**基于netfilter修改HTTP数据包（插入、修改、删除）**

1. **设计实现**
2. 概述：

测试内核：3.13.0-32-generic（ubuntu14.04/centos7）

网关服务器上的内核模块，实现功能：

* 修改经过网关的HTTP数据包，修改前后数据长度可变。

tcp数据包：next\_seq = 当前seq + 当前数据包长度

由于修改数据包长度后后续tcp连接seq以及ack都会发生变化，需要调用nf\_conntrack会话跟踪模块中的方法跟踪修改整个tcp连接后续的seq和ack。

模块基于netfilter hook编写，在NF\_INET\_FORWARD数据包转发处挂在模块，hook点信息：

{

.hook = hook\_func,

.pf = NFPROTO\_IPV4,

.hooknum = NF\_INET\_FORWARD,

.priority = NF\_IP\_PRI\_MANGLE,

.owner = THIS\_MODULE

},

1. 主要函数接口：

/\* \*

\* Generic function for mangling variable-length address changes inside

\* NATed TCP connections (like the PORT XXX,XXX,XXX,XXX,XXX,XXX

\* command in FTP).

\* \*/

static inline int nf\_nat\_mangle\_tcp\_packet(

struct sk\_buff \*skb,

struct nf\_conn \*ct,

enum ip\_conntrack\_info ctinfo,

// IP头偏移量

unsigned int protoff,

// 插入点偏移量（距离payload首部）

unsigned int match\_offset,

// 插入点覆盖长度

unsigned int match\_len,

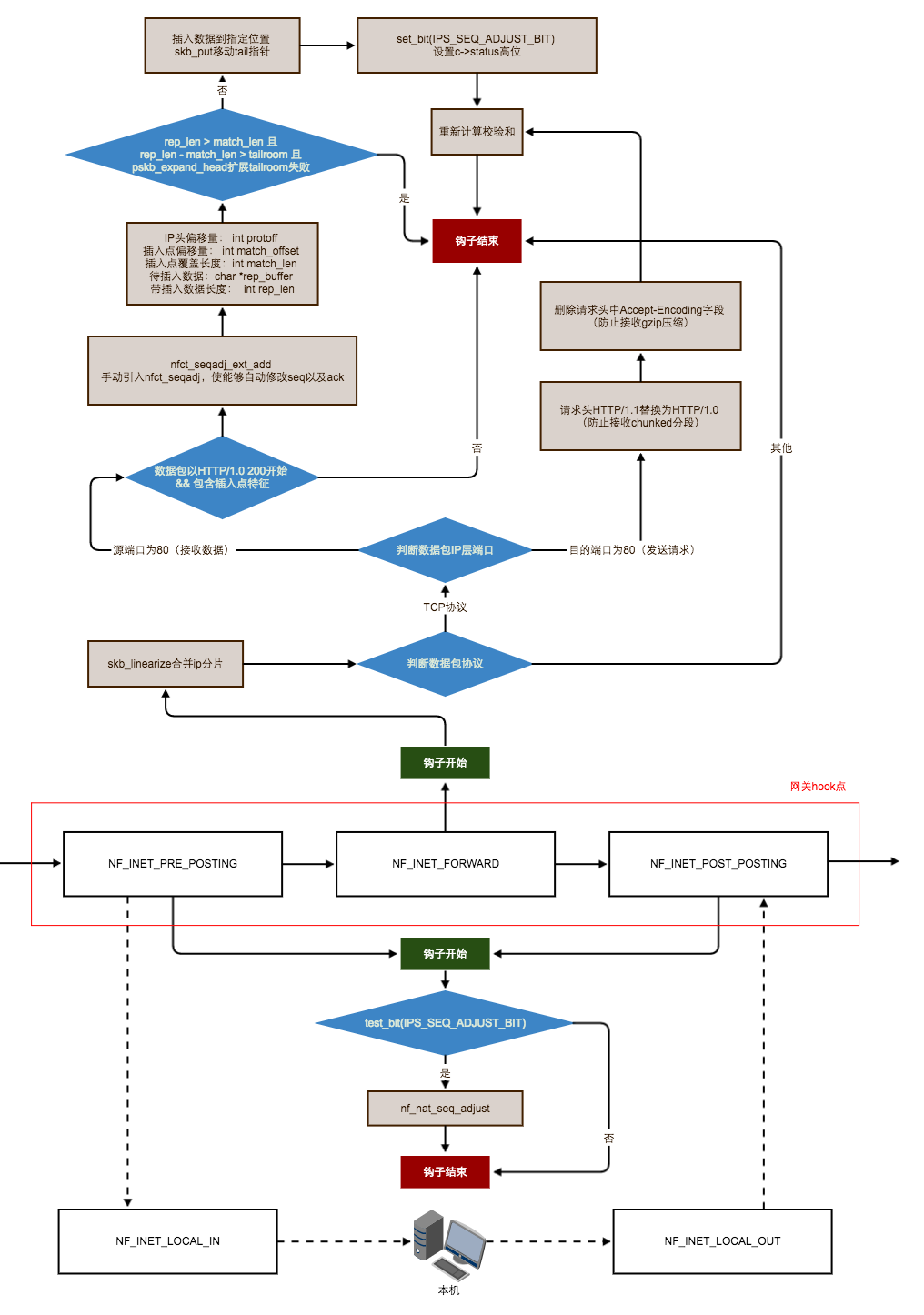
// 待插入数据

const char \*rep\_buffer,

// 带插入数据长度

unsigned int rep\_len)

1. 处理流程：



1. **注意事项**
2. 在调用nf\_nat\_mangle\_tcp\_packet前需要调用nfct\_seqadj\_ext\_add将当前连接跟踪信息加入seq修正。
3. 修改数据包长度后3.13内核可自动修改seq和ack

原理：nf\_nat\_mangle\_tcp\_packet时会做以下处理set\_bit(IPS\_SEQ\_ADJUST\_BIT, &ct->status)

后续内核跟踪钩子会修正test\_bit(IPS\_SEQ\_ADJUST\_BIT, &ct->status)的tcp数据包的seq和ack）

1. 内核版本3.2以及2.68以下只是在ct->status中高几位做了标记（IPS\_SEQ\_ADJUST\_BIT），还需要后续手动hook修改。

unsigned int fix\_seq(unsigned int hooknum, struct sk\_buff \*skb,

const struct net\_device \*in, const struct net\_device \*out, int(\*okfn)(struct sk\_buff \*))

{

enum ip\_conntrack\_info ctinfo;

struct nf\_conn \*ct = nf\_ct\_get(skb, &ctinfo);

if (ct && test\_bit(IPS\_SEQ\_ADJUST\_BIT, &ct->status)

&& (ctinfo != IP\_CT\_RELATED + IP\_CT\_IS\_REPLY) ) {

nf\_nat\_seq\_adjust(skb, ct, ctinfo);

}

return NF\_ACCEPT;

}

// 3.2以及2.68以下内核适用

{

.hook = fix\_seq,

.pf = PF\_INET,

.hooknum = NF\_INET\_PRE\_ROUTING,

.priority = NF\_IP\_PRI\_CONNTRACK\_CONFIRM,

.owner = THIS\_MODULE

},

{

.hook = fix\_seq,

.pf = PF\_INET,

.hooknum = NF\_INET\_POST\_ROUTING,

.priority = NF\_IP\_PRI\_CONNTRACK\_CONFIRM,

.owner = THIS\_MODULE

},

1. 处理gzip压缩：发送请求时删除请求头中Accept-Encoding字段，防止收到gzip压缩包
2. 处理chunked分段传输：发送请求时将HTTP/1.1修改为HTTP/1.0防止收到chunked数据包（Transfer-Encoding： chunked是HTTP 1.1中特有的）
3. 对数据包进行修改需要重新计算checksum
4. 处理完整数据包需要合并ip分片

// IP数据包frag合并

if (0 != skb\_linearize(skb)) {

return NF\_ACCEPT;

}

1. **使用方法**

* 查看模块信息

# modinfo hook\_ipv4.ko

* 加载模块

# echo 1 > /proc/sys/net/ipv4/ip\_forward

# iptables -t nat -A POSTROUTING -j MASQUERADE

# cd hook\_ipv4

# make

# insmod hook\_ipv4.ko

* 卸载模块

# rmmod hook\_ipv4

* 查看日志

# tail /var/log/kern.log（ubuntu）

# tail /var/log/message.log（centos）