uses function $TsallisEn$ to calculate Tsallis entropy value pairs for all flow features in each time bin (line 16-18). At last, it outputs the results (line 20-25).

4.3 Semi-automatic Training Module

Training module helps us to obtain the detection thresholds for all flow feature distributions. First of all, we select a long time flow data for training, and mark anomalous time bin for each flow feature. We then use Algorithm $AutoTrain$ to obtain the final threshold pair for each feature distribution. As described in Algorithm 4 $AutoTrain$, we employ a while-loop to obtain the fine threshold pairs step by step. We set the max false positive rate (MFPR) as the termination condition. If the current false positive rate $fp1$ or $fp2$ is smaller than MFPR,

Algorithm 2. Tsallis.reduce

Input: The set of $< key, value - list >$ pairs (MS), q_l, q_h
Output: Tsallis entropy file EF

foreach $< key, value - list > \in MS$ **do**
  create $HashMap < String, int >$ set (HS) from hm_1 to hm_5;
  foreach $val \in value - list$ **do**
    $< flowObj, sum > \leftarrow split(val)$;
    if $flowObj.contains(\text{“}sIp\text{”})$ **then**
      $update\_hm(flowObj, hm\_1)$;
    else if $flowObj.contains(\text{“}sPt\text{”})$ **then**
      $update\_hm(flowObj, hm\_2)$;
    else if $flowObj.contains(\text{“}dIp\text{”})$ **then**
      $update\_hm(flowObj, hm\_3)$;
    else if $flowObj.contains(\text{“}dPt\text{”})$ **then**
      $update\_hm(flowObj, hm\_4)$;
    else if $flowObj.contains(\text{“}Bt\text{”})$ **then**
      $update\_hm(flowObj, hm\_5)$;
  % compute Tsallis entropy;
  foreach $hm\_i \in HS$ **do**
    $S_{q_l\_i} \leftarrow TsallisEn(hm\_i, q_l)$;
    $S_{q_h\_i} \leftarrow TsallisEn(hm\_i, q_h)$;
  % output Tsallis entropy;
  $srcIpEntro \leftarrow < S_{q_l\_1}, S_{q_h\_1} >$;
  $srcPtEntro \leftarrow < S_{q_l\_2}, S_{q_h\_2} >$;
  $dstIpEntro \leftarrow < S_{q_l\_3}, S_{q_h\_3} >$;
  $dstPtEntro \leftarrow < S_{q_l\_4}, S_{q_h\_4} >$;
  $byteEntro \leftarrow < S_{q_l\_5}, S_{q_h\_5} >$;
  $entropy \leftarrow srcIpEntro + srcPtEntro + dstIpEntro + dstPtEntro$
  $EF \leftarrow EF \cup < key, entropy >$;

Threshold T_{q_l} minuses δ or T_{q_h} pluses δ. We obtain the final results when the while-loop is over.

4.4 Detection Module

The detection module includes both an offline detection module and an online detection module. The offline detection module is running on the whole Hadoop platform, while the online detection module is running on a single node.

Offline Detection Module. Offline detection module can detect all the historical data and find the anomalies. It calls the entropy calculation module to compute the entropy values for all time bins, then uses the thresholds obtained from training to detect anomalies.

Algorithm 3. TsallisEn

Input: HS, k, q_l, q_h
Output: $< S_{q_l}, S_{q_h} >$ value pairs

foreach $hm_i \in HS$ **do**
  $total \leftarrow 0$;
  foreach $val \in hm_i$ **do**
    $total \leftarrow total + sum$;
    List $lst \leftarrow val$;
  $total1 \leftarrow 0$;
  $total2 \leftarrow 0$;
  foreach $val \in lst$ **do**
    $sum1 \leftarrow sum1 + (\frac{val}{total})^{q_l}$;
    $sum2 \leftarrow sum2 + (\frac{val}{total})^{q_h}$;
  $S_{q1} \leftarrow \frac{k}{q1-1} \times (1 - sum1)$;
  $S_{q2} \leftarrow \frac{k}{q2-1} \times (1 - sum2)$;
  Output $< S_{q1}, S_{q2} >$ value pair

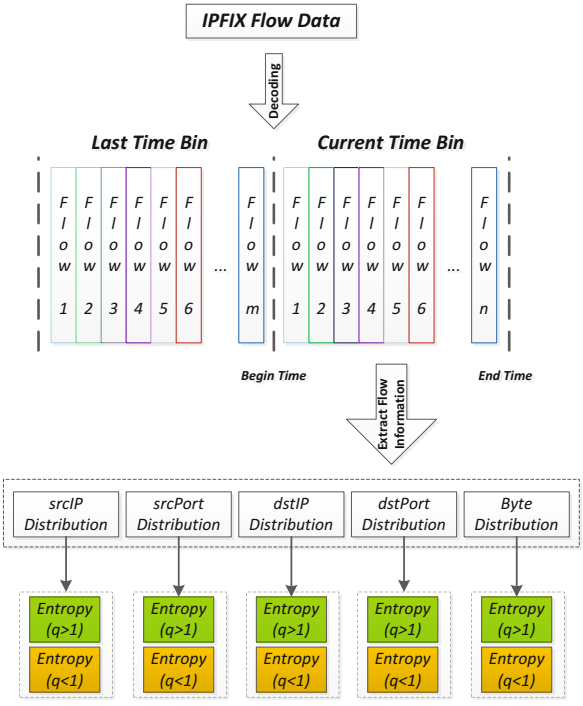


Fig. 8. Entropy Calculation for Online Detection

Algorithm 4. AutoTrain

Input: Entropy value file $eFile$, time bin list for all flow features (LST), the max false positive rate $MFPR$, increase/decrease degree δ, q_l, q_h
Output: Threshold T_{q_l}, T_{q_h}, false positive rate pair $< fp1, fp2 >$, false negative rate pair $< fn1, fn2 >$

```
% obtain initial threshold value;
T_{q_h} ← 0;
T_{q_l} ← 1;
foreach flow feature i do
    runFlag1 ← true;
    runFlag2 ← true;
    while runFlag1||runFlag2 do
        foreach line ∈ eFile do
            < timeBin, S_{q_l}, S_{q_h} > ← readEntro(line, i);
            if (S_{q_l} > T_{q_l})&runFlag1 then
                list1.add(timeBin);
            if (S_{q_h} < T_{q_h})&runFlag2 then
                list2.add(timeBin);
        < fp1_i, fn1_i > ← compare(list1, LST);
        < fp2_i, fn2_i > ← compare(list2, LST);
        if fp1 < MFPR then
            T_{q_l}_i ← T_{q_l}_i − δ;
        else
            break;
            T_{q_l}_i ← T_{q_l}_i + δ;
            runFlag1 ← false;
        if fp2 < MFPR then
            T_{q_2}_i ← T_{q_2}_i + δ;
        else
            break;
            T_{q_2}_i ← T_{q_2}_i − δ;
            runFlag2 ← false;
        if !(runFlag1||runFlag2) then
            Output < T_{q_l}_i, T_{q_h}_i, fp1_i, fn1_i, fp2_i, fn2_i >;
```

Online Detection Module. Online detection module achieves online detection without employing a distributed processing. It consists of two parts: entropy calculation and anomaly detection. The entropy calculation part aims to calculate entropy values for the current time bin. As shown in Fig. 8, after decoding the NetFlow format data into text format flow information, the online detection module extracts flow features with properties between the begin time and the end time, and calculates entropy values for all flow features in this time bin after

the end time. Finally, we obtain the detection results by comparing with the thresholds, and show them on web page.

5 Experiments

5.1 Experiment Environment

We have deployed TADOOP with a cluster of five servers in Tsinghua University Campus Network. Each server integrates two 2.60 GHz Intel Xeon E5-2630 CPU with 12 cores, 32G memory and 9T hard disk. The five servers are connected with a Gigabit Ethernet switch.

5.2 Data

We study the proposed anomaly detection methods using 1.3T IPFIX format flow data collected from one edge router of Tsinghua University Campus Network for the period from 2014-3-2 23:39:20 to 2014-3-11 13:03:00. The sampling ratio is 1:1. For our experiment, we use the data of the period from 2014-3-2 23:39:20 to 2014-3-6 10:58:00 for training, and use the rest data to detect traffic anomalies.

5.3 Detection in Tsinghua University Campus Network

In order to make comparisons, we leverage both Tsallis entropy and DTE-FP to detect anomalies of incoming flows, in which only the destination IP addresses belong to Tsinghua University.

Training for Detection Parameters. Before actual detection, the detection parameters and thresholds should be obtained by the training module. Therefore, we must obtain a fixed time interval, a suitable $< q_h, q_l >$ value pair and all thresholds for the used flow features in the training phase.

Tian et al. [19] shows that a smaller time interval is more sensitive for detecting traffic anomalies by Shannon entropy, because there are less flows in a time bin, and it is more likely for us to find the anomalies of a certain scale. Additionally, we also find that, in a time bin of too many flows, some traffic anomalies will be masked in our Tsallis entropy based method too. Therefore, we refer to the experiment parameter in [19] and set 10s as our time interval.

The work of Tellenbach et al. finds that the selection $q = 2, ..., -2$ gives sufficient information to detect network anomalies [4]. According to the experiments shown in Fig. 1 and Fig. 2, we also find that the q of a too big or small value will not results in a good detection result. Therefore, we select $q = 1.5, 1.1, 0.8, 0.2, -0.5$ as q value candidates for DTE-FP, and use these q values to calculate Tsallis entropy values for each flow feature. For calculating the actual false positive rate, we analyze the whole training data, and both find out and label all anomalous time bins for each feature. In order to obtain good detection results, we ignore the time bins whose flow numbers is under 2k in

training and detection, because it is likely that a lot of flows which should be in these time bins were lost in the collection process.

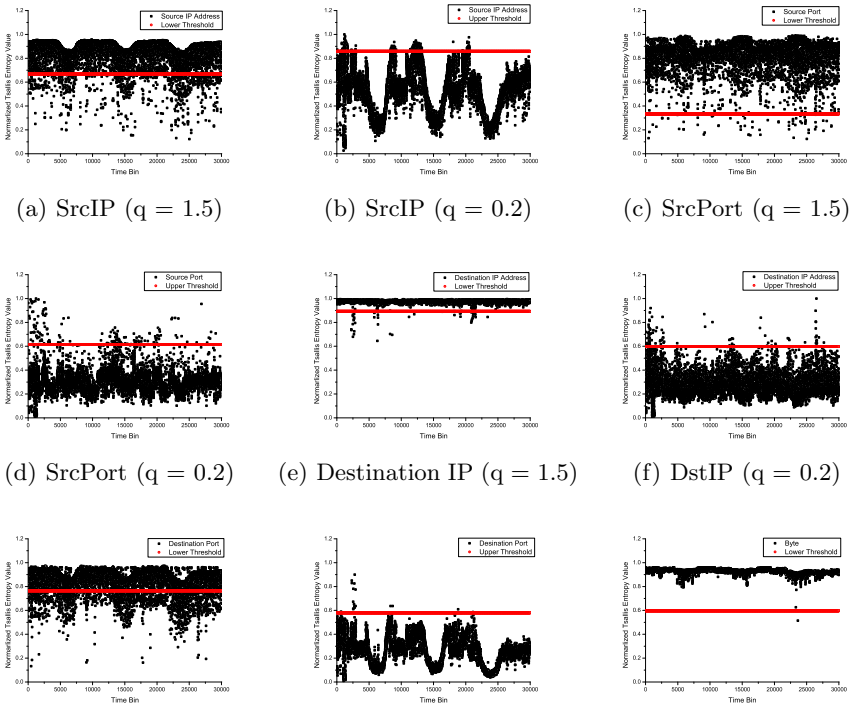
After that, we set MFPR as 0%, 1%, 2% and 5% respectively, and employ Algorithm 4 *AutoTrain* to obtain the upper threshold (UT) and lower threshold (LT) for each feature. Table 2 shows the detection thresholds, anomaly number (AN) and false positive number (FPN) for the training data. From this table, for all MFPR values, we can clearly find that the detection capability for the lower thresholds decrease obviously when q value becomes smaller from 1.5 to -0.5, and we even cannot detect any anomaly by the lower threshold when $q = -0.5$. However, the detection capability for the upper threshold becomes stronger when q changes from 1.5 to 0.2, and it has the best detection capability when $q = 0.2$. Therefore, we can use both the best q value for upper thresholds $q = 0.2$ and the best q value for lower thresholds $q = 1.5$ to form the $< q_h, q_l >$ value pair of DTE-FP.

Detection Capability Comparison. After training, we can mine traffic anomalies from the flow data for detection, Fig. 9 shows the detection results by DTE-FP when $MFPR = 5\%$, and all results are summarised in Table 3.

Table 2. Detection Thresholds and Capability in Training

FP	q	Lower Threshold (LT) & Upper Threshold (UT)					AN & FPN		
		sicip	srcpt	dstip	dstpt	byte	LT	UT	both
0%	1.5	0.625,0.965	0.329,0.988	0.891,0.994	0.732,0.979	0.597,1	1860,0	36,0	1860,0
	1.1	0.488,0.999	0.233,0.955	0.573,0.967	0.500,0.984	0.421,1	1243,0	73,0	1245,0
	0.8	0.327,0.968	0.107,0.862	0.235,0.869	0.234,0.877	0.019,1	652,0	95,0	678,0
	0.2	0.081,0.862	0.023,0.618	0.026,0.604	0.027,0.581	0.008,1	43,0	303,0	332,0
	-0.5	0,1.000	0,0.578	0,0.392	0,0.515	0,1	11,0	123,0	134,0
	DTE	0.625,0.862	0.329,0.618	0.891,0.604	0.732,0.581	0.597,1	1860,0	303,0	2003,0
1%	1.5	0.631,0.965	0.330,0.988	0.892,0.994	0.743,0.979	0.597,1	1862,1	36,0	1862,1
	1.1	0.493,0.999	0.233,0.955	0.573,0.967	0.512,0.984	0.421,1	1392,2	73,0	1394,2
	0.8	0.330,0.968	0.235,0.862	0.107,0.869	0.239,0.877	0.019,1	668,2	95,0	693,2
	0.2	0.330,0.861	0.107,0.618	0.235,0.603	0.239,0.581	0.019,1	43,0	311,0	340,0
	-0.5	0,1.000	0,0.578	0,0.391	0, 0.515	0, 1	11,0	125,0	136,0
	DTE	0.631,0.861	0.329,0.618	0.891,0.603	0.732,0.581	0.597,1	1862,1	311,0	2011,1
2%	1.5	0.638,0.965	0.331,0.988	0.893,0.993	0.750,0.979	0.597,1	2112,3	58,0	2112,3
	1.1	0.500,0.999	0.234,0.955	0.573,0.966	0.522,0.984	0.421,1	1495,3	76,0	1497,3
	0.8	0.335,0.968	0.108,0.862	0.235,0.868	0.243,0.877	0.019,1	753,3	97,0	778,3
	0.2	0.081,0.861	0.023,0.617	0.026,0.602	0.027,0.581	0.008,1	43,0	313,0	342,0
	-0.5	0,1.000	0, 0.578	0, 0.390	0, 0.515	0, 1	11,0	126,0	137,0
	DTE	0.638,0.877	0.331,0.613	0.893,0.599	0.750,0.580	0.597,1	2112,3	313,0	2256,3
5%	1.5	0.667,0.965	0.334,0.988	0.893,0.991	0.765,0.979	0.597,1	2344,4	130,0	2344,4
	1.1	0.525,0.999	0.235,0.954	0.573,0.964	0.536,0.984	0.421,1	1665,5	88,0	1668,5
	0.8	0.350,0.968	0.110,0.861	0.235,0.866	0.252,0.877	0.019,1	891,5	102,0	915,5
	0.2	0.082,0.859	0.023,0.615	0.026,0.598	0.028,0.580	0.008,1	44,1	333,0	363,1
	-0.5	0,1.000	0, 0.577	0, 0.385	0, 0.515	0, 1	11,0	131,0	142,0
	DTE	0.667,0.859	0.334,0.615	0.893,0.598	0.765,0.580	0.597,1	2344,4	333,0	2502,4

From this table, we can find that DTE-FP has better detection capability than any single Tsallis entropy. If a bigger MFPR is selected, more anomalies will be detected. We validate the anomalous time bins by using automatic and manual check. If a time bin has obvious heavy hitters, we mark it as anomalous time bin. For the rest detected ones, we manually check them by flow feature distribution. For example, as shown in Fig. 10(a), by checking the heavy hitters for source IP, source port and destination port in time bin 1, we find the source IP address 241.119.171.133 used 8534 flows to scan No.1443 port of a large number of hosts. The same, we also find there was a port scan attack that scaned different ports of 88.15.139.82 in time bin 2. Note that the IP addresses are anonymized by our system.



(a) SrcIP (q = 1.5) (b) SrcIP (q = 0.2) (c) SrcPort (q = 1.5)

(d) SrcPort (q = 0.2) (e) Destination IP (q = 1.5) (f) DstIP (q = 0.2)

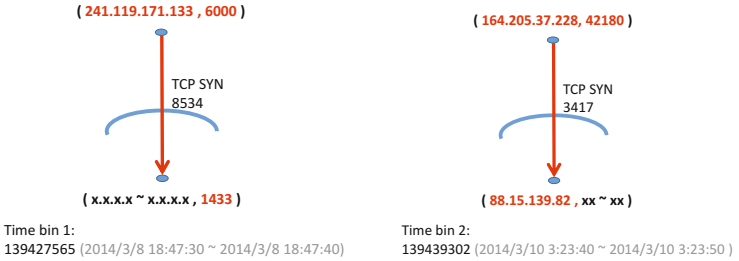
(g) DstPort (q = 1.5) (h) DstPort (q = 0.2) (i) Byte (q = 1.5)

Fig. 9. DTE-FP for Anomaly Detection

Anomaly Classification. We can classify the detection results by different flow features. As shown in TABLE 4, we find 13 kinds of entropy patterns when we detect anomalies by the thresholds of DTE-FP when $MFPR = 5\%$.

Table 3. Detection Capability

q	Anomaly Number & False Positive Number											
	MFPR=0%			MFPR=1%			MFPR=2%			MFPR=5%		
	LT	UT	**both**	LT	UT	**both**	LT	UT	**both**	LT	UT	**both**
1.5	659,0	4,0	**659,0**	721,0	4,0	**721,0**	756,0	6,0	**756,0**	841,0	23,0	**841,0**
1.1	299,0	18,0	**301,0**	348,0	18,0	**350,0**	387,0	20,0	**389,0**	445,0	23,0	**448,0**
0.8	81,0	74,1	**146,1**	87,0	74,1	**152,1**	94,0	75,1	**160,1**	102,0	75,1	**168,1**
0.2	0,0	415,0	**415,0**	0,0	423,0	**423,0**	0,0	425,0	**425,0**	0,0	445,0	**445,0**
-0.5	0,0	98,0	**98,0**	0,0	101,0	**101,0**	0,0	101,0	**101,0**	0,0	105,0	**105,0**
DTE	659,0	415,0	**949,0**	721,0	423,0	**957,0**	756,0	425,0	**1047,0**	841,0	445,0	**1148,0**



(a) port scan 1 (b) port scan 2

Fig. 10. Examples for Anomaly Validation

Table 4. Anomaly Classification

Feature					Anomaly Number
sicip	srcpt	dstip	dstpt	byte	DTE-FP
			√		474
		√			10
		√	√		52
	√				10
	√		√		22
	√	√	√		8
√					365
√			√		122
√		√			3
√		√	√		8
√	√				16
√	√		√		31
√	√	√	√		27
Total Anomaly Number					1148

6 Discussion

DTE-FP is only used to detect the flow-level traffic anomalies with a certain scale, such as DoS, DDoS, port scan, network scan, worm and spam. However, it doesn't care about other kinds of anomalies without flow-level feature deviation, e.g. virus and Trojan.

In this paper, in order to make comparisons between DTE-FP and Tsallis entropy, we use a constant threshold for entropy value to detect traffic anomalies. But DTE-FP is independent of detection algorithm. We can use other detection algorithms, such as a change-based algorithm for entropy, to detect anomalies.

7 Conclusion and Future Work

In this paper, we analyze the characteristics of Tsallis entropy for flow-level network traffic anomaly detection, and propose a new traffic anomaly detection method DTE-FP. Additionally, we implement a Hadoop-based system named TADOOP, which supports semi-automatic training, offline detection and online detection. finally, we deploy our system in Tsinghua University Campus Network, and use it to mine traffic anomalies. The experiment results reveal that DTE-FP performs much better than Tsallis entropy and TADOOP plays a good role in traffic anomaly detection. In our future work, we plan to use a change-based algorithm for entropy to detect traffic anomalies, and make comparisons between the two methods.

Acknowledgement. We thank for the support of Cisco-Tsinghua Joint Lab Research Project Funding and the helpful comments of our shepherd Mohan Dhawan. This work is also partially supported by the National High Technology Research and Development Program of China (863 Program) No. 2015AA016105, the National Natural Science Foundation of China (Grant No. 61202357, 61402253), and the Project for 2012 Next Generation Internet technology research and development, industrialization, and large scale commercial application of China (No. 2012 1763).

References

1. Lakhina, A., Crovella, M., Diot, C.: Mining anomalies using traffic feature distributions. In: Proceedings of the 2005 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM 2005), pp. 217–228. ACM, New York (2005)
2. Gu, Y., McCallum, A., Towsley, D.: Detecting anomalies in network traffic using maximum entropy estimation. In: Proceedings of the 5th ACM SIGCOMM Conference on Internet Measurement, IMC 2005, pp. 32–32. USENIX Association, Berkeley (2005)
3. Nychis, G., Sekar, V., Andersen, D.G., Kim, H., Zhang, H.: An empirical evaluation of entropy-based traffic anomaly detection. In: Proceedings of the 8th ACM SIGCOMM Conference on Internet Measurement, pp. 151–156. ACM (2008)

4. Tellenbach, B., Burkhart, M., Sornette, D., Maillart, T.: Beyond shannon: characterizing internet traffic with generalized entropy metrics. In: Moon, S.B., Teixeira, R., Uhlig, S. (eds.) PAM 2009. LNCS, vol. 5448, pp. 239–248. Springer, Heidelberg (2009)
5. Bereziński, P., Szpyrka, M., Jasiul, B., Mazur, M.: Network anomaly detection using parameterized entropy. In: Saeed, K., Snášel, V. (eds.) CISIM 2014. LNCS, vol. 8838, pp. 465–478. Springer, Heidelberg (2014)
6. Dean, J., Ghemawat, S.: Mapreduce: Simplified data processing on large clusters. Commun. ACM **51**(1), 107–113 (2008)
7. Apache hadoop (2014). http://hadoop.apache.org
8. Lee, Y., Lee, Y.: Toward scalable internet traffic measurement and analysis with hadoop. SIGCOMM Comput. Commun. Rev. **43**(1), 5–13 (2013)
9. Zhang, L., Wang, J., Lin, S.: Design of the network traffic anomaly detection system in cloud computing environment. In: 2012 International Symposium on Information Science and Engineering (ISISE), pp. 16–19. IEEE (2012)
10. Hodge, V.J., Jackson, T., Austin, J.: A hadoop-based framework for parallel and distributed feature selection (2013)
11. Bhuyan, M., Bhattacharyya, D., Kalita, J.: Network anomaly detection: Methods, systems and tools. IEEE Communications Surveys Tutorials **16**(1), 303–336 (2014)
12. Fontugne, R., Mazel, J., Fukuda, K.: Hashdoop: a mapreduce framework for network anomaly detection. In: 2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), pp. 494–499, April 2014
13. Ziviani, A., Gomes, A.T.A., Monsores, M., Rodrigues, P.: Network anomaly detection using nonextensive entropy. IEEE Communications Letters **11**(12), 1034–1036 (2007)
14. Wang, Z., Yang, J., Li, F.: An on-line anomaly detection method based on a new stationary metric-entropy-ratio. In: 2014 IEEE 13th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), pp. 90–97. IEEE (2014)
15. Tsallis, C.: Possible generalization of boltzmann-gibbs statistics. Journal of Statistical Physics **52**(1–2), 479–487 (1988)
16. Tsallis, C.: Nonextensive statistics: theoretical, experimental and computational evidences and connections. Brazilian Journal of Physics **29**(1), 1–35 (1999)
17. Tsallis, C.: Entropic nonextensivity: a possible measure of complexity. Chaos, Solitons & Fractals **13**(3), 371–391 (2002)
18. IPFIX library (2014). http://libipfix.sourceforge.net/
19. Tian, G., Wang, Z., Yin, X., Li, Z., Shi, X., Lu, Z., Zhou, C., Yu, Y., Guo, Y.: Mining network traffic anomaly based on adjustable piecewise entropy. In: IEEE/ACM International Symposium on Quality of Service (IWQoS), June 2015