**ARP欺骗攻击**

由于局域网的网络流通不是根据IP地址进行，而是根据MAC地址进行传输。所以，MAC地址在A上被伪造成一个不存在的MAC地址，这样就会导致网络不通，A不能Ping通C！这就是一个简单的ARP欺骗。

ARP攻击主要是存在于局域网中，通过伪造IP地址和MAC地址实现ARP欺骗，能够在网络中产生大量的ARP通信量使网络阻塞，攻击者只要持续不断的发出伪造的ARP响应包就能更改目标主机ARP缓存中的IP-MAC条目，造成网络中断或中间人攻击。

解决措施

一、双绑措施  
  双绑是在路由器和终端上都进行IP-MAC绑定的措施，它可以对**ARP**欺骗的两边，伪造网关和截获数据，都具有约束的作用。这是从**ARP**欺骗原理上进行的防范措施，也是最普遍应用的办法。它对付最普通的**ARP**欺骗是有效的

二、**ARP**个人防火墙  
  在一些杀毒软件中加入了**ARP**个人防火墙的功能，它是通过在终端电脑上对网关进行绑定，保证不受网络中假网关的影响，从而保护自身数据不被窃取的措施。**ARP**防火墙使用范围很广，有很多人以为有了防火墙，**ARP**攻击就不构成威胁了，其实完全不是那么回事。

三、VLAN和交换机端口绑定  
  通过划分VLAN和交换机端口绑定，以图防范**ARP**，也是常用的防范方法。做法是细致地划分VLAN，减小广播域的范围，使**ARP**在小范围内起作用，而不至于发生大面积影响。同时，一些网管交换机具有MAC地址学习的功能，学习完成后，再关闭这个功能，就可以把对应的MAC和端口进行绑定，避免了病毒利用**ARP**攻击篡改自身地址。也就是说，把**ARP**攻击中被截获数据的风险解除了。这种方法确实能起到一定的作用。

四、PPPoE  
网络下面给每一个用户分配一个帐号、密码，上网时必须通过PPPoE认证，这种方法也是防范**ARP**措施的一种。PPPoE拨号方式对封包进行了二次封装，使其具备了不受**ARP**欺骗影响的使用效果，很多人认为找到了解决**ARP问题**的终极方案。

5.DHCP协议的交互过程。单个和多个DHCP server的区别。存在DHCP relay的情况

交互过程：

1. 新的client加入网络时，会使用0.0.0.0作为源地址，发送discover广播报文，查询网络上有哪些DHCP server，以及这些DHCP server 能Offer哪些IP地址   
   2. DHCP服务器接收到DHCP Discover报文后，回应Offer报文，提供IP地址（可能包含DNS等其他信息）给client   
   3. client 根据收到的Offer报文，选择一个DHCP server，并选择它提供的IP地址。然后广播Request报文，想DHCP Server请求该IP地址，同时想本地网络（尤其是其他DHCP Server）公告自己已经选择了某个DHCP Server的某个IP地址。   
   4. DHCP Server 回应ACK报文，将IP地址分配给Client端 （特殊情况：DHCP Server在发送Offer报文和接收到Request的短暂时间内把IP分配给了其他主机）   
   5. DHCP Client 收到ACK报文后，会针对获得的IP地址发送ARP Request，进行IP地址冲突检测。   
   6. 如果IP地址已经被其他主机使用，则Client放弃该IP地址，想Server发送DHCP DECLINE报文告诉Server该地址不能使用。然后一段时间后（一般10s）再此尝试获取该IP地址   
   7. 如果Client仍然无法使用该IP地址，则发送DHCP RELEASE报文，放弃该地址

(1)Discover： Client在以广播的形式DHCP Discover报文(DstMAC和DstIP都是全1，SrcIP全0，SrcMAC是Client的MAC)，广播域内的所有Server都会收到该报文。另外，Client可以在选项字段中加入“request paramter list” 选项，表明自己想要获得的各种参数。  
�6�8�6�8(2)Offer：广播域内的Server收到DHCP Discover报文之后，会发送DHCP Offer报文，报文中包含了Server准备分配给Client的IP地址，以及一些选项字段。  
�6�8�6�8(3)Request：如果广播域内有多个Server，那么Client会收到多个DHCP Offer报文。Client只能选择处理其中一个，一般的做法是选择最先收到的DHCP Offer报文。**然后Client就以广播方式回答一个DHCP request请求信息，该信息中包含Client选中的DHCP Server的IP地址和Client想要的IP地址。**之所以要以广播方式回答，是为了通知所有的Server，Client将选择某台Server所提供的IP地址。  
�6�8�6�8(4)ACK：Server收到DHCP Request报文后，判断选项字段中的DHCP Server的IP地址是否与自己的地址相同。如果不相同就不回应ACK报文，并且判断自己是否给该Request报文的Client发送过DHCP Offer报文。如果发送过DHCP Offer报文，则清除相应IP地址记录。如果选项字段中的DHCP Server的IP地址与自己的IP地址相同，Server就会响应一个DHCP ACK报文，其内容同DHCP Offer类似并在选项字段中增加了IP地址使用租期选项。Client收到ACK报文后，如果Server分配的IP地址不跟其他设备的IP地址冲突，那么Client将使用它作为自己的IP。

**SYN Flood 攻击原理及防御**

SYN Flood是当前最流行的DoS（拒绝服务攻击）与DDoS（分布式拒绝服务攻击）的方式之一，这是一种利用**TCP**协议缺陷，发送大量伪造的**TCP**连接请求，从而使得被攻击方资源耗尽（CPU满负荷或内存不足）的攻击方式。

原理：  
问题出在**TCP**连接的三次握手中，恶意的攻击者大量发送SYN报文，服务器端将为了维护一个非常大的半连接列表而消耗非常多的资源----数以万计的半连接，即使是简单的保存并遍历也会消耗非常多的CPU时间和内存，何况还要不断对这个列表中的**IP**进行SYN+ACK的重试。实际上如果服务器的**TCP**/**IP**栈不够强大，最后的结果往往是堆栈溢出崩溃---即使服务器端的系统足够强大，服务器端也将忙于处理攻击者伪造的**TCP**连接请求而无暇理睬客户的正常请求（毕竟客户端的正常请求比率非常之小），此时从正常客户的角度看来，服务器失去响应，这种情况我们称作：服务器端受到了SYN Flood攻击（SYN洪水攻击）。  
攻击方式：

1. Direct Attack 攻击方使用固定的源地址发起攻击，这种方法对攻击方的消耗最小
2. Spoofing Attack 攻击方使用变化的源地址发起攻击，这种方法需要攻击方不停地修改源地址，实际上消耗也不大
3. Distributed Direct Attack 这种攻击主要是使用僵尸网络进行固定源地址的攻击

防御方法：

1. 无效连接监视释放  
   不停监视系统的半开连接和不活动连接，当达到一定阈值时拆除这些连接，从而释放系统资源。
2. 延缓TCB分配方法  
   从前面SYN Flood原理可以看到，消耗服务器资源主要是因为当SYN数据报文一到达，系统立即分配TCB，从而占用了资源。而SYN Flood由于很难建立起正常连接，因此，当正常连接建立起来后再分配TCB则可以有效地减轻服务器资源的消耗。常见的方法是使用Syn Cache和Syn Cookie技术。  
   Syn Cache技术：  
   这种技术是在收到SYN数据报文时不急于去分配TCB，而是先回应一个SYN ACK报文，并在一个专用HASH表（Cache）中保存这种半开连接信息，直到收到正确的回应ACK报文再分配TCB。  
   Syn Cookie技术：  
   Syn Cookie技术则完全不使用任何存储资源，它使用一种特殊的算法生成Sequence Number，这种算法考虑到了对方的**IP**、端口、己方**IP**、端口的固定信息，以及对方无法知道而己方比较固定的一些信息，如MSS、时间等，在收到对方的ACK报文后，重新计算一遍，看其是否与对方回应报文中的（Sequence Number-1）相同，从而决定是否分配TCB资源。
3. 使用SYN Proxy防火墙  
   防火墙中提供一种SYN代理的功能，主要原理是对试图穿越的SYN请求进行验证后才放行。

#### 为什么TCP建立连接要三次握手

首先得回答三次握手的目的是同步连接双方的序列号和确认号并交换 **TCP** 窗口大小信息。  
然后可以回答为什么两次握手不行，两次握手可能因为丢包而出现死锁，假设在两次握手场景中，C向S发送请求，S收到并发送确认请求给C，这时候S认为连接已经建立，并开始发送数据给C，但是那个确认请求丢包了，C不认为请求建立了，C当然会拒绝接受S发送来的数据，并且再去请求连接。这样，一个资源就死锁了。  
最后回答握手当然可以四次五次一直握下去，但三次已经够了，就没有必要了。  
总结下来一句话，主要目的防止在网络发生延迟或者丢包的情况下浪费资源。

**DoS具有代表性的攻击手段包括PingofDeath**

**[](https://baike.baidu.com/pic/dos%E6%94%BB%E5%87%BB/3792374/0/62667cd0781718c2a0ec9cc7?fr=lemma&ct=single)dos攻击快闪族**

**、TearDrop、UDPflood、SYNflood、LandAttack、IPSpoofingDoS**

CP协议作为一个可靠的面向流的传输协议，其可靠性和流量控制由滑动窗口协议保证，而拥塞控制则由控制窗口结合一系列的控制算法实现。  
一、滑动窗口协议  
     关于这部分自己不晓得怎么叙述才好，因为理解的部分更多，下面就用自己的理解来介绍下TCP的精髓：滑动窗口协议。  
     所谓滑动窗口协议，自己理解有两点：1. “窗口”对应的是一段可以被发送者发送的字节序列，其连续的范围称之为“窗口”；2. “滑动”则是指这段“允许发送的范围”是可以随着发送的过程而变化的，方式就是按顺序“滑动”。在引入一个例子来说这个协议之前，我觉得很有必要先了解以下前提：  
-1. TCP协议的两端分别为发送者A和接收者B，由于是全双工协议，因此A和B应该分别维护着一个独立的发送缓冲区和接收缓冲区，由于对等性（A发B收和B发A收），我们以A发送B接收的情况作为例子；  
-2. 发送窗口是发送缓存中的一部分，是可以被TCP协议发送的那部分，其实应用层需要发送的所有数据都被放进了发送者的发送缓冲区；  
-3. 发送窗口中相关的有四个概念：已发送并收到确认的数据（不再发送窗口和发送缓冲区之内）、已发送但未收到确认的数据（位于发送窗口之中）、允许发送但尚未发送的数据以及发送窗口外发送缓冲区内暂时不允许发送的数据；  
-4. 每次成功发送数据之后，发送窗口就会在发送缓冲区中按顺序移动，将新的数据包含到窗口中准备发送；  
     TCP建立连接的初始，B会告诉A自己的接收窗口大小，比如为‘20’：  
     字节31-50为发送窗口  
  
     A发送11个字节后，发送窗口位置不变，B接收到了乱序的数据分组：  
  
     只有当A成功发送了数据，即发送的数据得到了B的确认之后，才会移动滑动窗口离开已发送的数据；同时B则确认连续的数据分组，对于乱序的分组则先接收下来，避免网络重复传递：

二、流量控制  
     流量控制方面主要有两个要点需要掌握。一是TCP利用滑动窗口实现流量控制的机制；二是如何考虑流量控制中的传输效率。  
1. 流量控制  
     所谓流量控制，主要是接收方传递信息给发送方，使其不要发送数据太快，是一种端到端的控制。主要的方式就是返回的ACK中会包含自己的接收窗口的大小，并且利用大小来控制发送方的数据发送：  
  
     这里面涉及到一种情况，如果B已经告诉A自己的缓冲区已满，于是A停止发送数据；等待一段时间后，B的缓冲区出现了富余，于是给A发送报文告诉A我的rwnd大小为400，但是这个报文不幸丢失了，于是就出现A等待B的通知||B等待A发送数据的死锁状态。为了处理这种问题，TCP引入了持续计时器（Persistence timer），当A收到对方的零窗口通知时，就启用该计时器，时间到则发送一个1字节的探测报文，对方会在此时回应自身的接收窗口大小，如果结果仍未0，则重设持续计时器，继续等待。  
2. 传递效率  
     一个显而易见的问题是：单个发送字节单个确认，和窗口有一个空余即通知发送方发送一个字节，无疑增加了网络中的许多不必要的报文（请想想为了一个字节数据而添加的40字节头部吧！），所以我们的原则是尽可能一次多发送几个字节，或者窗口空余较多的时候通知发送方一次发送多个字节。对于前者我们广泛使用Nagle算法，即：  
\*1. 若发送应用进程要把发送的数据逐个字节地送到TCP的发送缓存，则发送方就把第一个数据字节先发送出去，把后面的字节先缓存起来；  
\*2. 当发送方收到第一个字节的确认后（也得到了网络情况和对方的接收窗口大小），再把缓冲区的剩余字节组成合适大小的报文发送出去；  
\*3. 当到达的数据已达到发送窗口大小的一半或以达到报文段的最大长度时，就立即发送一个报文段；  
     对于后者我们往往的做法是让接收方等待一段时间，或者接收方获得足够的空间容纳一个报文段或者等到接受缓存有一半空闲的时候，再通知发送方发送数据。

、拥塞控制  
     网络中的链路容量和交换结点中的缓存和处理机都有着工作的极限，当网络的需求超过它们的工作极限时，就出现了拥塞。拥塞控制就是防止过多的数据注入到网络中，这样可以使网络中的路由器或链路不致过载。常用的方法就是：  
1. 慢开始、拥塞控制  
2. 快重传、快恢复  
     一切的基础还是慢开始，这种方法的思路是这样的：  
-1. 发送方维持一个叫做“拥塞窗口”的变量，该变量和接收端口共同决定了发送者的发送窗口；  
-2. 当主机开始发送数据时，避免一下子将大量字节注入到网络，造成或者增加拥塞，选择发送一个1字节的试探报文；  
-3. 当收到第一个字节的数据的确认后，就发送2个字节的报文；  
-4. 若再次收到2个字节的确认，则发送4个字节，依次递增2的指数级；  
-5. 最后会达到一个提前预设的“慢开始门限”，比如24，即一次发送了24个分组，此时遵循下面的条件判定：  
\*1. cwnd < ssthresh， 继续使用慢开始算法；  
\*2. cwnd > ssthresh，停止使用慢开始算法，改用拥塞避免算法；  
\*3. cwnd = ssthresh，既可以使用慢开始算法，也可以使用拥塞避免算法；  
-6. 所谓拥塞避免算法就是：每经过一个往返时间RTT就把发送方的拥塞窗口+1，即让拥塞窗口缓慢地增大，按照线性规律增长；  
-7. 当出现网络拥塞，比如丢包时，将慢开始门限设为原先的一半，然后将cwnd设为1，执行慢开始算法（较低的起点，指数级增长）；  
  
     上述方法的目的是在拥塞发生时循序减少主机发送到网络中的分组数，使得发生拥塞的路由器有足够的时间把队列中积压的分组处理完毕。慢开始和拥塞控制算法常常作为一个整体使用，而快重传和快恢复则是为了减少因为拥塞导致的数据包丢失带来的重传时间，从而避免传递无用的数据到网络。快重传的机制是：  
-1. 接收方建立这样的机制，如果一个包丢失，则对后续的包继续发送针对该包的重传请求；  
-2. 一旦发送方接收到三个一样的确认，就知道该包之后出现了错误，立刻重传该包；  
-3. 此时发送方开始执行“快恢复”算法：  
\*1. 慢开始门限减半；  
\*2. cwnd设为慢开始门限减半后的数值；  
\*3. 执行拥塞避免算法（高起点，线性增长）；

### 2.2 TCP传输阶段

#### 2.2.1 滑动窗口以及拥塞控制

**TCP**协议作为一个可靠的面向流的传输协议，其可靠性和流量控制由滑动窗口协议保证，而拥塞控制则由控制窗口结合一系列的控制算法实现。

详情请移步 [**tcp**窗口滑动以及拥塞控制](http://www.cnblogs.com/woaiyy/p/3554182.html)

### 2.3 TCP断开阶段

#### 2.3.1 设置TIME\_WAIT的原因

1. 可靠地实现**TCP**全双工连接的终止  
   **TCP**协议在关闭连接的四次握手过程中，最终的ACK是由主动关闭连接的一端（后面统称A端）发出的，如果这个ACK丢失，对方（后面统称B端）将重发出最终的FIN，因此A端必须维护状态信息（TIME\_WAIT）允许它重发最终的ACK。如果A端不维持TIME\_WAIT状态，而是处于CLOSED 状态，那么A端将响应RST分节，B端收到后将此分节解释成一个错误（在java中会抛出connection reset的SocketException)。  
   因而，要实现**TCP**全双工连接的正常终止，必须处理终止过程中四个分节任何一个分节的丢失情况，主动关闭连接的A端必须维持TIME\_WAIT状态
2. 允许老的重复分节在网络中消逝  
   **TCP**分节可能由于路由器异常而“迷途“，在迷途期间，**TCP**发送端可能因确认超时而重发这个分节，迷途的分节在路由器修复后也会被送到最终目的地，这个迟到的迷途分节到达时可能会引起问题。在关闭“前一个连接”之后，马上又重新建立起一个相同的**IP**和端口之间的“新连接”，“前一个连接”的迷途重复分组在“前一个连接”终止后到达，而被“新连接”收到了。为了避免这个情况，**TCP**协议不允许处于TIME\_WAIT状态的连接启动一个新的可用连接，因为TIME\_WAIT状态持续2MSL，就可以保证当成功建立一个新**TCP**连接的时候，来自旧连接重复分组已经在网络中消逝。

#### 2.3.2 大量 TIME\_WAIT 产生的原因及解决办法

原因：对于基于**TCP**的HTTP协议，关闭**TCP**连接的是Server端，这样，Server端会进入TIME\_WAIT状态，可想而知，对于访问量大的Web Server，会存在大量的TIME\_WAIT状态。

解决办法：

1. 开启socket重用，允许将TIME\_WAIT的socket重新用于**TCP**连接
2. 开启快速回收。

#### 2.3.3 大量CLOSE\_WAIT产生的原因及解决办法

原因：对方关闭连接之后服务器程序自己没有进一步发出ack信号。换句话说，就是在对方连接 关闭之后，程序里 没有检测到，或者程序压根就忘记了这个时候需要关闭连接，于是这个资源就一直被程序占着。  
解决办法：  
具体问题具体解决，总结一句话就是在处理资源时一定要记住自己申请的资源要记得主动释放。  
详情请移步 [再谈应用环境下的TIME\_WAIT和CLOSE\_WAIT](http://blog.csdn.net/shootyou/article/details/6622226)

如何处理广播风暴？  
(1)什么是广播风暴：网络上的广播帧（由于被转发）数量急剧增加而影响正常的网络通讯的反常现象，广播风暴会占用相当可观的网络带宽，造成整个网络无法正常工作。  
(2)广播风暴只会在二层出现，广播报文跨越不了网段。  
(3)广播风暴的产生有几个可能的原因：网卡损坏、网络环路、网络病毒。  
(4)广播风暴的避免/抑制/解决方法：  
a)开启交换机端口的广播风暴控制，当端口收到的广播帧累计到预定门限值时，端口将自动丢弃收到的广播帧。  
b)vlan隔离：按照端口/MAC/子网划分vlan，可以将广播风暴的影响范围限制在vlan内。

c)交换机开启生成树协议(STP/RSTP/MSTP)，可以避免网络环路造成的广播风暴。  
8d)开启安全防护软件，可以减少网络病毒造成的广播风暴。