Border Gateway Protocol (BGP) is the de facto protocol for inter-domain routing in the Internet. BGP performance has historically been accepted but rapidly evolving Internet demands better performance guarantees from BGP. BGP convergence time is notorious for its unpredictable and unbounded limits.

We present a novel way where we dissipate BGP state change messages from multiple Autonomous Systems (ASes). This results in multiplication of BGP propagation effort. We used OpenFlow capable routers distributed across the internet to optimally automate this process. Our approach also helps to mitigate IP prefix hijack and Multiple Origion Autonomous System (MOAS) conflicts (quantify). We used Oregon RouteViews in our simulations. Experimental results showed that this scheme can significantly increase BGP propagation (quantify) and alleviate BGP security issues (quantify). It is applicable to most of the internet. Our approach is also endorsed by mathematical models.

Categories and Subject Descriptors
C.2.1 [Computer Communication Networks]: [Network Architecture and Design]

General Terms
Design, Security

Keywords
BGP, SDN, Multiple Origin AS, Internet Architecture

1. INTRODUCTION & MOTIVATION

The Internet is a collection of thousands of Autonomous Systems (ASes), knitted together using BGP. AS is an economic entity operated by a single institution. It consists of switches and other equipment with a clearly defined routing policy. Border routers of each AS announce a list of IP prefixes that originate from within that AS. Each prefix consists of a 32-bit address and a mask length (e.g., 192.168.0.0/24 consists of IP addresses from 192.168.0.0 to 192.168.0.255). These IP prefixes are not fixed and ASes can withdraw and announce new prefixes at any time.

Whenever an AS has to update or withdraw an IP prefix, it uses BGP UPDATE message to announce its new set of IP address prefixes to BGP routers in ASes with direct BGP peering. A BGP peering is a direct relation between two ASes where they exchange traffic based on economic interests. The BGP routers are responsible for propagating information of the prefixes to routers in other ASes. Theoretically, this consume and share strategy is designed to result in a state of full convergence over the whole Internet.

In practice, however, BGP shows unsatisfactory convergence properties. BGP convergence time has become unpredictable and problematic [17] due to an ever growing AS map, BGP configuration errors [12], errors in software routers [20] and persistent router oscillations [19, 6]. During the early 2000s, number of solutions were proposed to address BGP convergence issues [?, ?] but this problem has reemerged as Internet bandwidth has exponentially increased in recent years [?, ?]. On the other hand, cloud computing, Internet telephony and HTML5/AJAX applications have become the predominantly popular services on the Internet. The responsiveness requirements of these applications have renewed challenges for BGP as convergence of BGP in the Internet is no more acceptable. Kushman et al. show how slow BGP convergence time adversely effects VoIP calls made between Europe and USA [10]. Similarly, QoS improvements in BGP are also limited due to slow convergence of BGP [?].

Moreover, unpredicted BGP convergence time, combined with configuration errors is a major source of security issues. As the Internet lacks authoritative verification of IP prefix ownership, a mischievous router can announce IP prefixes of routes it does not own and even manipulate route attributes in the routing updates it relays to neighboring routers. This leads to IP prefix hijacking which is a major security threat to tackle [3]. Similarly, a Multiple Origion Autonomous System (MOAS) conflict occurs when more than one AS appears to originate the same prefix [22]. (Suppose, prefix $p$ is reachable from AS path $(a_1, a_2, a_3, ..., a_m)$ and also from AS path $(b_1, b_2, b_3, ..., b_n)$. A MOAS conflict occurs when $a_m \neq b_n$.) These security issues have caused routing anomalies of global prominence. For example, the propagation of incorrect UPDATEs from the Chinese AS23724 causing all Facebook traffic to be routed through China and the global outage of YouTube [7] resulting from Pakistan’s PTCL wrong configuration of BGP.

In this paper, we propose a system to help reduce
BGP convergence time and mitigate Internet routing tables corruption due to IP prefix hijack and MOAS attacks. Our solution is not clean-slate and can be incrementally integrated in the current BGP network. With a little participation from individual ASes, experimental analysis of our system shows that even unconvergable IP prefixes also tend to converge along with speeding up convergence time of other IP prefixes. We show over 50% reduction in convergence time and propose ways to further improve the convergence rate.

The rest of the paper is organized as follows: Section 2 presents our approach at addressing BGP convergence using SDN and gives justification for our approach followed by a section on analysis of our system. Then we develop a Mathematical Model of our claims in Section 4. We present a section on evaluations and analysis, followed by related work in the field and then conclusion in Section 7.

2. OUR APPROACH

In this section, we will describe how we exploit the increasing number of OpenFlow capable routers spread across AS edges to address the problems of BGP convergence.

A change in BGP state is propagated breadth-first. At times, the reachability information (AS_PATH) in one router is updated to a relatively optimal path learned from another router. This breadth-wise increase in propagation and subsequent optimization of paths continues and eventually, the whole Internet is updated to a new BGP state. We aim at solving some of BGP’s problems by reasserting reachability information from multiple points (herald routers), hence multiplexing convergence rate.

Herald routers form a bed of OpenFlow capable BGP speakers spread across the Internet with centrally managed OpenFlow controller(s). Large Internet companies that operate multiple data centers distributed globally, are adopting SDN for better control of their inter-datacenter network. As of this writing, Google has 13 such ASes with OpenFlow-enabled switches. Google primarily uses OpenFlow for traffic engineering and QoS purposes [9]. In our study, we use Google’s inter-domain SDN deployment as an example. We exploit the fact that routers in these ASes are managed through a trusted web of OpenFlow controllers with speedy and high deliverability guarantees.

These routers are not self-similar: they are geographically distributed and have different edge densities. Our strategy uses herald routers along with their controller(s) to efficiently relay UPDATE messages from one part of the Internet to another.

As a single institution owns herald routers, they are theoretically connected to only one controller. But to add resilience to the system, they can be connected to different controllers that form a distributed controller. In the case of a distributed controller, BGP messages intercepted at all herald routers can still be observed centrally [18]. There are two ways by which OpenFlow protocol allows us to direct BGP state change messages (keep alive or UPDATE) to the controller: (i) direct all BGP flows from switches to the controller via data path and (ii) intercept BGP message and direct them to the controller over the control channel [11]. We have developed a Floodlight OpenFlow controller [2] based application that collects all BGP UPDATE messages (Update or Withdraw) using both of these approaches.

As we can dynamically query Forwarding Information Base (FIB) in a router, we can proactively perform offline calculations to determine the optimal paths between all pairs of these routers. Whenever a change of topology initiates at herald routers, we update optimal reachability path between these routers at our controller, without waiting from network to update us. If a certain topology change in the Internet can update the optimal paths between the herald routers,
BROADCASTING STATE CHANGE MESSAGES AND UPDATES IN A BGP NETWORK

Figure 3: On receiving UPDATE from m finds itself in AS_PATH and ignores the UPDATE. But this UPDATE is also pushed to n which further relays it. l will eventually receive an UPDATE from k.

The information is propagated through BGP state change (UPDATE) messages. These state change messages, combined with UPDATE messages resulting from change in peering policies eventually arrive at one of the herald routers. UPDATE messages are intercepted and sent to our OpenFlow controller that in turn appends the pre-calculated optimal path between a pair of our herald routers to the AS_PATH we have received. The UPDATE message we have received is updated with this new AS_PATH and this UPDATE is injected in our respective herald router. Hence, our controller maintains an updated map of optimal paths between all herald routers. Furthermore, all herald routers also have optimized AS_PATH that would have been generated after BGP convergence.

Algorithm 1 Calculate best Herald Router

1: \( q \leftarrow \) Herald Router that received state change message
2: \( r \leftarrow \) Herald Routers - \( q \)
3: \( x \leftarrow \) weight of degree of connectivity, \( x \in [0, 1] \)
4: \( y \leftarrow \) weight of distance, \( y \in [0, 1] \)
5: \( a_{ij} \leftarrow \) total peers of i, \( i \in r \)
6: \( a_{\text{max}} \leftarrow \max(\forall a_{ij}, i \in r) \)
7: \( a_{\text{min}} \leftarrow \min(\forall a_{ij}, i \in r) \)
8: \( b_{ij} \leftarrow \) total hops between i and j
9: \( b_{\text{min}} \leftarrow \min(\text{total hops between } i \text{ and } j, \forall j \in r) \)
10: \( b_{\text{max}} \leftarrow \max(\text{total hops between } i \text{ and } j, \forall j \in r) \)
11: \( f \leftarrow \) weighted harmonic mean
12: for \( i \) in \( r \) do
13: \( T_i \leftarrow f\left(\frac{a_{ij} - a_{\text{min}}}{a_{\text{max}} - a_{\text{min}}} \times x, \frac{b_{ij} - b_{\text{min}}}{b_{\text{max}} - b_{\text{min}}} \times y\right) \)
14: end for
15: if MOAS or IP Prefix Hijack then
16: \( j \leftarrow \) router originating MOAS or IP Prefix hijack
17: \( r \leftarrow \) filter(r, \( b_{ij} > \text{const} \))
18: end if
19: for \( l \) in \([\text{sorted}(r), \text{w.r.t } T] \) do
20: if \( l \) MRAI counter has expired and \( b_{ij} > 2 \) then
21: \( z \leftarrow l \)
22: break
23: end if
24: end for

Broadcasting state change messages to all herald routers is inefficient and is not scalable. As illustrated in Algorithm 1, we have devised a formula to optimally select herald router(s) for dissipation of the state change messages.

Our algorithm considers edge density of an AS and its distance from AS where original UPDATE was created to probabilistically find a candidate router where UPDATE should be injected. We further discuss these strategies in Analysis section.

As a BGP message with AS_PATH attribute passes through an autonomous system, the AS number of the transmitting router is added to the AS_PATH attribute. If a router can find its own AS number in that path list, it considers it to be a routing loop and ignores it. To avoid routing loops and still remain effective, we insert UPDATE in the middle of the path between two edges of the Internet. This strategy is illustrated in Fig. 3.

IP Prefix hijack and MOAS attacks can be observed using IP Prefix Hijack Alert System (PHAS). PHAS notifies us about routers in the BGP network that are announcing faulty BGP routes. Our strategy is to find a herald router in proximity of the misbehaving router and push a more optimal path as compared to route published by faulty router. Our pushed UPDATE cancels out the disturbance caused by faulty UPDATE and brings back system to normal state. The closer and denser herald router(s) we find in proximity, the faster mitigation is possible. Detailed approach is illustrated in Algorithm 1.

3. ANALYSIS

Unlike other architecturally similar but centralized SDN applications [23, 24], we do not calculate FIB entries for individual routers. In centralized methodologies, FIB entries are pushed to each router after calculating them offline. To calculate FIB entries, we have to maintain and continuously update the BGP table in the controller. BGP table contain approximately 160000 entries, and the number is increasing gradually in recent years [5]. So, storing BGP table in the controller, along with the calculation of FIB entries for all routers is computationally expensive and difficult to scale. We refrain from calculating the FIB entries for each router in the controller. Instead, we inject a BGP UPDATE using BGP daemon that we initiate for pushing a state change message into the router. This hybrid approach is scalable and does not load our controller with processing of the RIB table.

OpenFlow is de facto protocol of SDN. OpenFlow’s control packets are destination bound and flow over TCP that, as compared to BGP, provides high transport rate and guarantees. BGP messages have to propagate through the network: each hop has to process the message before deciding if and whom to forward the message. We exploit the fact that we already have a parallel and trusted network with high deliverability guarantees in form of SDN control network.

It must be noted that in our scheme, all capabilities attained using OpenFlow can also be attained by a client-server architecture. We give a comparative study in related work section about such proposals.

4. MATHEMATICAL FOUNDATIONS

Assume a connected network of N nodes with the average rate of infection \( \beta \) across all links with \( I \) representing the total number of nodes infected at time \( t \). Then assuming the same rate of infection in the subgraphs, the rate of change of nodes infected is defined as follows:
\[ \frac{dt}{dt} = \beta I(N - 1) \]

Above equation results into :

\[ I = \frac{e^{\beta t}}{c + e^{\beta t}} \]

And the time it takes to reach the certain level, \( I \) is:

\[ t = \frac{\ln \frac{I(N - 1)}{N - 1}}{\beta} \]

A graph \( G \) contains a set of nodes \( V \) and edges that are the links between two different BGP peers.

When the original BGP UPDATE from its native router start dissipating, the herald router receives this UPDATE and injects the updated UPDATE into an injection point selected by controller using Algorithm 1. There will be instances in the network graph where the original and imitated UPDATE will over lap. These are the points where the cuts are to be made in the graph. Take these nodes as: \( C \in V \). The set \( C \) will also include the node which due to export/import policy of the AS will result in directed links. subgraphs resulted from the cuts \( C \) are:

\[ S = \text{cut}(G, C) \]

In case of more than one subgraphs, at least one injection node will be chosen in the each subgraph consequently generating different epidemic dissipation scenarios. With each model starting the spread at \( t_0 \) and subgraphs \( G_n \) where each subgraph contains \( k_n \) number of nodes.

\[ |V| = \sum k_n \]

and each model has to reach \( m_n \) level of infection, where

\[ I = \sum m_n \]

Time for each subgraph, \( G_n \), of size \( k_n \) to reach infection level \( m_n \) is:

\[ t_n = \frac{\ln \frac{m(k_n - 1)}{k_n - m_n}}{\beta} \]

With conditions, \( k_n \leq N \), \( m_n \leq I \) and \( t_n \leq t \) Now time for all the subgraphs to reach a global level of infection, in our case advertisements, through achieving individual infection level \( m_n \) is

\[ t' = \max(t_1, t_2, ..., t_n) \]

Which clearly shows that

\[ t' \leq t \]

This proves that the time is bound to decrease or, in the worst case, stay the same

5. EVALUATION

We evaluated our system for BGP convergence on real, derived and artificial topologies. Our simulation results support our claims that reassertions can help reduce convergence time. We used SSFNet to simulate BGP network.

![OpenFlow Controller with Clique (left) and Near Clique (right)](image)

5.1 Clique and Near Clique

We define near clique graph as

\[ \max(\text{total hops between } i \text{ and } j, \forall i, j \in r) \leq 2 \]

whereas, strictly clique graph is

\[ \max(\text{total hops between } i \text{ and } j, \forall i, j \in r) = 1 \]

Fig. 4 illustrate such cases, where we measured the convergence time when an UPDATE is pushed from multiple nodes. By introducing an OpenFlow controller, we are adding two extra hops in the topology which explain why our strategy had no advantage. As expected, introducing reasserting nodes did not help converge the system. It is important to note that introducing our strategy did not adversely effect convergence.

In some cases, a MRAI timer in BGP speaker can cause more delay than time taken by our controller to push an UPDATE. In those cases, our approach would still have impact, specially in cases of near cliques.

5.2 Real World Topology at Internet Edges

Our strategy is specifically useful in scenarios where internet edges are not well connected. Internet connectivity between South Asia and America is one such case. Fig. 5 shows how AS55541 (Pakistan MCB) is connected to AS308.
reachability index, least three or more hops away. In the internet, on average 97 percent of the routers are at a distance of 4 hops away. In the internet core, we calculated that for a given AS, on average, 2.37 hops are required to reach the destination. We have simplified this graph by removing longer paths between any two ASes. Table 1 shows how we reduced their convergence time using our scheme.

<table>
<thead>
<tr>
<th>Case</th>
<th>BGP convergence from AS 355541 (Pakistan MCB) to AS 5089 (DoD)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal Convergence</td>
<td>2.370451</td>
</tr>
<tr>
<td>Insertion at 3rd Hop (AS 721)</td>
<td>1.674433</td>
</tr>
<tr>
<td>Insertion at 5th Hop (AS 200)</td>
<td>1.505769</td>
</tr>
</tbody>
</table>

5.3 Internet Core

We evaluated our system by simulating different experiments on internet AS graph. RIP provided by Oregon RouteViews for July 1st, 2013 was used to create AS graph of the internet. We calculated that for a given AS in the internet, on average 97 percent of the routers are at least three or more hops away. i.e, reachability index,

\[
\frac{\sum \{ J \}}{\sum \{ J \}} \times 100 = 96.95\%
\]

This percentage shows that even random selection of herald routers in greater internet has significant effect on convergence time.

5.4 A Case of Google

In this section, we exploit the fact that we already have some OpenFlow routers in the internet which can become herald routers. We analyze a case where Google’s OpenFlow capable network act as herald routers.

Table 2 shows distance between these Google ASes. These ASes have 60% reachability index which is lower than global average but this is still good enough for our scheme to work. We traced an UPDATE which eventually used one of Google’s AS (ASN 32381) to spread across the internet. Table 2 shows how we reduced their convergence time using our scheme.

6. RELATED WORK

There are many proposed solutions to BGP problems that could not get adopted because of reliance on centralized architecture leads to privacy and security issues. Specially, the fact that giving a third party ability to modify your traffic routes or influence behavior of your network leads to rejection of such proposed solutions by network service providers. We draw novelty leads to rejection of such proposed solutions by network service providers.

Table 2: Distance between hops between ASes owned by Google

<table>
<thead>
<tr>
<th>AS</th>
<th>15169</th>
<th>16591</th>
<th>36492</th>
<th>41264</th>
<th>40873</th>
<th>32381</th>
</tr>
</thead>
<tbody>
<tr>
<td>15169</td>
<td>2 1 2 1 1 2 1 2 3 1 1 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16591</td>
<td>2 2 3 2 3 2 2 3 3 2 3 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>36492</td>
<td>2 3 3 2 3 3 3 3 3 2 3 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>41264</td>
<td>2 3 3 2 3 3 3 3 3 2 3 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>40873</td>
<td>3 3 2 3 3 2 3 3 2 3 4 4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32381</td>
<td>3 3 4 3 3 3 4 3 4 3 4 4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

7. CONCLUSIONS

The results above clearly show that by introducing UPDATEs from multiple vantage points, we can increase rate of convergence. This does not give guarantees of...
convergence or does not increase worst-case bounds, but average rate of convergence increases as documented. To our knowledge, this is first paper which banks on currently present infrastructure and does not require any explicit coordination between providers.

8. REFERENCES