

实验说明

中国科学院计算技术研究所

网络技术研究中心

提纲

- Mininet实验环境
- 实验说明
 - BGP分析实验
 - BGP前缀劫持攻击及检测实验
 - TCP公平性实验
 - HTTP服务器实验

Mininet实验环境

- Mininet是一个可以支持快速搭建模拟网络的平台
 - <http://mininet.org/>
- Mininet环境只能安装在Linux系统下，推荐使用Ubuntu发行版，版本号从20.04到最新都可以，64位或32位都行
- 如果物理机为Windows系统，可以使用虚拟机方式安装Linux系统，推荐使用VirtualBox虚拟机
- 运行Mininet环境时需要root权限

Mininet安装和验证

- \$ sudo apt install mininet

- \$
- n
- n

```
alvin@alvin-ubuntu: ~/networking/mininet
alvin@alvin-ubuntu:~/networking/mininet$ sudo mn
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Starting CLI:
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2
h2 -> h1
*** Results: 0% dropped (2/2 received)
mininet> 
```

BGP分析实验

背景知识

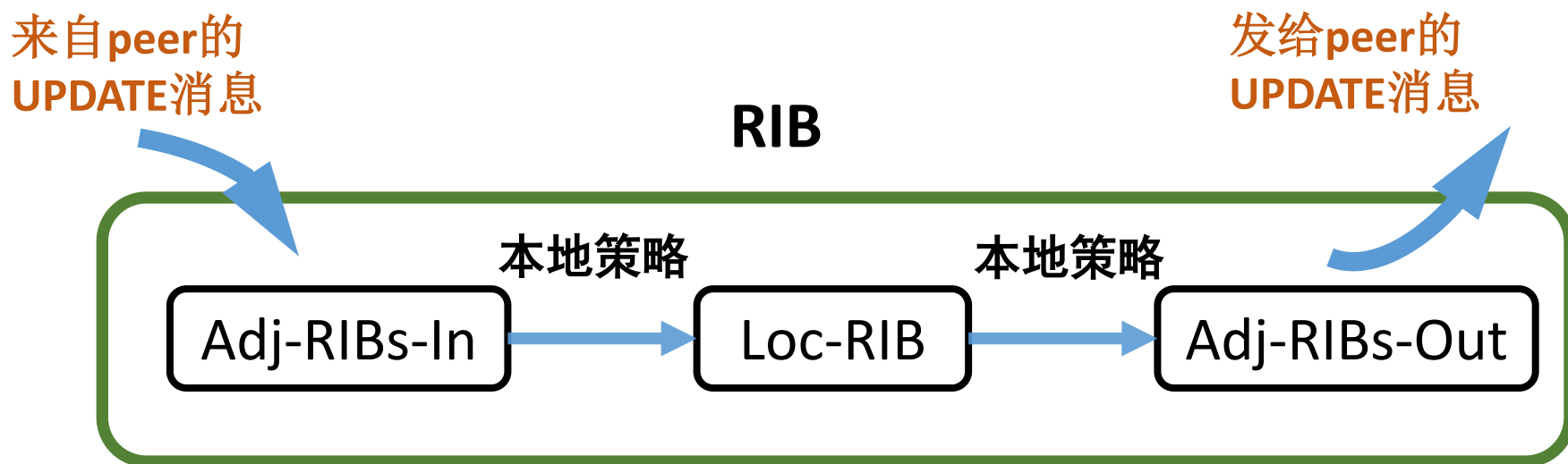
- BGP路由器之间使用BGP消息来交换网络层可达性信息（Network Layer Reachability Information, NLRI），其中包括该可达性信息穿过的自治系统（AS）列表（即AS_PATH）
- 使用AS_PATH属性可以构建AS连接图（AS拓扑），进而用于网络管理、网络优化等研究
 - 如修剪路由环路、在AS级别上执行某些策略决策等
- AS_PATH属性可从BGP路由器记录的Routing Information Base（RIB）和BGP UPDATE数据中获取

基本概念

- BGP路由器之间使用TCP传输消息，建立连接的对端称为peer
- 收到来自peer的BGP消息后，路由器将其存储在自己的路由信息库（Routing Information Base, RIB）中，根据策略，决定是否将自己的AS号加在AS_PATH中并向其他BGP路由器发送
- 理论上，RIB由三部分组成：Adj-RIBs-In，Loc-RIB和Adj-RIBs-Out
 - Adj-RIBs-In：从peer的UPDATE消息中学到的路由信息
 - Loc-RIB：结合了本地策略从Adj-RIBs-In中选取、用于本地路由的路由信息
 - Adj-RIBs-Out：存储本地选取后用于告知其他路由器的路由信息

更新过程

- 在BGP router间建立连接后，首先会向peer发送Adj-RIBs-Out中的路由信息，此后当Adj-RIBs-Out有变化时才会向peer发送增量的更新（UPDATE）



数据集

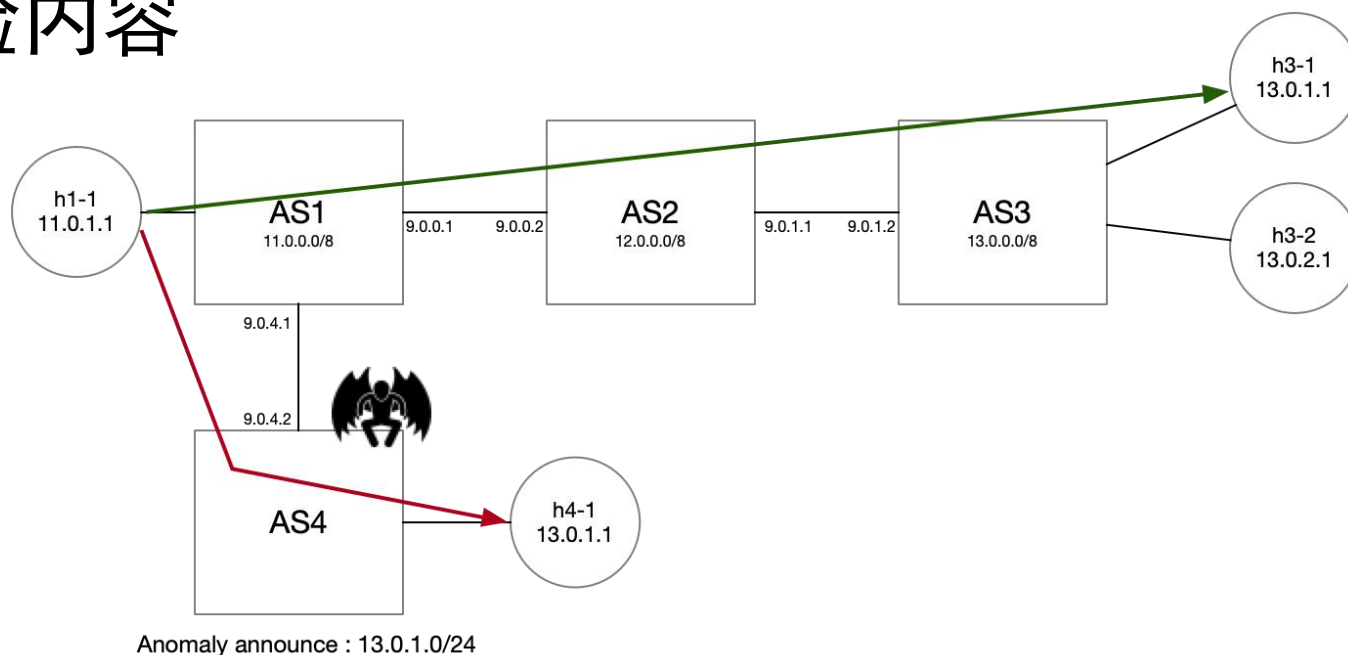
- Route Views 和RIPE NCC多年前开始在全球多处收集路由数据，主要的存储格式是MRT格式（RFC 6396），本实验即使用这些数据
 - <http://www.routeviews.org/routeviews/index.php/archive>
 - <https://www.ripe.net/analyse/internet-measurements/routing-information-service-ris/ris-raw-data>
- CAIDA提供BGPStream工具获取并分析这些数据

实验要求

- 编写脚本程序处理上述数据，使用RIB和UPDATE中的AS_PATH属性构建任意时刻互联网AS级别拓扑，分析如下内容：
 - 对比分析中国与美国的AS拓扑差别（选取任一时刻即可）
 - 对比分析IPv4与IPv6网络的AS拓扑差别（选取任一时刻即可）
 - 从AS拓扑上分析近两年来IPv6的部署情况（选取几个时刻即可）
- BGP中存在BGP路由泄漏（BGP Route Leak，参考RFC 7908）的问题。近年来有过多次因为BGP路由泄漏导致网络瘫痪的事件，例如2017年由于Google工程师配置错误导致的BGP路由泄漏引起了日本发生大规模网络故障。请从互联网上寻找三个BGP路由泄漏案例，在数据集中分析BGP泄漏发生前、发生中、发生后，受影响AS的BGP RIB或者受影响网络的AS拓扑的变化
- **提交：**分析代码 & 分析报告

BGP前缀劫持攻击及检测实验

实验内容



- 实验拓扑：如图，AS1-AS4 是 4 个边界网关路由器。AS1 拥有 11.0.0.0/8，该网段内有一个 host（h1-1），它作为客户端访问位于 AS3 的 h3-1 和 h3-2 服务器。AS3 拥有 13.0.0.0/8。正常情况下 h1-1 访问 13.0.1.1 的流量按照绿色路径所示。
- 发动攻击：AS4 是一个恶意 AS，它会劫持 h1-1 通往 13.0.1.1 的流量，转发到自己域内的恶意服务器。劫持方式是 AS4 宣告一条更长的网络前缀 13.0.1.0/24。劫持后 h1-1 的请求会按照红色路径到达恶意服务器 h4-1。通往 h3-2 的流量不受影响。

实验工具环境搭建

- 实验环境

- 工具：FRRouting（需要 bgpd 实现 BGP 协议交互、zebra 将路由表项下发到 Linux 内核）、bgpdump（分析 bgp updates 消息）

- 工具安装 `sudo bash ./install.sh`

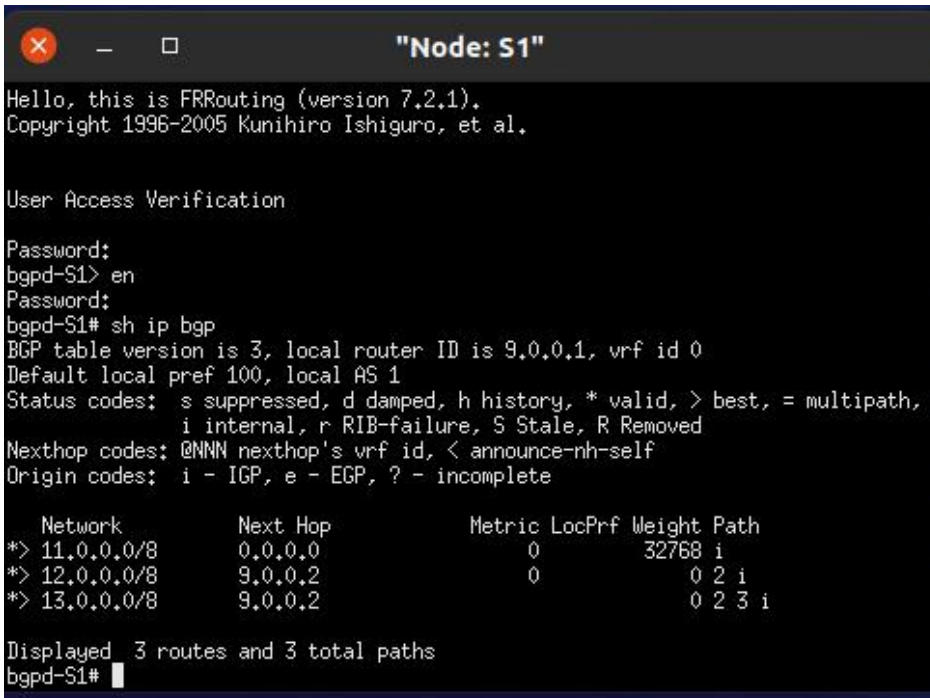
- FRRouting
 - bgpdump

启动实验

- 运行 `sudo bash ./auto_configure.sh` 配置环境（只需执行一次，配置后就不要再更改实验目录）
- 修改 BGP configure 文件：
 - 仿照 AS1 和 AS2 的 configure 文件补充 AS3 和 AS4 的 bgp 配置文件（bgpd-S3.conf bgpd-S4.conf）
- 启动 mininet 环境，进行实验
 - `$ sudo python bgp.py`

实验步骤：查看 S1 边界路由器的转发表

- 启动 S1 的 xterm 终端 `mininet> xterm S1`
- (S1 xterm terminal)# `./connect.sh`
- Password: `en`
- `bgpd-S1> en`
- Password: `en`
- `bgpd-S1# sh ip bgp`



```

Hello, this is FRRouting (version 7.2.1).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

User Access Verification
Password:
bgpd-S1> en
Password:
bgpd-S1# sh ip bgp
BGP table version is 3, local router ID is 9.0.0.1, vrf id 0
Default local pref 100, local AS 1
Status codes: s suppressed, d damped, h history, * valid, > best, = multipath,
               i internal, r RIB-failure, S Stale, R Removed
Nexthop codes: @NNN nexthop's vrf id, < announce-nh-self
Origin codes:  i - IGP, e - EGP, ? - incomplete

   Network        Next Hop        Metric LocPrf Weight Path
*> 11.0.0.0/8      0.0.0.0          0           32768 i
*> 12.0.0.0/8      9.0.0.2          0             2 i
*> 13.0.0.0/8      9.0.0.2          0             2 3 i

Displayed 3 routes and 3 total paths
bgpd-S1#
```

Waiting 3 seconds for s
Starting zebra and bgpd
Starting zebra and bgpd
Starting zebra and bgpd
Config host h1-1-eth0 1
Config host h1-2-eth0 1
Config host h1-3-eth0 1
Config host h2-1-eth0 1
Config host h2-2-eth0 1
Config host h2-3-eth0 1
Config host h3-1-eth0 1
Config host h3-2-eth0 1
Config host h3-3-eth0 1
Config host h4-1-eth0 1
Config host h4-2-eth0 1
Config host h4-3-eth0 1
Starting web servers
*** Starting CLI:
mininet> xterm S1
mininet>

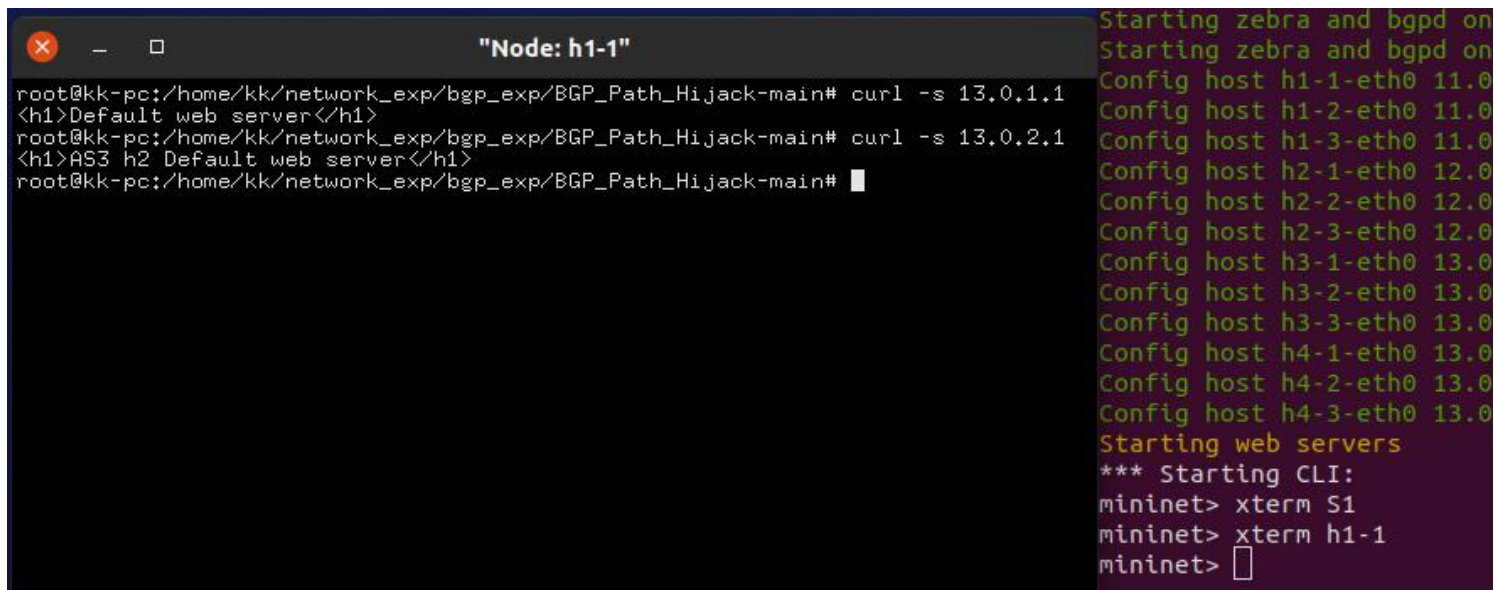
实验步骤： h1-1 正常访问 13.0.1.1 和 13.0.2.1

- mininet> *xterm h1-1*
- (h1-1 xterm terminal) # *curl -s 13.0.1.1*

<h1>Default web server</h1>

- (h1-1 xterm terminal) # *curl -s 13.0.2.1*

<h1>AS3 h2 Default web server</h1>



```
root@kk-pc:/home/kk/network_exp/bgp_exp/BGP_Path_Hijack-main# curl -s 13.0.1.1
<h1>Default web server</h1>
root@kk-pc:/home/kk/network_exp/bgp_exp/BGP_Path_Hijack-main# curl -s 13.0.2.1
<h1>AS3 h2 Default web server</h1>
root@kk-pc:/home/kk/network_exp/bgp_exp/BGP_Path_Hijack-main#
```

```
Starting zebra and bgpd on
Starting zebra and bgpd on
Config host h1-1-eth0 11.0
Config host h1-2-eth0 11.0
Config host h1-3-eth0 11.0
Config host h2-1-eth0 12.0
Config host h2-2-eth0 12.0
Config host h2-3-eth0 12.0
Config host h3-1-eth0 13.0
Config host h3-2-eth0 13.0
Config host h3-3-eth0 13.0
Config host h4-1-eth0 13.0
Config host h4-2-eth0 13.0
Config host h4-3-eth0 13.0
Starting web servers
*** Starting CLI:
mininet> xterm S1
mininet> xterm h1-1
mininet>
```


实验步骤：AS4 发动前缀劫持

- mininet> xterm S4
- (S4 terminal) #./start_rogue.sh
- (h1-1 terminal) # curl -s 13.0.1.1
- (h1-1 terminal) # curl -s 13.0.2.1

```

"Node: S4"
root@kk-pc:/home/kk/network_exp/bgp_exp/BGP_Path_Hijack-main# ./start_rogue.sh
Killing any existing rogue AS
Executed cmd: mnexec -a 391388 pgrep -f zebra-S4 | sudo xargs kill -9
./stop_rogue.sh: line 3: 403181 Killed                  sudo python run.py --nod
e S4 --cmd "pgrep -f zebra-S4 | sudo xargs kill -9"
Executed cmd: mnexec -a 391388 pgrep -f bgpd-S4 | sudo xargs kill -9
./stop_rogue.sh: line 4: 403189 Killed                  sudo python run.py --nod
e S4 --cmd "pgrep -f bgpd-S4 | sudo xargs kill -9"
Starting rogue AS
Executed cmd: mnexec -a 391388 /usr/lib/frr/zebra -f conf/zebra-S4.conf -d -i /t
mp/zebra-S4.pid > logs/S4-zebra-stdout
2022/09/29 16:59:38 warnings: ZEBRA: [EC 4043309105] Disabling MPLS support (no
kernel support)
Executed cmd: mnexec -a 391388 /usr/lib/frr/bgpd -f conf/bgpd-S4.conf -d -i /tmp
/bgpd-S4.pid > logs/S4-bgpd-stdout
root@kk-pc:/home/kk/network_exp/bgp_exp/BGP_Path_Hijack-main#

"Node: h1-1"
root@kk-pc:/home/kk/network_exp/bgp_exp/BGP_Path_Hijack-main# curl -s 13.0.1.1
(h1)*** Attacker web server ***</h1>
root@kk-pc:/home/kk/network_exp/bgp_exp/BGP_Path_Hijack-main#

h1-1 h1-2 h1-3 h2-1 h2-2 h2-3
*** Done
kk@kk-pc:~/network_exp/bgp_exp/BGP_Path_Hijack-main$
*** Creating network
*** Adding controller
*** Adding hosts:
h1-1 h1-2 h1-3 h2-1 h2-2 h2-3
*** Adding switches:
S1 S2 S3 S4 S5
*** Adding links:
(S1, S2) (S1, S4) (S1, S5) (S2, S3) (S2, S4) (S2, S5) (S3, S4) (S3, S5) (S4, S5)
*** Configuring hosts
h1-1 h1-2 h1-3 h2-1 h2-2 h2-3
*** Starting controller
c0
*** Starting 5 switches
S1 S2 S3 S4 S5
Waiting 3 seconds for sysctl
Starting zebra and bgpd on S1
Starting zebra and bgpd on S2
Starting zebra and bgpd on S3
Config host h1-1-eth0 11.0.1.1
Config host h1-2-eth0 11.0.1.2
Config host h1-3-eth0 11.0.1.3
Config host h2-1-eth0 12.0.1.1
Config host h2-2-eth0 12.0.1.2
Config host h2-3-eth0 12.0.1.3
Config host h3-1-eth0 13.0.1.1
Config host h3-2-eth0 13.0.1.2
Config host h3-3-eth0 13.0.1.3
Config host h4-1-eth0 13.0.1.1
Config host h4-2-eth0 13.0.1.2
Config host h4-3-eth0 13.0.1.3
Starting web servers
*** Starting CLI:
mininet> xterm S1
mininet> xterm h1-1
mininet> xterm S4
mininet>
```

实验步骤：检查 S1 路由表的变化

"Node: S1"

Origin codes: i - IGP, e - EGP, ? - incomplete

Network	Next Hop	Metric	LocPrf	Weight	Path
*> 11.0.0.0/8	0.0.0.0	0		32768	i
*> 12.0.0.0/8	9.0.0.2	0		0	2 i
*> 13.0.0.0/8	9.0.0.2			0	2 3 i

Displayed 3 routes and 3 total paths

bgpd-S1# sh ip bgp

BGP table version is 4, local router ID is 9.0.0.1, vrf id 0

Default local pref 100, local AS 1

Status codes: s suppressed, d damped, h history, * valid, > best, = multipath, i internal, r RIB-failure, S Stale, R Removed

Nexthop codes: @NNN nexthop's vrf id, < announce-nh-self

Origin codes: i - IGP, e - EGP, ? - incomplete

Network	Next Hop	Metric	LocPrf	Weight	Path
*> 11.0.0.0/8	0.0.0.0	0		32768	i
*> 12.0.0.0/8	9.0.0.2	0		0	2 i
*> 13.0.0.0/8	9.0.0.2			0	2 3 i
*> 13.0.1.0/24	9.0.4.2	0		0	4 i

Displayed 4 routes and 4 total paths

bgpd-S1#

刚启动时 S1 的 bgp 路由

AS4 前缀劫持之后的路由

"Node: h1-1"

root@kk-pc:/home/kk/network_exp/bgp_exp/BGP_Path_Hijack-main# curl -s 13.0.1.1

<h1>Default web server</h1>

root@kk-pc:/home/kk/network_exp/bgp_exp/BGP_Path_Hijack-main# curl -s 13.0.1.1

<h1>*** Attacker web server ***</h1>

root@kk-pc:/home/kk/network_exp/bgp_exp/BGP_Path_Hijack-main# curl -s 13.0.2.1

<h1>AS3 h2 Default web server</h1>

root@kk-pc:/home/kk/network_exp/bgp_exp/BGP_Path_Hijack-main#

通往 13.0.1.1 的流量被劫持

通往13.0.2.1的流量不变

Terminal

*** Starting 4 switches

S1 S2 S3 S4

Waiting 3 seconds for systemctl changes to take effect...

Starting zebra and bgpd on S1

Starting zebra and bgpd on S2

Starting zebra and bgpd on S3

Config host h1-1-eth0 11.0.1.1/24, gateway: 11.0.1.254

Config host h1-2-eth0 11.0.2.1/24, gateway: 11.0.2.254

Config host h1-3-eth0 11.0.3.1/24, gateway: 11.0.3.254

Config host h2-1-eth0 12.0.1.1/24, gateway: 12.0.1.254

Config host h2-2-eth0 12.0.2.1/24, gateway: 12.0.2.254

Config host h2-3-eth0 12.0.3.1/24, gateway: 12.0.3.254

Config host h3-1-eth0 13.0.1.1/24, gateway: 13.0.1.254

Config host h3-2-eth0 13.0.2.1/24, gateway: 13.0.2.254

Config host h3-3-eth0 13.0.3.1/24, gateway: 13.0.3.254

Config host h4-1-eth0 13.0.1.1/24, gateway: 13.0.1.254

Config host h4-2-eth0 13.0.2.1/24, gateway: 13.0.2.254

Config host h4-3-eth0 13.0.3.1/24, gateway: 13.0.3.254

Starting web servers

*** Starting CLI:

mininet> xterm S1

mininet> xterm h1-1

mininet> xterm S4

mininet>

"Node: S4"

在 S4 开始劫持

root@kk-pc:/home/kk/network_exp/bgp_exp/BGP_Path_Hijack-main# ./start-rogue.sh

Killing any existing rogue AS

Executed cmd: mnexec -a 323109 pgrep -f zebra-S4 | sudo xargs kill -9

./stop-rogue.sh: line 3: 323310 Killed sudo python run.py --nod

a S4 --cmd "pgrep -f zebra-S4 | sudo xargs kill -9"

Executed cmd: mnexec -a 323109 pgrep -f bgpd-S4 | sudo xargs kill -9

./stop-rogue.sh: line 4: 323318 Killed sudo python run.py --nod

a S4 --cmd "pgrep -f bgpd-S4 | sudo xargs kill -9"

Starting rogue AS

Executed cmd: mnexec -a 323109 /usr/lib/frr/zebra -f conf/zebra-S4.conf -d -i /tmp/zebra-S4.pid > logs/S4-zebra-stdout

2022/09/26 10:17:27 warnings: ZEBRA: [IEC 4043309105] Disabling MPLS support (no kernel support)

Executed cmd: mnexec -a 323109 /usr/lib/frr/bgpd -f conf/bgpd-S4.conf -d -i /tmp/bgpd-S4.pid > logs/S4-bgpd-stdout

root@kk-pc:/home/kk/network_exp/bgp_exp/BGP_Path_Hijack-main#

攻击检测

- 问题：作为 AS3 的管理员，如何检测自己遭到了前缀劫持攻击？
- 查看节点收到的 update 消息：
 - 每个 router 收到的 update 消息保存在 updates 目录下
 - `$ cd updates`
 - `$ sudo chmod +r ./*` (使用 bgpdump 读取需要添加读权限)
 - `$ bgpdump <file-name>` 查看文件内容
 - `$ bgpdump -m <file-name>` 可以使用紧凑格式输出，方便后续在攻击检测工具中处理数据
- 设计一个实时检测工具，持续读取 AS3 收到的 update 消息并提出警告

实验要求

- 补充bgpd-S3/4.conf配置文件，完成BGP前缀劫持攻击
- 设计实现一个实时检测工具，持续读取 AS3 收到的 update 消息并提出警告
- 提交配置文件、工具代码和实验报告

附件文件列表

- auto_configure.sh # 配置设置，只需要执行一次
- bgp.py # BGP实验主文件
- conf # 各节点BGP配置文件
- connect.sh # 连接到bgpd查看路由信息
- install.sh # 安装实验相关工具
- logs
- start_rogue.sh # 发起BGP前缀劫持攻击
- stop_rogue.sh # 停止BGP前缀劫持攻击
- updates
- webserver.py # HTTP服务器

参考资料

- <https://docs.frrouting.org/en/latest/bgp.html>

TCP公平性实验

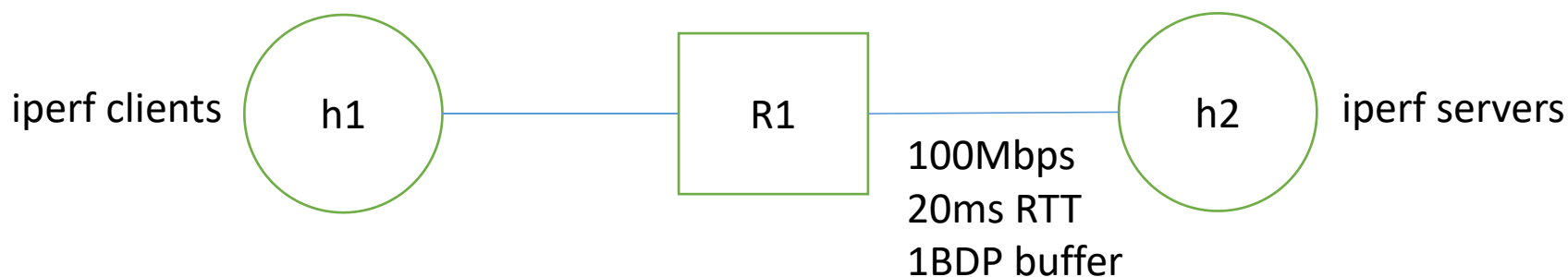
背景

- TCP协议作为应用需求和网络环境之间的性能适配器，其机制和算法一直在演进中
- Linux网络协议栈
 - 将TCP拥塞控制接口抽象出来，支持不同的TCP拥塞控制算法实现
 - 用户只需要实现特定接口，就可以自定义实现面向特定网络场景的拥塞控制逻辑
- Cubic算法：只使用丢包作为拥塞信号，收敛速度慢
- BBR算法：周期性探测带宽和延迟，在高带宽变化环境下收敛性差、公平性差

实验要求

- 设计并实现一种基于丢包和延迟的TCP拥塞控制机制
 - 能够在Linux 4.15+内核环境下编译成模块，并可以加载到内核中
 - 相比于Cubic/BBR算法，所设计的拥塞控制机制具有更好的收敛速度和公平性
- 提交：代码 & 实验报告

实验内容



- 实验拓扑： h1-R1-h2 直连，在 R1-h2 路径上设置速率限制
- 公平性和收敛性测试过程：在 h1 上每隔时间 t 启动 iperf 启动一条 TCP 流，直到有 n 条流同时发送。之后每隔时间 t 结束一条流，直到所有流停止发送
- 测试期间使用 tcpdump 抓包，供后续公平性分析使用

实验工具

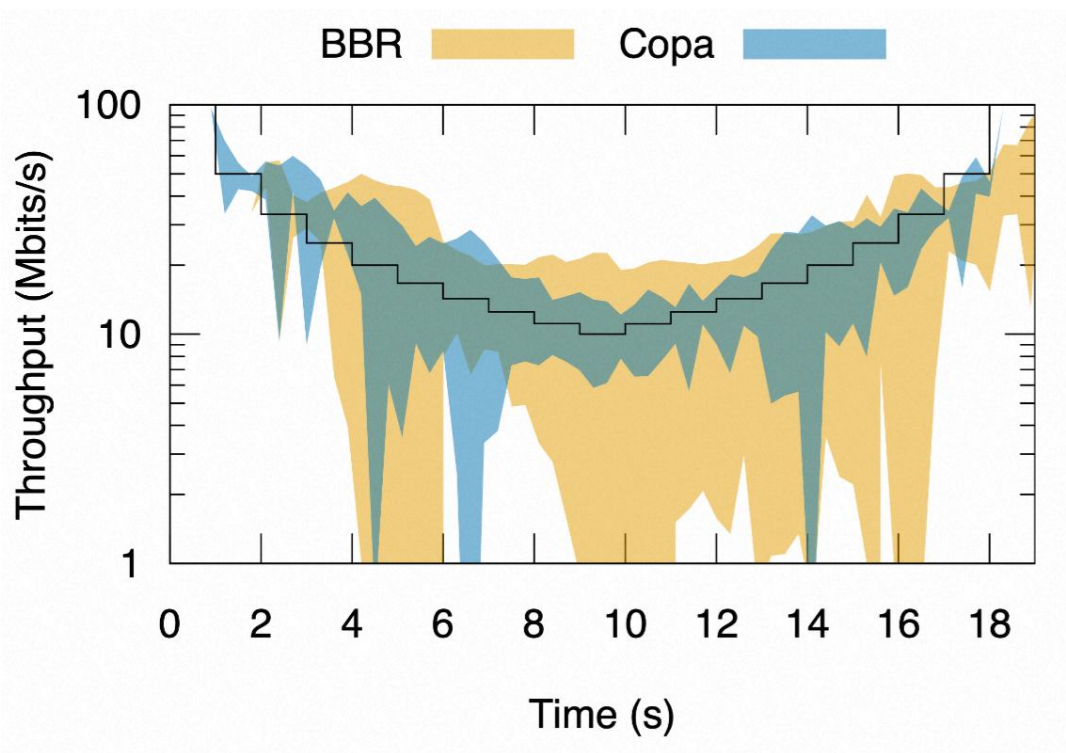
- # python topo.py -c alg_name 创建实验拓扑并测试目标算法
 - 公平性与收敛速度测试函数:
fairness_evaluation(net,cc_name,inter_flow_time = 1,num_flows = 10)每隔 inter_flow_time 加入一条流，直到加入 num_flows 条流为止。之后每 inter_flow_time 一条流结束发送，直到所有流都结束发送
 - 实验过程中， tcpdump 会抓取 h2-eth0 的数据包，供后续公平性分析使用

实验工具

- 公平性分析工具 (`pcap_analyse_tool.py`)
 - `def analyse_algorithm(pcap_file_path, cache=True)` 会分析目标 pcap 文件，由于数据处理会花费大量时间，为了方便后续处理设置了缓存机制
 - `def draw(ax,alg,alpha=0.3,color="r")` 函数说明：以 0.1s 为单位，统计每个时间区间内所有流的平均吞吐量(mean) 和所有流吞吐量的标准差(standard deviation)，绘制出 $\text{mean} \pm \text{standard deviation}$ 的范围

draw() 效果示意图

- 效果如下图所示，图中应有三种拥塞控制算法：Cubic、BBR，以及自己实现的算法。黑色线是理想值。



附件文件列表

- pcap_analyse_tool.py # 分析Pcap文件工具
- topo.py # 实验拓扑
- util.py # 实验相关辅助功能

HTTP服务器实验

HTTP服务器实验

- 实现：使用C语言实现最简单的HTTP服务器
 - 同时支持HTTP（80端口）和HTTPS（443端口）
 - 使用两个线程分别监听各自端口
 - 只需支持GET方法，解析请求报文，返回相应应答及内容

需支持的状态码	场景
200 OK	对于443端口接收的请求，如果程序所在文件夹存在所请求的文件，返回该状态码，以及所请求的文件
301 Moved Permanently	对于80端口接收的请求，返回该状态码，在应答中使用Location字段表达相应的https URL
206 Partial Content	对于443端口接收的请求，如果所请求的为部分内容（请求中有Range字段），返回该状态码，以及相应的部分内容
404 Not Found	对于443端口接收的请求，如果程序所在文件夹没有所请求的文件，返回该状态码

实验流程

- 根据上述要求，实现HTTP服务器程序
- 执行`sudo python topo.py`命令，生成包括两个端节点的网络拓扑
- 在主机h1上运行HTTP服务器程序，同时监听80和443端口
 - `h1 # ./http-server`
- 在主机h2上运行测试程序，验证程序正确性
 - `h2 # python3 test/test.py`
 - 如果没有出现`AssertionError`或其他错误，则说明程序实现正确

HTTP服务器分发视频

- 在主机h1上运行http-server，所在目录下有一个小视频（30秒左右）
- 在主机h2上运行vlc（注意切换成普通用户），通过网络获取并播放该小视频
 - 媒体 -> 打开网络串流 -> 网络 -> 请输入网络URL -> 播放

实验要求

- 提交：代码和实验报告
 - 实现越完整越好，测试越充分越好
 - 通过抓包分析，说明HTTP服务器和VLC客户端是如何传输视频文件的

附件文件列表

- dir
- index.html # 主页文件
- keys # 私钥和证书
- Makefile
- test/test.py # （客户端）测试脚本
- topo.py # 测试拓扑

实验概况

实验题目	是否依赖Mininet	是否有基础代码	团队人数上限
BGP分析实验	否	否	2人
BGP前缀劫持攻击和检测实验	是	是	
TCP公平性实验	是	否	
HTTP服务器实验	是	否	

作业在课程网站上提交，**截止时间：11月29日（4周时间）**

在作业提交页面的文本框中，**注明选择题目、团队人员姓名、学号**