Network Diversion: Forcing Internet Traffic Through Desired Paths

Executive Summary

David Risius, MAJ, USA
Christopher Wood, Capt, USMC

Background

There is an ever increasing trend in our society to create network-dependent devices, interactions, weapons, business processes, etc. As this dependency continues, the Internet continues to grow in complexity of routers, servers, data protocols, and clients. We attempt to model one small facet of the Internet, the 'network' of autonomous systems (AS). We then seek to conduct 'network diversion', in attempting to force traffic between two nodes through a particular exploitation node. Network diversion can be broadly categorized into two methods, incentivization and restriction. Our solutions fall neatly into these categories, providing three possible solutions to accomplish network diversion.

An AS is a group of routers and networks managed by a single organization. An autonomous system is assigned a globally unique number, called an Autonomous System Number (ASN). [1] In turn, an Internet Service Provider (ISP), a university, or a business will constitute an AS[2]. To visualize an AS, one can view a computer as an Internet Protocol (IP) address which has internet connection through an 'owning' AS.

In an entirely fabricated example, internet communication from a Marine at the Naval Postgraduate School to Headquarters Marine Corps in Quantico, VA could travel from the California University Network AS (ASN:4152), to the Nevada AT&T AS (ASN:26980), to the Washington D.C. Comcast AS (ASN:356), to the Quantico MCB AS(ASN:1765), to the final destination at our recipients NMCI computer. As alluded to in the previous example, ASs peer with each other to exchange traffic, and these peering relationships define the high-level global Internet topology. For the purposes of analysis, these peering relationships are represented with an AS graph, where nodes represent ASes and edges represent peering relationships. An autonomous system shares routing information with other autonomous systems using the Border Gateway Protocol (BGP). Among other functions, these protocols record and advertise the AS-level path taken by internet communication.

The Cooperative Association for Internet Data Analysis (CAIDA) is a collaborative undertaking among organizations in the commercial, government, and research sectors aimed at promoting greater cooperation in the engineering and maintenance of a robust, scalable global Internet infrastructure. CAIDA collects several different types of data at geographically and topologically diverse locations, and makes this data available to the research community to the extent possible while preserving the privacy of individuals and organizations who donate data or network access.[3] Among this data is the IP-address level path extracted from BGP. CAIDA then performs an AS-lookup to assign this IP path to the owning

[1] Rouse, Margaret, http://searchnetworking.techtarget.com/definition/autonomous-system
[2] Alderson, David, 'The Many Facets Of Internet Topology and Traffic'
[3] http://www.caida.org/data/overview/

AS of each IP, leaving us with an AS-level path of internet communication for real-world traffic. We extracted 24 hours worth of global internet traffic, as seen from CAIDA sensors, from 0800, 01 April to 0800, 02 April to build a very brief snapshot of AS-level internet topology. [4]

Scope

The scope of this final project involves developing the AS numbers as a network using publicly available from CAIDA for one day worth of traffic on the internet. The initial input file for the project consists of a text file with 256,000 separate lines of data representing message traffic from one AS to another. We then abstract that network to represent the BGP routes, and finish by applying our three network diversion models to the network.

The objective of this project is to answer the following question:

1. **Can internet traffic between two ASs be forced through a particular third AS? In the abstract representation shown in Figure 1, our goal is to force message traffic from AS:714 to AS:4983 to go through AS:32, when traffic would have normally taken an alternative internet routing path.**

Limitations and Assumptions

We were chiefly limited by a lack of sufficient understanding of BGP and internet topology. Understanding either of these topics is an endeavor unto itself, and we were forced to make necessary simplifications and assumptions. We go on to assume that the CAIDA data collected is sufficient to form a complete AS topology of the internet. We also assume that there is a method to attack nodes (AS), however we do not specify how to attack them. Finally, we rely on the assumption that edge lengths are representative of BGP routing.

Three methods used for Network Diversion.

Method 1, Decreasing Shortest Path. This method involves incentivizing the interceptor node into the shortest path by incrementally decreasing the advertised cost to go through it. To start, we apply equal unit lengths to all edges in the network. Then, a candidate path is found between the desired start and end node containing the intercept node. Using the Networkx package in Python, we iterate over increasingly negative lengths until the intercept node is in the shortest path. One possible issue with this method is how we can advertise a negative cost over a node. A possible better way to address this would be to find a way to give incentives for going through our interceptor node.

Method 2, Sliding Shortest Path. This method takes the same candidate path found in the decreasing shortest path and eventually forces traffic through the interceptor node by incrementally attacking nodes not in the candidate path. The Python program begins at the start node and calculates the shortest path from the start to end nodes. If the first node in the shortest path is not the candidate

[4] The IPv4 Routed /24 AS Links Dataset - <01Apr2013 - 02Apr2013>, http://www.caida.org/data/active//ipv4_routed_topology_aslinks_dataset.xml.

path, the program attacks (removes) the node and recalculates the shortest path. We repeat this process until our interceptor node is contained in the shortest path calculation from the start to end. A benefit of this process is that it removes nodes along the same route that the data would take. A possible issue with this method may involve catastrophic damage to the network once all the attacks are completed.

Method 3, Forced Minimum-Cut. This method finds the minimum node cut of the network containing our interceptor node and then attacks all the nodes in the minimum cut except the interceptor to force traffic through our desired node. We achieve this by initially setting all node capacities to one and all edge capacities to infinity. We then set all the edge capacities and node capacities on the candidate path except the interceptor node to infinity. Next we use a linear program to find the minimum cut of the network. Since all the edges in our candidate path except the interceptor node are infinity, we guarantee that the interceptor node will be in the minimum cut. Next, we attack every node in the minimum cut except the interceptor node to force traffic through our node. An issue with this method is that if there are direct edges between intermediate nodes in the candidate, traffic can bypass our interceptor node. When we find the candidate path, we need to ensure there are no direct arcs between intermediate nodes.

Results

We now present the results of our three methodologies. Our first model, the incentive of the decreasing shortest path, yields a decreased path length on our intercept node of -4. This is relatively intuitive due to the fact that the average hop length in the network is approximately 3. Therefore, it is clear that a -4 will be necessary to overcome the best of all other alternatives. The second model, known as Sliding Shortest Path, utilizes a restriction of alternatives which cuts 355 nodes in order to force traffic through our intercept node. This is a remarkably low number of cuts, given the size of the network. Our final model, Maximum Flow Forced Minimum Cut, conducts a similar restriction as Sliding Shortest Path, but we now solve the number of nodes to cut with optimality. This produces a reduced cut list of 309 nodes.

Future Work

We would like to be able to reformulate our diversion paths with a set of interception nodes rather than a single node. This would likely be more representative of most network interception operations. The next capability we'd like to implement is the addition of costs and prohibitions to intercepting certain nodes, due to technological, financial, or peering relationship concerns. Finally, we've found that these three models provide a relatively robust solution set towards the general problem of network diversion. Many common network problems can be thought of as a network diversion problem in order to provide perspective and analysis on the network operations.