LVS SYNPROXY 概要设计

By 陈建 @360

1 背景

LVS 当前的主要功能是负载均衡和失败冗余，不能对 flood 类型的包攻击进行防护，导致攻击流量会被转发到后端 RS 上，而 RS 对此类攻击的防护工作 TCP 层，防护效率不高，同时大流量的包转发也会大大消耗 LVS 机器的 CPU，影响 LVS 的正常工作。

为了解决上述问题，

1. 将在 LVS 上增加一种攻击防护功能：Syn-Proxy，该功能借鉴 TCP 的 syn-cookie 机制，代理 Client 和 RS 之间的 TCP 连接握手过程，在与 Clinet 三次握手完成后，才去和 RS 建立连接，从而实现对 synflood 攻击的防御；
2. 采用“session 不存在直接丢弃”的策略，实现了对 ack/fin/rst 其它 TCP 标志位攻击的防御；
3. 采用“直接丢弃”的策略，实现 UDP/IP FRAG DDOS 攻击的防御；

注：Syn-Proxy 实现较复杂，下文将重点介绍；

2 目标

Syn-Proxy 作为一种攻击防护功能，实现如下功能：

1. 代理与 Client 的三次握手，包括：
   1） Client 的 Syn 报文到来时，采用 TCP 层的 syn-cookie 算法，回复 Syn/Ack 报文；
   2） Client 三次握手的 Ack 报文到来时，利用 syn-cookie 算法进行校验，校验不通过则丢弃报文，校验通过则完成与 Client 的三次握手；
   另外，还实现了 TCP 层的 defer 功能，在开启 defer 功能的情况下，只有客户端发来的 Ack 报文中带有 payload 时，才真正完成与 Client 的三次握手，否则将丢弃该 Ack 报文。
2. 代理与 RS 的三次握手，包括：
   1） 向 RS 发送 Syn 报文；
   2） 收到 RS 的 Syn/Ack 报文后，回复 Ack 报文，并记录 RS Syn/Ack 报文与 LVS 返回的 Syn/Ack 报文的 Seq 之差；
3. 后续报文的序列号修正；

功能要求：Syn-Proxy 工作在 NAT/FULLNAT 转发模式下。

3 总体思路

1. 增加 NF_INET_PRE_ROUTING 处 HOOK 点处理函数 ip_vs_pre_routing，用来处理 Client 发来的 Syn 报文，回复 Syn/Ack;
2. 在 tcp_conn_schedule 函数中，增加对 Ack 报文的 cookie 校验，校验通过创建 session，把 Ack 报文保存在 Session 结构体中，并向 RS 发送 syn 报文；
3. 在 ip_vs_out 函数中，增加对 RS 发来的 Syn/Ack 报文的处理，更新 syn_proxy_seq，并向 RS 发送 session 中保存的 Ack 报文。
4. 序列号处理：
   1） Packet IN 时，在 tcp_dnat_handler/tcp_fnat_in_handler 中，根据 syn_proxy_seq 修正 Ack seq，并修正 Sack 选项中的 Ack seq；
   2） Packet Out 时，在 tcp_snat_handler/tcp_fnat_out_handler 中，根据 syn_proxy_seq 修改 Seq；

4 关键技术点

（注：本次给出的是大部分是 IPV4 的实现，V6 的实现类似）

4.1 Syn Cookie 处理

Syn Cookie 处理包括两个部分：根据收到 Syn 报文，根据 Cookie 算法计算返回的 Syn/Ack Seq；根据收到的 Ack 报文，校验 Ack seq 是否合法；

目前 TCP 层已经实现了 syn-cookie 算法，但是 syn-cookie 和 TCP 的耦合性比较大：

1）cookie 生成算法中需要传入 sock 对象；

2）cookie 校验算法中需要传入 sock 对象，并且会在 cookie 校验函数中生成新到请求的 sock，这个就需要对现有的 cookie 校验接口进行修改，才能实现。

3）cookie 算法中对 Timestamp 等 Opt 的支持函数为 cookie_init_timestamp，该接口需要 request_sock 作为参数，为了使用，也需要对该接口进行修改。

因此，本设计会给出一种新的 Cookie 计算校验方法和接口，该接口可以方便地被 LVS 模块调用，同时为了简便起见，不同于 TCP 层的 Syn Cookie 算法，本设计会把 TCP 选项信息放入回复的序列号中，即 cookie 计算与解析过程中，加入 TCP 选项信息。

为了实现该方法，需要在现有文件中做如下修改：

1、 IPV4 支持：在 net/ipv4/syncookies.c 文件中增加如下函数：
   1） Cookie 计算函数：ip_vs_synproxy_cookie_v4_init_sequence
   2） Cookie 校验函数：ip_vs_synproxy_v4_cookie_check
2、 IPV6 支持：在 net/ipv6/syncookies.c 文件中增加如下函数：
   1） Cookie 计算函数：ip_vs_synproxy_cookie_v6_init_sequence
   2） Cookie 校验函数：ip_vs_synproxy_v6_cookie_check

4.2 三次握手报文的处理

Syn Proxy 需要对三次握手报文的处理包括如下部分：

1） Client 发来的 Syn 报文：尝试复用该 SKB 生成 Syn/Ack 报文；
2） Client 发来的 Ack 报文：根据该报文生成发向 RS 的 Syn 报文，并保存该 Ack 报文；
3） RS 发来的 Syn/Ack 报文：把保存的 Ack 报文发向 RS。

4.2.1 Client 的 Syn 报文处理

在函数 ip_vs_synproxy_syn_rcv 中实现，该函数在 ip_vs_pre_routing 中被调用，并且是主要实现逻辑，如下：

1） 判断是否可以复用 SKB，不行的话就是用 skb_copy 重新创建一个；
2） 设置 TCP option：调用自定义的 syn_proxy_parse_set_opts 函数，处理 Timestamp、Wscale、Sack 等。（具体内容见 4.4）。
3） 计算 Cookie。
4） 设置 TCP 头部标志位（Syn+Ack）、互换源目的 IP、源目的端口、Seq 和 Ack seq 及 ttl 和 tos。
5） 计算 IP 层 checksum；
6） 计算 TCP 层 checksum；
7） 互换源目的 MAC；
8） 发送报文：调用 dev_queue_xmit 函数；

4.2.2 Client 的 Ack 报文处理

在函数 ip_vs_synproxy_ack_rcv 中实现，该函数在 tcp_conn_schedule 中被调用，并在原有 session 创建代码的前面执行，其主要逻辑如下：

1、 验证 cookie 及 TCP OPT：调用 syn_proxy_v4_cookie_check：
2、 选择 RS，并创建 Session，创建成功，则继续处理（具体内容见 4.3）。
3、 向 RS 发送发送 Syn 报文：在 syn_proxy_send_rs_syn 函数中处理。

4.2.2 RS 的 Syn/Ack 报文处理

在函数 ip_vs_synproxy_synack_rcv 中实现，在 handle_response 中被调用，主要逻辑如下：

1. 获取 session 锁；
2. 判断是否是 RS 发来的 Syn/Ack 报文，如果判断成功，则继续；
3. 计算 syn_proxy_seq：
4. 向 RS 发送 Ack 报文：

4.3 TCP OPT 处理

参照函数 tcp_parse_options；

4.4.1 timestamp 等 Option 过滤

和 TCP 层保持一致，读取/proc/net/ipv4/下的 timestamp、wscale 和 sack 的开关，把不支持的内容设置为 NOP。

在函数 syn_proxy_parse_set_opts 中实现，在处理 Client 的 Syn 报文时被调用。

注：当前不支持 timestamp；

4.4.2 调整 MSS

在 Syn Proxy 开启时，LVS 代理与 Client 的三次握手，因此需要在回复的 Syn/Ack 报文中设置 MSS。

为了方便实现，我们在此定义 proc 参数 MSS-sysctl_ip_vs_synproxy_init_mss，作为预设值；

4.4.3 创建 RS Syn 报文 Tcp Opt

在完成与 Client 的三次握手后，LVS 需要向 RS 发送 Syn 报文，此时需要根据 Ack 报文中的 Option 信息，创建 Syn 报文的 Tcp Opt。

在函数 syn_proxy_syn_build_options 中实现，该函数在 syn_proxy_send_rs_syn 中被调用。

4.4.4 Out2In 报文的 Sack Opt 序列号转换

在 Syn Proxy 开启时，LVS 和 RS 回复的 Syn/Ack 报文 Seq 很有可能会不一致，因此在处理 Out-In 报文时，需要转换 Sack Opt 中的序列号。

在 syn_proxy_filter_opt_outin 函数中实现。

4.4 序列号转换

Client-LVS-RS 的六次握手过程，会将 lvs->client 和 rs->lvs 的 seq 相关信息保存到 syn_proxy_seq 中，根据该信息对 Session 后续的报文序列号进行转换，以保证 Client 和 RS 上序列号的正确性。

4.5 Session 复用

在 Syn Proxy 模式下，如果三次握手的 Ack 报文命中 LVS 的已有 session（Session 可能处理 IP_VS_TCP_S_TIME_WAIT、IP_VS_TCP_S_CLOSE 等状态），则 Ack 报文会被简单的转发到后端的 RS，LVS 不会完成六次握手过程，导致该请求不能被正确的处理。因此，需要增加 Session 复用功能，对命中 Session 的 Ack 报文进行特殊处理。

Session 复用主要实现的功能如下：

1）从 Client 过来的 Ack 报文数据包在 Session 命中的情况，判断 Session 当前的状态，并验证报文的标志位及 Cookie 校验等，如果验证通过，则重新设置 Session 的状态及 syn_proxy_seq 等，并向 RS 发送 Syn 报文，后续的处理不需要修改；

2）支持 CLOSE、TIME_WAIT、FIN_WAIT、CLOSE_WAIT 和 LAST_ACK 状态的复用，并且对每个状态都提供单独的 PROC 开关来决定是否进行复用；

3）提供一个 PROC 开关来控制该功能是否开启。

该功能由 ip_vs_synproxy_reuse_conn 实现。

4.6 向 RS 发送的 SYN 报文重传

在 Syn Proxy 模式下，为了完成六次握手，LVS 会向 RS 发送 Syn 报文，但没进行 Syn 报文的重传机制，会导致发送 Syn 报文丢弃时，该连接不能正常被处理。因此，需要添加 Syn 报文重传机制。

重传机制实现思想：当 session 释放时，检查 Session 所处状态，如果为 IP_VS_TCP_S_SYN_SENT 状态，表明 LVS 与 RS 间的握手过程没有完成，于是进行判断，如果重传功能开启并且尚未达到重传次数的阈值，则将该 Session 保存的 syn 报文重新发送。

4.6 其他技术点

4.6.1 RS 回复的 RST 报文序列号修正

在 LVS 与 RS 进行三次握手时，如果 RS 不回复 Syn/Ack 报文，而是回复 RST 报文，LVS 也需要进行正确的序列号处理，以便 Client 能够正确地释放 Socket。

实现思路：

在 ip_vs_synproxy_synack_rcv 函数中，如果判断 RS 回复的为 RST 报文，则同样计算 syn_proxy_seq.delta，并设置 Session 状态为 IP_VS_TCP_S_CLOSE，同时更新 Session 超时时间。

4.6.2 Defer 功能实现

TCP 协议栈的 Defer 功能（当第三次握手的 Ack 报文带有数据时，才建立连接）可有效防护慢连接攻击。在 Syn Proxy 基础上，可以很方便地实现该功能。

实现思路：

在 syn_proxy_ack_rcv 函数中，进行 Cookie 校验前，先判断该开关，如果开关开启，则判断该 Ack 报文是否有 Payload，有才建立连接。

4.6.3 GET 请求超过一个报文的处理

LVS 向 RS 发送 Syn 报文后，会等待 RS 发来的 Syn/Ack 报文，在此期间，会丢掉所有从 Client 发来的报文，当 GET 请求超过一个报文时，会导致报文被丢弃，影响请求处理的时间，为此，需要缓存这些报文。

4.7 控制接口

Syn Proxy 增加了 syn proxy 开关 1 个配置；

1. syn proxy 开关
   a) 该信息维护在 svc->flags 中，因此，需要在 ip_vs.h 头文件中增加
      #define IP_VS_CONN_F_SYNPROXY 0x4000 /* syn proxy */
   b) 需要修改 keepalived 和 ipvsadm 程序

5 其它因素考虑

5.1 对于其它模块的影响

Template/persistent/ipv6/UDP/ICMP 各个模块的影响，处理可能考虑不全；

5.2 测试考虑

1. 测试 NAT/DR/TUNNEL 兼容性
2. 测试 TCP/UDP/ICMP/AH/ESP 数据包
3. 测试 IPV6
4. 测试 Template/persistent