Online Detection of Concurrent Prefix Hijacks

Shen Su[1(✉)], Beichuan Zhang[2], and Binxing Fang[1]

[1] Harbin Institute of Technology, Harbin, China
johnsuhit@gmail.com, bxfang@pact518.hit.edu.cn
[2] The University of Arizona, Tucson, USA
bzhang@cs.arizona.edu

Abstract. Prefix hijacking is a major security threat to the global Internet routing system. Concurrent prefix hijack detection has been proven to be an effective method to defend routing security. However, the existing concurrent prefix hijack detection scheme considers no prefix ownership changes, and online concurrent prefix hijack detection endures seriously false positive. In this paper, we study the possible characters to filter out false positive events generated online by machine learning, and apply such characters in the online detection. Our result shows that our refined online concurrent prefix hijack detection can detect all offline detected events with no false positive. We also confirm that (1) neighboring ASes seldom hijack each other's prefixes; (2) large ISPs seldom suffer from prefix hijacks or conduct hijacks.

Keywords: Prefix hijack · False positive · Online detection

1 Introduction

Internet is composed of tens of thousands of ASes (Autonomous Systems), which uses BGP (Border Gateway Protocol) to exchange the routing information towards prefixes. Because BGP doesn't consider security, no authentication is required when exchanging routing information. As a result, an AS is able to announce any prefix without authentication (called prefix hijacks), and broadcast it to the rest of the world. Nowadays, prefix hijacking has been the most popular cyber attacks, and widely applied in man in the middle (MITM), phishing scams, and DDOS attacks towards SpamHaus and cloudflare. The traffic to victim prefixes are redirected to attacking networks by such attacks. Attackers may blackhole the victim prefix, impersonate the victim prefix to communicate other entities, or conduct MITM attacks.

To enable authentication, existing proposals [5,6,9,10] require to change BGP, i.e. they require to change existing network configurations and operations which are hard to deploy. Other proposals devote to detect prefix hijacking [3,7,8,11]. Such works collect routing messages and compare them with prefix ownership. The prefix ownership is known as priori [2,7] or inferred from collected routing information [4,8]. However, an up-to-date and complete priori prefix ownership need collaborative works, which is hard to deploy. Approaches

J. Tian et al. (Eds.): SecureComm 2014, Part II, LNICST 153, pp. 69–83, 2015.
DOI: 10.1007/978-3-319-23802-9_8

based on ownership inference also suffer seriously from false positive alarms because prefix hijacks and some legitimate operational practises have similar behaviors. Our previous work [1] develops a off-line scheme that detects concurrent prefix hijacks which greatly reduce the risk of false positive alarms. In this paper, we propose an improvement on their detection algorithm in online scenarios.

Our previous concurrent prefix hijack detecting scheme relies on the lifetime of prefix announcement to infer prefix ownership. When an AS is observed announcing a prefix for more than one day, it is inferred as an owner. However, in online environment, lifetime cannot reveal prefix ownership changes immediately, e.g. when an AS is a new owner of a prefix, its announcements' lifetime is short, and our previous scheme would treat its announcements as prefix hijacks. Consequently, our previous scheme endures more false positive in online environment.

Towards minimizing the false positive generated online, we analyse the potential characters of false positive alarms. Our idea is based on the assumption that offenders always tend to hijack prefixes effectively at little cost. In cases that the prefix owner can easily detect the routing announcement of the offender, or can easily tackle with the hijack, the prefix hijack is probably a false positive. In practise, we focus on the offender, the offending target (i.e. prefix owner), and the distance between them. As a offender, a transit provider takes risks to hurt its business interest once its customers realize the hijack; large ISPs (Internet service provider) invest more resource into the network security than stub ASes, hijacks towards their prefixes turn out to be bad ideas; hijacking neighboring ASes's prefixes can also be detected and tackled with easily. We look into the offline detection and false positive generated online. Our analysis shows that the prefix hijacks between neighboring ASes and large ISPs seldom occur, but a non-trivial number of false positives fall into the above two cases. Applying the two characters in the online detection, our result shows that we manage to filter out all false positive events. At the same time, our online detection detect the same set of prefix hijack events as the offline scheme.

In the rest of this paper, we discuss related works in Sect. 2. In Sect. 3 we introduce concurrent prefix hijacks detecting scheme and its limitations. In Sect. 4 we introduce our online detection scheme. In Sect. 5, we evaluate our scheme by experimental results and conclude in Sect. 6.

2 Related Works

A number of solutions have been proposed to eliminate the problem of false routing announcements. Such works can be categorized into two broad categories: prevention [5,6,9,10] and detection [3,7,8,11].

The prevention techniques attempt to prevent ASes from announcing false routes. Many prevention proposals [6,9] are difficult to deploy, because they require extensive cryptographic key distribution infrastructure, and/or a trusted central database. PGBGP and QBGP monitor the origin AS for each prefix

according to BGP updates, and a router avoids using new routes if the old route is still available. PGBGP focuses on minimizing few false negatives, however it ends up with many false positives which causes an increase in the time to adopt legitimate new routes. Instead, concurrent prefix hijack detection may have false negatives but zero false positives allows traffic source networks to automate their responses to prefix hijack events.

The detection techniques attempt to identify prefix hijack events through monitoring the routing system, including control plane and data plane information. Such techniques can be categorized as: (a) Traceroute based solutions and (b) Control-plane based solutions. These detection solutions require no change to BGP protocol and thus are more deployable. However it is important to note that existing detection systems are geared towards protecting individual prefix owners, i.e. safeguarding the allocated prefix block of a network against any on-going prefix hijacks. Whereas the traffic source networks needs to protect their entire routing table from any on-going prefix hijacking attacks in order to safeguard all of their data traffic. Therefore each existing detection system poses its own practical limitations in safeguarding an entire routing table for traffic source network, thereby making them ineffective.

Traceroute based solutions protect their prefixes by periodically probing data paths to the protected prefixes, such as iSPY [11] and Lightweight Probing [12]. Such solutions are good to be used when the quantity of prefixes to be protected is small. However, for traffic source networks, the protection list is too long to utilize such traceroute based solutions.

Control-plane-based solutions [8], monitor the entire routing table passively according to BGP data. However, due to limited vantage point locations and legitimate reasons for anomalous updates [4], the results include too many false positives as well as false negatives. Certain control-plane-based solutions, such as PHAS [7] and MyASN [2], use information prefix ownership information to filter out false positives, but then their effectiveness is limited by the number of participating prefix owners. Furthermore, certain solutions, such as [4], combine anomaly detection of control plane information with data-plane fingerprints to perform joint analysis, but the detection accuracy is still limited by the vantage points locations of both data sources. With the high false positives produced by existing control-plane-based solutions, traffic source network could suffer from erroneously dropping correct route updates and thus impacting Internet connectivity. In contrast to existing control plane based systems, concurrent prefix hijack detection correlates suspicious routing announcements along the time dimension and thus minimizes false positives, enabling automated response to prefix hijack attacks without requiring human intervention from traffic source networks.

3 Offline Detection Scheme and Its Limitation

In this section, we first briefly introduce our original detection scheme, then discuss its limitations and possible problems we may encounter when we apply it online.

3.1 Offline Prefix Hijack Detecting Scheme

Our previous concurrent prefix hijack detecting scheme relies on BGP routing data to infer prefix ownership. The inference is based on prefix's announcement lifetime. Usually the owner AS of a prefix is expected to announce the prefix persistently for a long duration. In our scheme, we associate every prefix with a stable set and a related set containing ASes that probably can legitimately announce the prefix. **Stable Sets** captures ASes that are likely owners of a prefix. In practise, any AS announcing a prefix cumulatively for one day or more within a year is included in the prefix's stable set. **Related Sets** captures ASes that are not the owner of the prefix but can legitimately announce it in operation. We have found the following four cases useful for our detection algorithm.

First, an AS in a prefix's stable set also belongs to related set of all its sub-prefixes.

Second, for all ASes in a prefix's stable set, their direct provider ASes also belong to this prefix's related set. For this purpose we use a simple heuristic to identify stable provider-customer inter-AS links. We start with a list of well-known tier-1 ASes, and given an AS path, the link from a tier-1 AS to a non-tier1 AS is provider-customer, and any link after that is also provider-customer due to the commonly deployed No-Valley policy. This can be considered as a subroutine in most of the existing AS relationship inference algorithms, and thus the accuracy in inferring provider-customer relationship should be similar, although we do not need to infer peer-peer or sibling-sibling relationship, which is the challenging part of general AS relationship inference.

Third, ASes participating in an Internet Exchange Point (IXP) can legitimately announce the IXP's prefixes, and similarly the IXP AS can also legitimately announce the prefixes of its participating ASes.

Fourth, ASes belonging to the same organization are related and can legitimately announce each others prefixes. We simply infer such relation from the domain name of the contact emails listed in the WHOIS [13] database.

Any AS not belonging to a prefix's stable set or related set but originating the prefix is deemed to be an offending AS, attempting to potentially hijack the prefix. In such case, we also say that the AS is offending the prefix's stable set, which represents the owner of the prefix. For an offending AS, we defense its offense value as the number of unique ASes that this AS is offending at any given moment. The offense value captures how many other networks are being potentially hijacked simultaneously. Based on the filtered global view of origin changes, we compute offense value for every AS for the entire year.

Algorithm 1 summarizes the above steps. It uses one year of archived BGP tables and updates, available at Route Views Oregon monitors, to construct stable and related sets. Thereafter every BGP routing announcement is checked whether it is suspicious or legitimate by checking origin AS against stable and related set of prefix. Anytime the offense value of an AS exceeds the threshold of 10, it is reported to be a concurrent hijack.

Algorithm 1. Offline prefix hijack detection scheme.

Input:

StableSets(p): stable set of prefix p;
RelatedSets(p): related set of prefix p;

```
1: FOR all BGP routing messages
2:     IF AS X announces prefix p at time t
3:         IF AS X ∉ StableSets(p) or RelatedSets(p)
4:             Update AS X's offense value by StableSet(p);
5:     ELSIF AS X withdraws prefix p at time t
6:         IF AS X ∉ StableSet(p) or RelatedSet(p)
7:             Reduce AS X's offense value by StableSet(p);
8:     Report prefix hijack event: if AS X's offense value >= 10
```

3.2 Limitations

[1] has proved our offline scheme can safely detect prefix hijack events with zero-false positive. However, we still face a few limitations when we apply it in online scenario. And such limitation may impact the detection accuracy.

First of all, to detect prefix hijack events of a year, the offline scheme requires to calculate stable and related sets of each prefix from the BGP routing messages of the entire year. While, in online scenarios, we have no access to the BGP messages which are generated after current time.

Second, the offline detection scheme considers no dynamical factors when inferring stable and related sets. However, prefix ownership, AS topology, and other dynamical factors change over time. So prefix announcement lifetime may mistakenly reflect prefix ownership, especially in online scenarios. Actually, ASes with a short announcement lifetime may legitimately announce a prefix when the above dynamical factors happens. When we observe announcements from such a legitimate announcer, our detecting scheme may report it as a prefix hijack announcement, which is a de facto false positive. For instance, AS A is an owner of prefix p, and we can persistently observe its announcement. So we take the announcement of AS A for granted. At time t, AS B becomes another owner of prefix p. However, we can not infer AS B as an owner until one day after t according to the lifetime. And during that period, we get a false positive.

The above two limitations make us lack the knowledge of whether our detection is suffering from Internet dynamics, i.e. when we observe an announcement originated from an inexperienced AS online, we cannot tell whether this is a prefix hijack or a legitimate announcement because we have no idea of what BGP messages are to be announced.

4 Online Prefix Hijack Detection Scheme

Based on the offline detection scheme, we now design the online detection scheme. Our principle is to minimize the false negative with assurance of no false positive.

To consider dynamical factors, we perform the online detection scheme over a moving observation window $[t - T, t)$ worth of BGP routing messages where

Algorithm 2. Online prefix hijack detection scheme.

Input:

t: current time, $t_0 = t$: time to update Stable and Related sets;
N: the interval to refresh the observation window;
T: size of observation window;

```
1: Initialize the observation window;
2: FOR all BGP routing messages in window (t − T, t)
3:     Track duration of prefix-origin AS and AS relation;
4: Initialize Stable and Related Sets of every prefix;
5: For all online observed BGP routing messages
6:     Put this message into the observation window;
7:     t=time stamp of this BGP message;
8:     IF AS X announces prefix p at time t
9:         IF AS X ∉ StableSets(p) or RelatedSets(p)
10:            Update AS X's offense value by StableSet(p);
11:    ELSIF AS X withdraws prefix p at time t
12:        IF AS X ∉ StableSet(p) or RelatedSet(p)
13:            Reduce AS X's offense value by StableSet(p);
14:    Report prefix hijack event: if AS X's offense value >= 10
15:    IF t >= t_0 + N
16:        Update duration of prefix-origin and AS relation;
17:        Update Stable and Related Sets of every prefix;
18:        t_0 = t;
```

t is the current time and T is the size of observation window. For the offline detection scheme, one year worth of training data is used to construct the initial stable and related sets for each prefix. Considering the stable and related sets for every prefix can not remain static, we need to dynamically update the stable and related sets with the movement of observation window. The duration of every prefix-origin AS pair and duration of every AS relation pair is updated by tracking the announcement and withdrawal BGP routing messages. With a certain frequency, aforementioned announcement durations are re-evaluated and the stable and related sets of every prefix are updated as shown in Algorithm 2.

Following the offline detection scheme described in Algorithm 1, we check each in-coming BGP routing message online to detect prefix hijack events. Since no future can be observed from the observation window, we require prediction in our online detection. Our idea is to look for characters of prefix hijack events to label false positive out. In the rest of this section, we first discuss the size and recalculating frequency of the observation window. Then we evaluate the seriousness of caused false positive online. Next, we discuss the strategies to filter out such false positives. Finally, we refine the above detection scheme.

4.1 Configuration

We first discuss the size of the observation window. Generally, more considered routing information generates more prefix ownership. However, prefix ownership

changes over time, thus our inference may includes outdated prefix ownership. So a bigger observation window in size generates more prefix ownership including more outdated prefix ownership. More prefix ownership helps us to detect more de facto prefix hijacks, while we take more risk with more outdated prefix ownership. Consequently, the size of the observation window is a tradeoff between false positive and false negative.

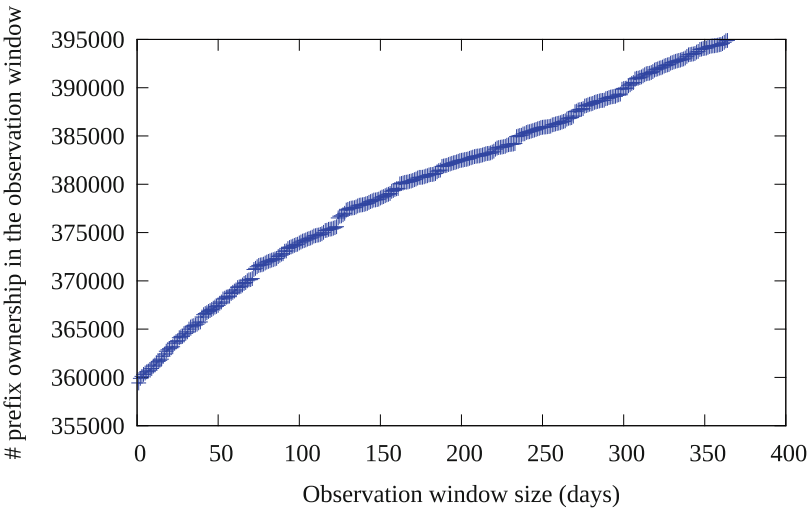

Fig. 1. # prefix ownership over time.

In Fig. 1, we show the total quantity of prefix ownership inferred from the routing information in the observation window during the year of 2011. Considering that the routing table itself is growing in size, we filter out the prefixes which cannot be observed according to the routing data before 2011 (i.e. new prefixes). The x-ray represents the size of the observation window, the y-ray represents corresponding quantity of prefix ownership inferred from the observation window. We notice that even with one-day's routing data, we can infer almost 360,000 (90 % of all) prefixes's ownership. The total quantity goes up to 400,000 linearly with a growth rate around 100 prefixes per day.

We compare the prefix ownership inferred from routing data of each day, and show the cumulated prefix ownership changes over time in Fig. 2. The x-ray represents the observing duration, and the y-ray represents the accumulated quantity of prefixes with a ownership change. We observe a linear growth rate (20–30 prefixes per day), and totally there are less than 10,000 (2.5 % of all) changing ownership prefixes.

To our surprise, neighbor prefix ownership quantity nor prefix ownership changing rates turn out to be a bottle neck factor to the observation window size. With a observation window in size of one day or one year, we have prefix ownership for almost the entire routing table and not too many prefixes

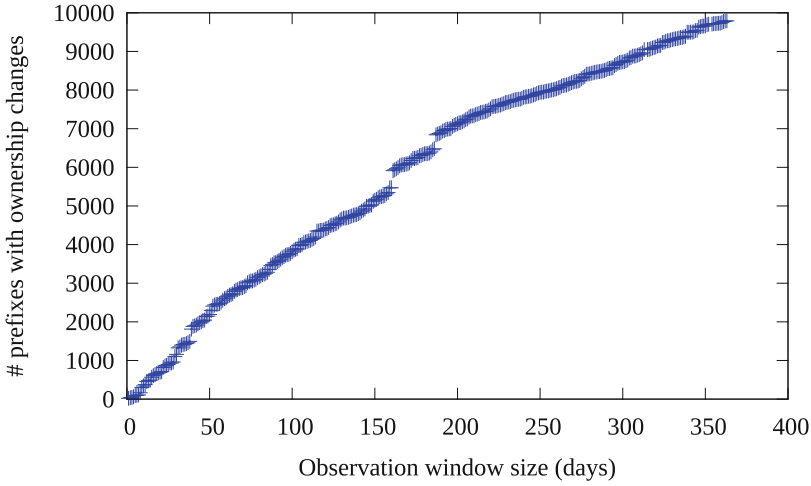

Fig. 2. # prefixes with ownership changing over time.

experiencing ownership changes. Considering our principle (minimizing the false negative with assurance of no false positive) and the tradeoff between false positive and false negative, we decide to set the size as one year. This is because we still need to involve prediction in the detection which is dedicated to filter out false positive alarms. And we want to maximize the value of our detection scheme.

For the frequency to update the stable and related sets, it is a tradeoff between computation cost and outdated information caused detection inaccuracy including both false positive and false negative. A intensive update schedule (update every hour) is unnecessary because there are only around 20–30 prefix ownership changes every day. Our program is written in perl and takes 2–5 minutes to conduct an update on one-year Oregon data. So we decide to update the stable and related sets every day.

4.2 False Positives

Given the outline of the detection scheme, our problem now is to eliminate the false positives generated online. To that end, we begin with depicting the false positive. In Fig. 3, we compare the quantity of prefix hijack instances detected by offline and online detection scheme (following Algorithm 1). We refer to a prefix hijack instance as an offending case when we observe an AS is announcing other ASes' prefix, noted as a triple (T, A, B). T refers to the time when the prefix hijack instance happens, A refers to the offending AS who conducts the prefix hijack instance, and B refers to the origin AS who owns the prefix. Curve "offline" represents the prefix hijack instances observed offline, and Curve "false positive online" represents the online prefix hijack instances which are not observed offline.

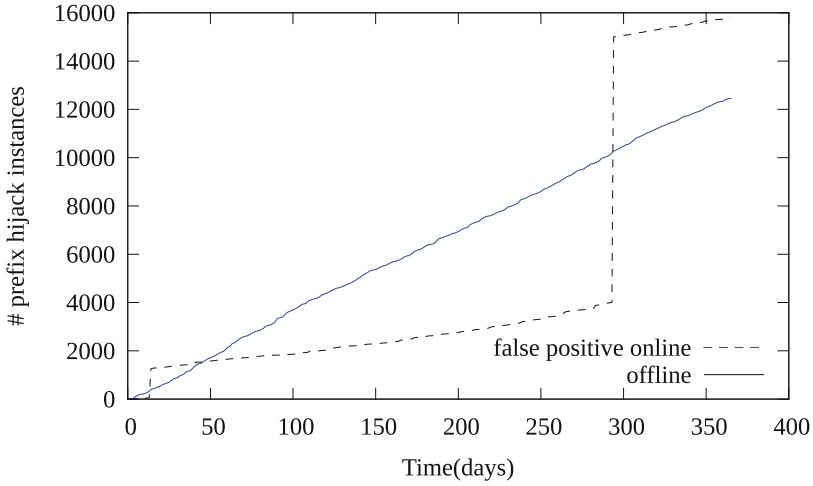

Fig. 3. # prefix hijack instances.

As observed in Fig. 3, online generated false positive are even more than prefix hijack instances detected offline (besides the prefix hijack events happened in Jan and in Oct). So online detected false positive is seriously impacting the accuracy of our online prefix hijack detection. We also notice the linear growth rate of the false positive which is consistent to the linear growth of the prefix ownership changes.

4.3 Increase Threshold

One simple idea comes to us is to increase the threshold. If the offense value of a prefix hijack event is T offline, and T' online. The threshold should be increased by $T' - T$. Following the detection scheme described above, we change the threshold and compare the generated reports with the one generated offline for the year of 2011 on Oregon data [14] (show in Table 1).

In the offline detection, our scheme detects 12 prefix hijack events. When the threshold is 10, we get 12 false positive events. With the threshold going up, the quantity of false positive events gets smaller, but the detection misses more de facto prefix hijack events. When the threshold is 67, we clear all false positive events, but there is only 3 prefix hijack events left in the detection. Consequently, simply increasing threshold can hardly solve the false positive problem. Since the increasement varies among prefix hijack events, if the increasement is small, there are still lots of false positive events; if the increasement is big, the offline detected prefix hijack events are also filtered out.

4.4 Characters of False Positive Instances

Our further strategies to predict false positive is: analysing possible characters which can be used to discriminate false positive instances and de facto prefix

Table 1. Results of online detection with different threshold.

Threshold	# online detected events	# false positive events	# false negative events
10	24	12	0
11	21	10	1
12	19	8	1
13	16	6	2
14	15	5	2
17	12	4	4
19	10	2	4
21	9	1	4
24	8	1	5
28	7	1	6
67	3	0	9

hijack instances (as described in Sect. 4.2). Such characters require to satisfy one of the following conditions: (1) no de facto prefix hijack instance has this character, and some false positive instances have this character; (2) all de facto prefix hijack instances have this character, and some prefix hijack instances don't have this character. For the former, we ignore instances with such characters; for the latter, we ignore without such characters. Since it is a hard job to find a character that all de facto prefix hijack instances have, in this paper, we focus on the characters satisfy the former condition. We analyse characters by first assuming possible characters, then validating with actual BGP data from Oregon in 2011. Totally, we get 12457 false positive instances, and 15786 de facto prefix hijack instances.

Neighboring ASes. Since the information we collect to infer related sets is incomplete, the de facto related set is bigger than the inferred one. Our scheme may takes an AS in the de facto related set as an offender, and cause a false positive. Considering that most ASes in the related set are neighboring ASes of the prefix owner, we assume "the offender and the prefix ownership are neighboring ASes" as a character of the false positive instance.

In practise, among all the false positive instances during 2011, there are 1046 (8.4 % of all) false positive instances which have this character. Among the de facto prefix hijack instances, there are only 43 (0.27 % of all) instances which have this character. Consequently, the character "the offender and the prefix ownership are neighboring ASes" satisfy our requirement to label the false positive instances.

Large ISP. A number of ASes lie in the central of Internet, and provide transit service for a lot of stub ASes. Comparing with stub ASes, such ASes controlled by large ISPs invest much more manpower into their network's routing security. Consequently, prefix hijack towards such ASes are more likely detected and

tackled with. In the meanwhile, if a large ISP conduct a prefix hijack to other ASes's prefix, its business interest will be badly harmed once its customer ASes or peering ASes realize the hijack. Thus we assume "the offender or the prefix owner belongs to a large ISP" as a character of the false positive instances.

According to the common understanding that the number of neighboring ASes of an AS can reflect its scale, in this paper, we determine the scale of an AS's belonged ISP based on its quantity of neighboring ASes. In Fig. 4, we show the CDF of offended ASes' (i.e. prefix owners') scale. Curve "false positive" represents false positive instances, and curve "offline" represents de facto prefix hijack instances. We observe that the scales of offended ASes in false positive instances are obviously bigger than the ones in de facto prefix hijack instances. Only 3.2 % of all de facto instances's offended AS are in a scale of more than 100 neighboring ASes. For false positive instances, 17 % of all instances's offended ASes fall into that scale. As a result, we derive a more specific character "the prefix owner's neighboring ASes are more than 100" to label false positive instances.

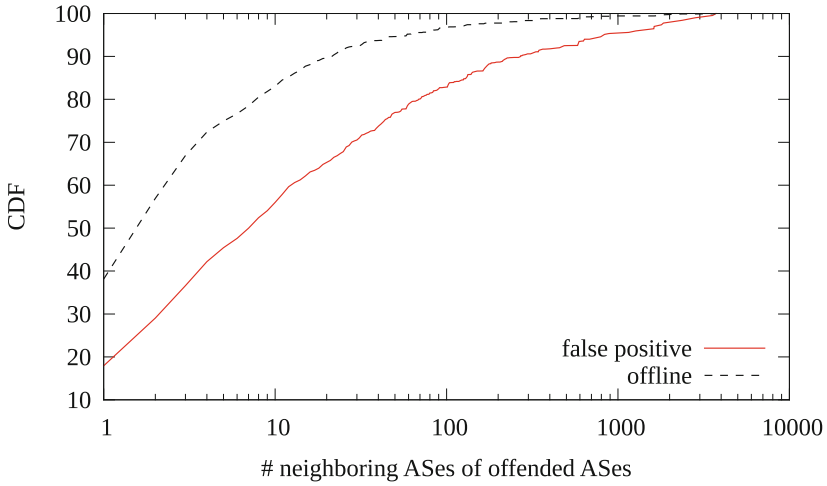

Fig. 4. CDF of offended ASes' scale.

Similarly, in Fig. 5, we show the CDF of offending ASes' scale. As observed, the scales of offending ASes in false positive instances are obviously bigger than that of prefix hijack instances. Only 8.5 % of all de facto instances's offending AS are in a scale of more than 40 neighboring ASes. For false positive instances, 51.2 % of all instances's offending ASes fall into that scale. As a result, we derive a character "the offending AS's neighboring ASes are more than 40" to label false positive instances.

As shown in Algorithm 3, we refine our detection scheme. For each in-coming BGP routing message online, we check if the instance has the two characters before updating the offense value. If it has, we ignore that instance.

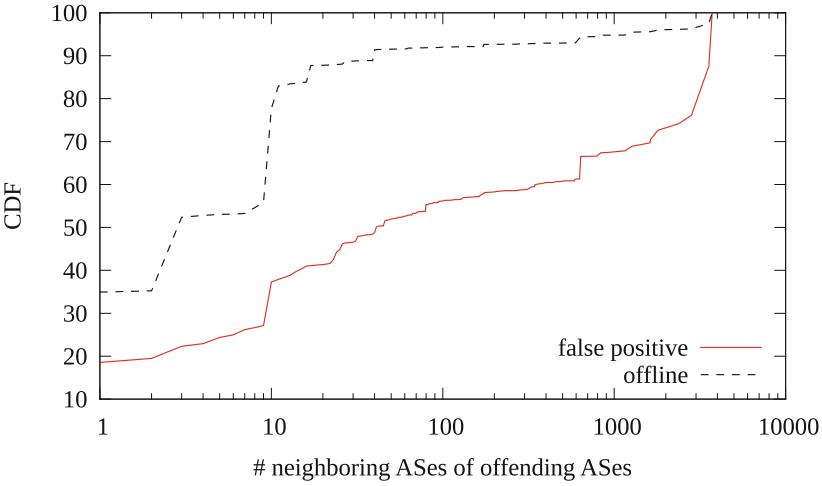

Fig. 5. CDF of offending ASes' scale.

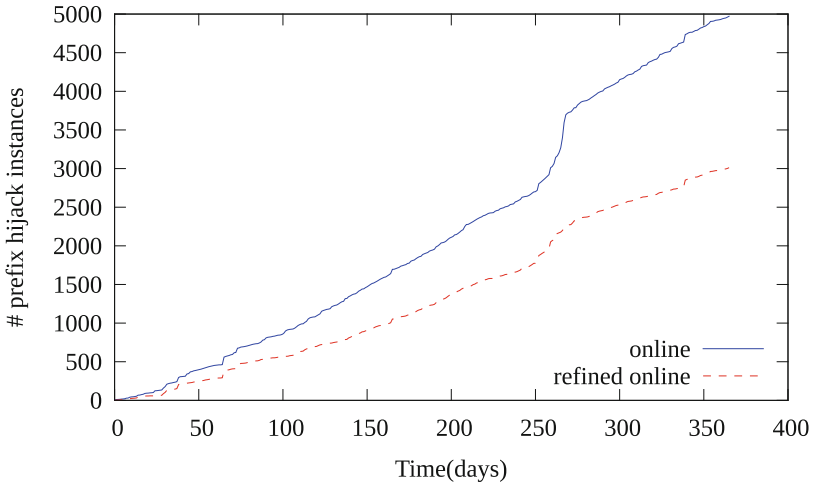

Fig. 6. De facto prefix hijack instances online.

5 Results

In this section, we evaluate our online detection scheme. Since we study the configuration and characters of false positive instances according to the Oregon data in 2011, to prove that our scheme is independent from the data set measured, we use Oregon data in 2012 in this section.

In Fig. 6, we show the CDF of de facto prefix hijack instances detected online. Curve "online" represents de facto prefix hijack instances detected before applying false positive characters, and curve "refined online" represents de facto prefix

Algorithm 3. Online prefix hijack detection scheme.

Input:

t: current time, $t_0 = t$: time to update Stable and Related sets;
N: the interval to refresh the observation window;
T: size of observation window;

Initialize the observation window;
FOR all BGP routing messages in window $(t - T, t)$
    Track duration of prefix-origin AS and AS relation;
Initialize Stable and Related Sets of every prefix;
For all online observed BGP routing messages
    Put this message into the observation window;
    t=time stamp of this BGP message;
    IF AS X announces prefix p at time t
        IF AS X is a neighboring AS of $StableSets(p)$
            next;
        ESLIF AS X has more than 40 neighboring ASes or an AS in $StableSets(p)$ has more than 100 neighboring ASes
            next;
        ESLIF AS $X \notin StableSets(p)$ or $RelatedSets(p)$
            Update AS X's offense value by $StableSet(p)$;
    ELSIF AS X withdraws prefix p at time t
        IF AS X is a neighboring AS of $StableSets(p)$
            next;
        ESLIF AS X has more than 40 neighboring ASes or an AS in $StableSets(p)$ has more than 100 neighboring ASes
            next;
        ESLIF AS $X \notin StableSet(p)$ or $RelatedSet(p)$
            Reduce AS X's offense value by $StableSet(p)$;
    Report prefix hijack event: if AS X's offense value >= 10
    IF $t >= t_0 + N$
        Update duration of prefix-origin and AS relation;
        Update Stable and Related Sets of every prefix;
        $t_0 = t$;

hijack instances detected after applying false positive characters. We notice that about 40 % of the prefix hijack instances are filtered out after we refine the detection scheme. However, we also notice that the two curves share a similar shape. Their cliff points are at almost the same set of time points, i.e. both curves capture the same prefix hijack events. Consequently, our refined detection scheme can capture the de facto events at the cost of offense value decrease.

In Fig. 7, we show the CDF of false positive prefix hijack instances detected online. Curve "false positive online" represents false positive instances detected before applying false positive characters, and curve "false positive refined" represents false positive instances detected after applying false positive characters. Our refined scheme labels out 80 % of the false positive instances. Moreover, our refined scheme does not have obviously cliff points which may induce false positive events.

Actually, during the year of 2012, the offline detection scheme detects 11 prefix hijack events. The original online detection scheme detect 46 prefix hijack

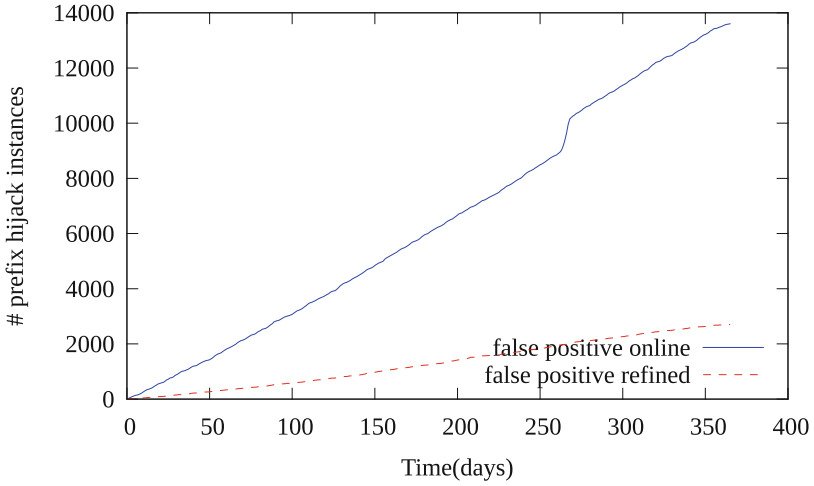

Fig. 7. False positive instances online.

events, in which 35 of them are false positive events. While the refined online detection scheme detects 11 prefix hijack events during 2012, which matches perfectly with the offline detection.

6 Conclusion

In this paper, we propose an online concurrent prefix hijack detection scheme. By analysing characters of false positive instances, we manage to filter out false positive events generated online. In the meanwhile, our detection scheme captures all the prefix hijack events offline.

In the process of studying suitable characters for filtering out false positive instances, we also learn that attackers seldom hijack neighboring ASes' prefixes. Attackers also seldom hijack large ISP's prefix, and vice versa.

Acknowledgement. This research was partially supported by the National Basic Research Program of China (973 Program) under grant No. 2011CB302605, the National High Technolgy Research and Development Program of China (863 Program) under grants No. 2011AA010705 and No. 2012AA012506, China Internet Network Information Center (CNNIC) under grants No. K201211043, the National Key Technology R&D Program of China under grant No. 2012BAH37B00, the National Science Foundation of China (NSF) under grants No. 61173145 and No. 61202457.

References

1. Varun, K., Qing, J., Zhang, B.: Concurrent prefix hijacks: occurrence and impacts. In: IMC (2012)
2. RIPE myASN System. http://www.ris.ripe.net/myasn

3. Chi, Y.-J., Oliveiro, R., Zhang, L.: Cyclops: the AS level connectivity observatory. SIGCOMM Comput. Commun. Rev. **38**(5), 5–16 (2008)
4. Hu, X., Mao, Z. M.: Accurate real-time identification of IP prefix hijacking. In: IEEE Symposium on Security and Privacy (2007)
5. Karlin, J., Forrest, S., Rexford, J.: Pretty Good BGP: improving BGP by cautiously adopting routes. In: ICNP (2006)
6. Kent, S., Lynn, C., Mikkelson, J., Seo, K.: Secure border gateway protocol (S-BGP). IEEE JSAC **18**, 103–116 (2000)
7. Lad, M., Massey, D., Pei, D., Wu, Y., Zhang, B., Zhang, L.: PHAS: a prefix hijack alert system. In: USENIX Security Symposium (2006)
8. Qiu, J., Gao, L., Ranjan, S., Nucci, A.: Detecting bogus BGP route information: going beyond prefix hijacking. In: SecureComm (2007)
9. Subramanian, L., Roth, V., Stoica, I., Shenker, S., Katz, R. H.: Listen and whisper: security mechanisms for BGP. In: NSDI (2004)
10. Zhang, M., Liu, B., Zhang, B.: Safeguarding data delivery by decoupling path propagation and adoption. In: INFO-COM (2010)
11. Zhang, Z., Zhang, Y., Hu, Y. C., Mao, Z. M., Bush, R.: iSPY: detecting IP prefix hijacking on my own. In: SIG-COMM, pp. 327–338 (2008)
12. Zheng, C., Ji, L., Pei, D., Wang, J., Francis, P.: A light-weight distributed scheme for detecting IP prefix hijacks in real-time. In: ACM SIGCOMM (2007)
13. Whois Database. http://www.whois.net/
14. University of Oregon Route Views Archive Project. http://www.routeview.org