Detecting BGP Route Anomalies with Deep Learning

Kyle McGlynn, H. B. Acharya, Minseok Kwon
Rochester Institute of Technology
kjm9969@rit.edu, acharya@mail.rit.edu, jmk@cs.rit.edu

***Abstract*—Fake or mistaken BGP updates can cause serious damage to Internet routing. We note that an expert network administrator can develop a "gut feeling" that lets them identify mistaken or attack updates. Might it be possible to capture such intuition in a learning-based anomaly detection mechanism? Our idea is that good route updates share characteristics which bad updates do not, and even if such characteristics are subtle (i.e. cannot be captured in clear rules), they can be learned. More specifically, an auto-encoder (trained on known-good BGP routing data) will successfully encode good route updates, but will perform more poorly with random or malicious updates.**

In our system, we use two auto-encoders, each trained to detect a specific type of anomalous BGP update. If either auto-encoder performs poorly (i.e. shows large differences between input and output), we report the BGP update as likely-anomalous. Our detector shows promising early results in identifying anomalous MOAS conflicts as well as prefix hijack attacks.

I. Introduction

The Internet consists of thousands of Autonomous Systems, which advertise routes to each other using BGP. However, such route updates are not always correct or trustworthy. Bad updates, as seen in anomalous Multiple-Origin AS (MOAS) conflicts and prefix hijacks, have been used in black holing, denial of service (1), and phishing (2) attacks.

Hu *et al.*(3) outline four different types of anomalous BGP updates. In general, an AS may originate fake advertisements (claim to host an IP it does not have), fake routes (short paths to another AS), or add or delete ASes from real routes (to make them seem longer or shorter). There do exist mechanisms to detect such updates, such as iSPY(4), which uses regular probes to test how an AS connects to the rest of the network. However, as such tools are reactive in nature, they are slow to detect an attack.

In this context, we note that machine learning has been demonstrated as a tool to identify good network routes (5) . The authors extract features from long-lived routes, then pick out the good routes using boosted classification trees. We are thus motivated to ask if such an approach can be extended to detect not only misconfigurations, but also malicious attacks.

In this project, we apply a deep learning algorithm – specifically, an auto-encoder – to the problem of detecting anomalous MOAS conflicts as well as prefix hijacking attacks.

An auto-encoder learns the *essential features* of its input: its task is to re-create the input at the output, while only passing a minimal amount of information through its small hidden layer. By training auto-encoders on legitimate data, we ensure that they are good at reconstructing similar (i.e. legitimate) data, while performing poorly on other (i.e. anomalous) data. We note that our approach is not only the first deep learning approach to detecting BGP attacks, but also *locally deployable*, as it can be deployed on single routers.

II. Approach

Our approach detects anomalous BGP updates using *auto-encoders* (6), generative neural networks with encoding and decoding layers. In an auto-encoder, the encoding layers compress the input data, and the decoding layers decompress it. If the output is a successful reconstruction of the input, then the data passed through the small hidden layer (which is much smaller than the input and output) is sufficient to capture the input. In other words, an auto-encoder learns the *essential features* of input data.

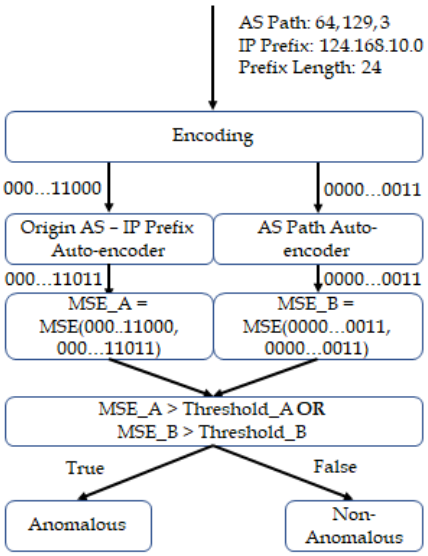


Fig. 1: Work flow of our detection mechanism.

Our idea is to train auto-encoders on clean data; the auto-encoder trains itself to successfully reproduce clean data, but not attack data (which does not share the same essential features). We flag anomalies for cases where the auto-encoder fails to reconstruct the input well. This approach, unlike previous techniques(5), *does not need labeled attack data*.

We use two auto-encoders, with standard activation (ReLU), optimization (Nadam), and loss (MSE) functions. The first auto-encoder takes as input, (legitimate) origin ASes, the IP prefixes, and the length of the IP prefixes, in order to learn which origin ASes match which IP prefixes. The second takes AS paths, to learn which ASes are connected together.

The second auto-encoder (for AS paths) requires data cleaning: we prune consecutive occurrences of an AS number to one instance.[1] As auto-encoders need a fixed input size, we also pad the AS path to a fixed length of 50. (We chose this length as a practical upper bound, as the AS graph is densely connected and we never saw paths of 50 or more hops.)

[1]BGP allows ASes to append their own AS number to an AS path any number of times. This is sometimes abused by bad actors, who use such padding to make paths long and thus unattractive.

978-1-7281-1878-9/19/$31.00 ©2019 IEEE

Our workflow is illustrated in Figure 1. Features extracted from an incoming BGP update message are encoded and passed to the trained auto-encoders. If the error of either auto-encoder (i.e. the difference between its output and input) exceeds a margin, we flag the update message as anomalous.

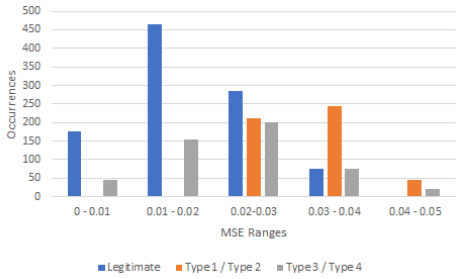
III. Evaluation

We implement our auto-encoder designs in Tensorflow (7), using the Keras API. The training dataset is the BGP data seen by one BGP speaker in one day, collected from the University of Oregon's Route Views Project (8). Unfortunately, we were not able to find collections of anomalous BGP announcements to test against. Hence, we crafted our own attack data, by editing random updates to construct four different types of anomalous BGP route advertisement[2](3). Our adversarial updates are constructed to ensure that their IP addresses are not associated with the AS path origin, and their origin AS is not a neighbor of the AS path origin.

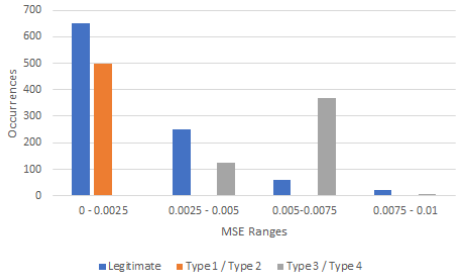
In Experiment 1, we modify 1,000 randomly-chosen BGP updates. Figure 2a shows that at a threshold of 0.02, the origin AS - IP prefix auto-encoder can identify all the fake updates of types one and two (origin AS - IP prefix mismatch), and 60% of types three and four (AS path modification), for a total f-score of 0.82. Types three and four can be captured using a threshold of 0.0039 with the AS path auto-encoder as seen in Figure 2b; the f-score is 0.83.

In Experiment 2, we choose and modify 40,000 BGP updates. Performance is not as good: with 40,000 bad updates, it is harder to find a suitable threshold, and the f-score also drops. In Figure 2c, nearly half the attack updates of types one and two lie in the range 0.0 - 0.01; only the cases where the attack data has a more specific IP prefix, cross this threshold. In Figure 2d, the f-score is 0.75.

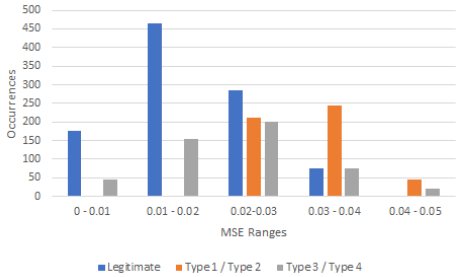
We conclude that auto-encoders are suitable for identifying anomalous BGP updates, and will focus on improving detection performance in our future work.



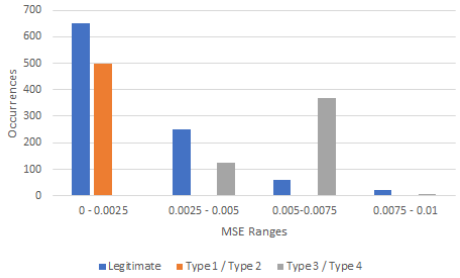
(a) Expt. 1 - AS origin



(b) Expt. 1 - AS Path



(c) Expt. 2 - AS origin



(d) Expt. 2 - AS Path

Fig. 2: Auto-Encoder Error. In the best case, Legitimate (Blue) and Attack (Other) Data should be well separated.

References

[1] H. Ballani, P. Francis, and X. Zhang, "A study of prefix hijacking and interception in the internet," in *ACM SIGCOMM Computer Communication Review*, vol. 37, no. 4, 2007, pp. 265–276.

[2] M. Apostolaki, A. Zohar, and L. Vanbever, "Hijacking bitcoin: Routing attacks on cryptocurrencies," in *Security and Privacy (SP), 2017 IEEE Symposium on*. IEEE, 2017, pp. 375–392.

[3] X. Hu and Z. M. Mao, "Accurate Real-time Identification of IP Prefix Hijacking," *IEEE Symposium on Security and Privacy*, pp. 3–17, 2007.

[4] Z. Zhang, Y. Zhang, Y. Hu, Z. Mao, , and R. Bush, "iSPY: Detecting IP Prefix Hijacking on My Own," *ACM SIGCOMM*, pp. 327–338, 2008.

[5] A. Lutu, M. Bagnulo, J. Cid-Suerio, and O. Maennel, "Separating Wheat from Chaff: Winnowing Unintended Prefixes using Machine Learning," *Proceedings of IEEE INFOCOM*, April 2014.

[2]To create a false edge to the legitimate owner of an IP prefix, the victim's origin AS is appended to the selected AS path. For a more specific IP prefix, the victim's IP length is incremented by one. Finally, for an AS falsely claiming to be the owner of an IP prefix, we simply replace the victim's path with the selected AS path.

[6] I. Goodfellow, Y. Bengio, and A. Courville, "Deep Learning," 2016, http://www.deeplearningbook.org.

[7] TensorFlow, "TensorFlow," 2018, https://www.tensorflow.org/guide/keras.

[8] U. of Oregon, "University of Oregon Route Views Archive Project," 2018, http://www.routeviews.org.