# DDOS 定义

服务：系统提供的，用户在对其使用中会受益的功能

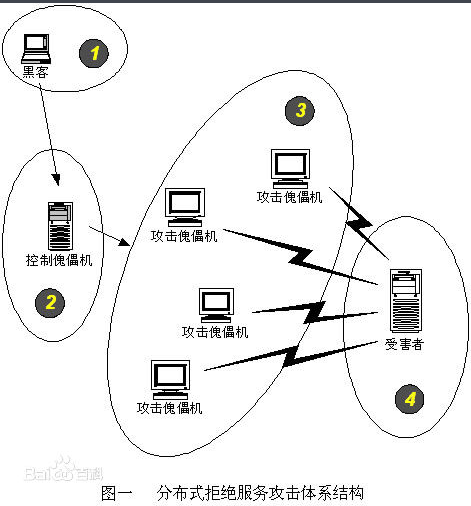
拒绝服务：任何对服务的干涉如果使其可用性降低或者失去可用性均称为拒绝服务。

拒绝服务攻击：是指攻击者通过某种手段，有意地造成计算机或网络不能正常运转从而不能向合法用户提供所需要的服务或者使得服务质量降低。

在信息安全的三要素：“保密性”、“完整性”和“可用性”中，DoS（Denial of Service），即拒绝服务攻击，针对的目标正是“可用性”。该攻击方式利用目标系统网络服务功能缺陷或者直接消耗其系统资源，使得该目标系统无法提供正常的服务。

DDoS：Distributed Denial of Service，即分布式拒绝服务攻击。处于不同位置的多个攻击者同时向一个或者数个目标发起攻击，或者一个或多个攻击者控制了位于不同位置的多台机器并利用这些机器对受害者同时实施攻击，由于攻击的发出点是分布在不同地方的，这类攻击称为分布式拒绝服务攻击。

借助于客户/服务器技术，将多个计算机联合起来作为攻击平台，对一个或多个目标发动DDoS攻击，从而成倍地提高拒绝服务攻击的威力。通常，攻击者使用一个偷窃帐号将DDoS主控程序安装在一个计算机上，在一个设定的时间主控程序将与大量代理程序通讯，代理程序已经被安装在网络上的许多计算机上。代理程序收到指令时就发动攻击。利用客户/服务器技术，主控程序能在几秒钟内激活成百上千次代理程序的运行。



# 攻击流程

DDoS并不象入侵一台主机那样简单。了解这些原理之后，你便会更加明白入侵者的意图，从此便掌握了预防的技巧。一般来说，黑客进行DDoS攻击时会经过这样的步骤：

## 搜集资料

下列情况是黑客非常关心的情报：被攻击目标主机数目、地址情况 目标主机的配置、性能 目标的宽带。

对于DDoS攻击者来说，攻击互联网上的某个站点，有一个重点就是确定到底有多少台主机在支持这个站点，一个大的网站可能有很多台主机利用负载均衡技术提供同一个网站的www服务。

如果要进行DDoS攻击的话，应该攻击哪一个地址呢？使这台机器瘫痪，但其他的主机还是能向外提供www服务，所以想让别人访问不到网站的话，要所有这些IP地址的机器都瘫掉才行。在实际的应用中，一个IP地址往往还代表着数台机器：网站维护者使用了四层或七层交换机来做负载均衡，把对一个IP地址的访问以特定的算法分配到下属的每个主机上去。这时对于DDoS攻击者来说情况就更复杂了，他面对的任务可能是让几十台主机的服务都不正常。

所以说事先搜集情报对DDoS攻击者来说是非常重要的，这关系到使用多少台傀儡机才能达到效果的问题。简单地考虑一下，在相同的条件下，攻击同一站点的2台主机需要2台傀儡机的话，攻击5台主机可能就需要5台以上的傀儡机。有人说做攻击的傀儡机越多越好，不管你有多少台主机我都用尽量多的傀儡机来攻就是了，反正傀儡机超过了时候效果更好。

但在实际过程中，有很多黑客并不进行情报的搜集而直接进行DDoS的攻击，这时候攻击的盲目性就很大了，效果如何也要靠运气。其实做黑客也象网管员一样，是不能偷懒的。一件事做得好与坏，态度最重要，水平还在其次。

## 占领

黑客最感兴趣的是有下列情况的主机：网络状态好的主机 性能好的主机 安全管理水平差的主机

这一部分实际上是使用了另一大类的攻击手段：利用形攻击。这是和DDoS并列的攻击方式。简单地说，就是占领和控制被攻击的主机。取得最高的管理权限，或者至少得到一个有权限完成DDoS攻击任务的帐号。对于一个DDoS攻击者来说，准备好一定数量的傀儡机是一个必要的条件，下面说一下他是如何攻击并占领它们。

首先，黑客做的工作一般是扫描，随机地或者是有针对性地利用扫描器去发现互联网上那些有漏洞的机器，像程序的溢出漏洞、cgi、Unicode、ftp、数据库漏洞…(简直举不胜举啊)，都是黑客希望看到的扫描结果。随后就是尝试入侵了，具体的手段就不在这里多说了，感兴趣的话网上有很多关于这些内容的文章。

总之黑客占领了一台傀儡机了！然后他做什么呢？除了上面说过留后门擦脚印这些基本工作之外，他会把DDoS攻击用的程序上载过去，一般是利用ftp。在攻击机上，会有一个DDoS的发包程序，黑客就是利用它来向受害目标发送恶意攻击包的。

## 实际攻击

经过前2个阶段的精心准备之后，黑客就开始瞄准目标准备发射了。前面的准备做得好的话，实际攻击过程反而是比较简单的。就象图示里的那样，黑客登录到做为控制台的傀儡机，向所有的攻击机发出命令："预备~ ，瞄准~，开火!"。这时候埋伏在攻击机中的DDoS攻击程序就会响应控制台的命令，一起向受害主机以高速度发送大量的数据包，导致它死机或是无法响应正常的请求。黑客一般会以远远超出受害方处理能力的速度进行攻击，他们不会"怜香惜玉"。

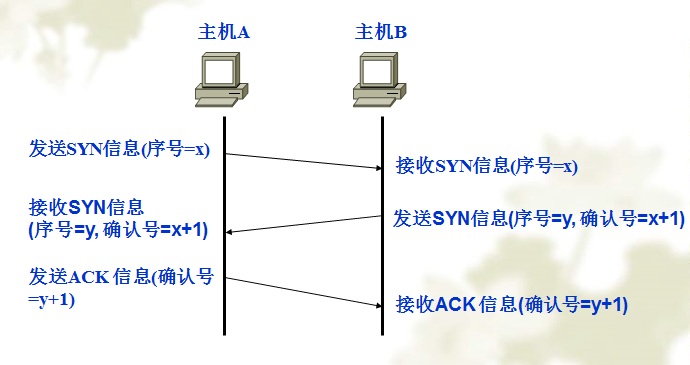
老道的攻击者一边攻击，还会用各种手段来监视攻击的效果，在需要的时候进行一些调整。简单些就是开个窗口不断地ping目标主机，在能接到回应的时候就再加大一些流量或是再命令更多的傀儡机来加入攻击。

# DDOS分类

## 网络层攻击

### Syn-flood

SYN攻击属于DDoS攻击的一种，它利用TCP协议缺陷，通过发送大量的半连接请求，耗费CPU和内存资源。SYN攻击除了能影响主机外，还可以危害路由器、防火墙等网络系统，事实上SYN攻击并不管目标是什么系统，只要这些系统打开TCP服务就可以实施。服务器接收到连接请求（syn= x），将此信息加入未连接队列，并发送请求包给客户（syn=y,ack=x+1），此时进入SYN\_RECV状态。当服务器未收到客户端的确认包时，重发请求包，一直到超时，才将此条目从未连接队列删除。配合IP欺骗，SYN攻击能达到很好的效果，通常，客户端在短时间内伪造大量不存在的IP地址，向服务器不断地发送syn包，服务器回复确认包，并等待客户的确认，由于源地址是不存在的，服务器需要不断的重发直至超时，这些伪造的SYN包将长时间占用未连接队列，正常的SYN请求被丢弃，目标系统运行缓慢，严重者引起网络堵塞甚至系统瘫痪。



网上有一些加固的方法，例如调整内核参数的方法，可以减少等待及重试，加速资源释放，在小流量syn-flood的情况下可以缓解，但流量稍大时完全不抵用。防御syn-flood的常见方法有：syn proxy、syn cookies、首包（第一次请求的syn包）丢弃等。

### ACK-flood

对于虚假的ACK包，目标设备会直接回复RST包丢弃连接，所以伤害值远不如syn-flood。DDOS的一种原始方式。

ACK Flooding攻击是在TCP连接建立之后，所有的数据传输TCP报文都是带有ACK标志位的，主机在接收到一个带有ACK标志位的数据包的时候，需要检查该数据包所表示的连接四元组是否存在，如果存在则检查该数据包所表示的状态是否合法，然后再向应用层传递该数据包。如果在检查中发现该数据包不合法，例如该数据包所指向的目的端口在本机并未开放，则主机操作系统协议栈会回应RST包告诉对方此端口不存在。

这里，服务器要做两个动作：查表、回应 ACK/RST。这种攻击方式显然没有SYN Flood给服务器带来的冲击大，因此攻击者一定要用大流量ACK小包冲击才会对服务器造成影响。按照我们对TCP协议的理解，随机源IP的ACK小包应该会被Server很快丢弃，因为在服务器的TCP堆栈中没有这些ACK包的状态信息。但是实际上通过测试，发现有一些TCP服务会对ACK Flood比较敏感，比如说JSP Server，在数量并不多的ACK小包的打击下，JSP Server就很难处理正常的连接请求。对于Apache或者IIS来说，10kpps的ACK Flood不构成危胁，但是更高数量的ACK Flood会造成服务器网卡中断频率过高，负载过重而停止响应

当攻击程序每秒钟发送ACK报文的速率达到一定的程度，DDOS攻击防护才能使主机和防火墙的负载有大的变化。当发包速率很大的时候，主机操作系统将耗费大量的精力接收报文、判断状态，同时要主动回应RST报文，正常的数据包就可能无法得到及时的处理。这时候客户端（以IE为例）的表现就是访问页面反应很慢，丢包率较高。这就是ACK攻击

### UDP-flood

使用原始套接字伪造大量虚假源地址的UDP包，目前以DNS协议为主。

UDP Flood是日渐猖厥的流量型DoS攻击，原理也很简单。常见的情况是利用大量UDP小包冲击DNS服务器或Radius认证服务器、流媒体视频服务器。 100k pps的UDP Flood经常将线路上的骨干设备例如防火墙打瘫，造成整个网段的瘫痪。由于UDP协议是一种无连接的服务，在UDP FLOOD攻击中，攻击者可发送大量伪造源IP地址的小UDP包。但是，由于UDP协议是无连接性的，所以只要开了一个UDP的端口提供相关服务的话，那么就可针对相关的服务进行攻击。

正常应用情况下，UDP包双向流量会基本相等，而且大小和内容都是随机的，变化很大。出现UDP Flood的情况下，针对同一目标IP的UDP包在一侧大量出现，并且内容和大小都比较固定。

### ICMP-flood

Ping洪水，比较古老的方式。

ICMP Flood 的攻击原理和ACK Flood原理类似，属于流量型的攻击方式，也是利用大的流量给服务器带来较大的负载，影响服务器的正常服务。由于目前很多防火墙直接过滤ICMP报文，因此ICMP Flood出现的频度较低。

## 应用层攻击

### CC

ChallengeCollapsar的名字源于挑战国内知名安全厂商绿盟的抗DDOS设备-“黑洞”，通过botnet的傀儡主机或寻找匿名代理服务器，向目标发起大量真实的http请求，最终消耗掉大量的并发资源，拖慢整个网站甚至彻底拒绝服务。

互联网的架构追求扩展性本质上是为了提高并发能力，各种SQL性能优化措施：消除慢查询、分表分库、索引、优化数据结构、限制搜索频率等本质都是为了解决资源消耗，而CC大有反其道而行之的意味，占满服务器并发连接数，尽可能使请求避开缓存而直接读数据库，读数据库要找最消耗资源的查询，最好无法利用索引，每个查询都全表扫描，这样就能用最小的攻击资源起到最大的拒绝服务效果。

互联网产品和服务依靠数据分析来驱动改进和持续运营，所以除了前端的APP、中间件和数据库这类OLTP系统，后面还有OLAP，从日志收集，存储到数据处理和分析的大数据平台，当CC攻击发生时，不仅OLTP的部分受到了影响，实际上CC会产生大量日志，直接会对后面的OLAP产生影响，影响包括两个层面，一个当日的数据统计完全是错误的。第二个层面因CC期间访问日志剧增也会加大后端数据处理的负担。

CC是目前应用层攻击的主要手段之一，在防御上有一些方法，但不能完美解决这个问题。

### DNS flood

伪造源地址的海量DNS请求，用于是淹没目标的DNS服务器。对于攻击特定企业权威DNS的场景，可以将源地址设置为各大ISP DNS服务器的ip地址以突破白名单限制，将查询的内容改为针对目标企业的域名做随机化处理，当查询无法命中缓存时，服务器负载会进一步增大。

DNS不只在UDP-53提供服务，同样在TCP协议提供服务，所以防御的一种思路就是将UDP的查询强制转为TCP，要求溯源，如果是假的源地址，就不再回应。对于企业自有权威DNS服务器而言，正常请求多来自于ISP的域名递归解析，所以将白名单设置为ISP的DNS server列表。对于源地址伪造成ISP DNS的请求，可以通过TTL值进一步判断。

### 慢速连接攻击

针对http协议，以知名的slowloris攻击为起源：先建立http连接，设置一个较大的content-length，每次只发送很少的字节，让服务器一直以为http头部没有传输完成，这样的连接一多很快就会出现连接耗尽。

目前出现了一些变种，http慢速的post请求和慢速的read请求都是基于相同的原理。

## 攻击方式

### 混合型

在实际大流量的攻击中，通常并不是以上述一种数据类型来攻击，往往是混杂了TCP和UDP流量，网络层和应用层攻击同时进行。

### 反射型

2004年时DRDOS第一次披露，通过将SYN包的源地址设置为目标地址，然后向大量的真实TCP服务器发送TCP的SYN包，而这些收到SYN包的TCP server为了完成3次握手把SYN|ACK包“应答”给目标地址，完成了一次“反射”攻击，攻击者隐藏了自身，但有个问题是攻击者制造的流量和目标收到的攻击流量是1:1，且SYN|ACK包到达目标后马上被回以RST包，整个攻击的投资回报率不高。

反射型攻击的本质是利用“质询-应答”式协议，将质询包的源地址通过原始套接字伪造设置为目标地址，则应答的“回包”都被发送至目标，如果回包体积比较大或协议支持递归效果，攻击流量会被放大，成为一种高性价比的流量型攻击。

反射型攻击利用的协议目前包括NTP、Chargen、SSDP、DNS、RPC portmap等等。

### 流量放大型

以上面提到的DRDOS中常见的SSDP协议为例，攻击者将Search type设置为ALL，搜索所有可用的设备和服务，这种递归效果产生的放大倍数是非常大的，攻击者只需要以较小的伪造源地址的查询流量就可以制造出几十甚至上百倍的应答流量发送至目标。

### 脉冲型

很多攻击持续的时间非常短，通常5分钟以内，流量图上表现为突刺状的脉冲。

之所以这样的攻击流行是因为“打-打-停-停”的效果最好，刚触发防御阈值，防御机制开始生效攻击就停了，周而复始。蚊子不叮你，却在耳边飞，刚开灯想打它就跑没影了，当你刚关灯它又来了，你就没法睡觉。

自动化的防御机制大部分都是依靠设置阈值来触发。尽管很多厂商宣称自己的防御措施都是秒级响应，但实际上比较难。

网络层的攻击检测通常分为逐流和逐包，前者根据netflow以一定的抽样比例（例如1000:1）检测网络是否存在ddos攻击，这种方式因为是抽样比例，所以精确度较低，做不到秒级响应。第二种逐包检测，检测精度和响应时间较短，但成本比较高，一般厂商都不会无视TCO全部部署这类方案。即便是逐包检测，其防御清洗策略的启动也依赖于阈值，加上清洗设备一般情况下不会串联部署，触发清洗后需要引流，因此大部分场景可以做秒级检测但做不到秒级防御，近源清洗尚且如此，云清洗的触发和转换过程就更慢了。所以利用防御规则的生效灰度期，在触发防御前完成攻击会有不错的效果，在结果上就表现为脉冲。

### 链路泛洪

随着DDOS攻击技术的发展，又出现了一种新型的攻击方式link-flooding attack，这种方式不直接攻击目标而是以堵塞目标网络的上一级链路为目的。对于使用了ip anycast的企业网络来说，常规的DDOS攻击流量会被“分摊”到不同地址的基础设施，这样能有效缓解大流量攻击，所以攻击者发明了一种新方法，攻击至目标网络traceroute的倒数第二跳，即上联路由，致使链路拥塞。国内ISP目前未开放anycast，所以这种攻击方式的必要性有待观望。

对一级ISP和IXP的攻击都可以使链路拥塞。

# 多层防御结构

DDOS攻击本质上是一种只能缓解而不能完全防御的攻击，它不像漏洞那样打个补丁解决了就是解决了，DDOS就算购买和部署了当前市场上比较有竞争力的防御解决方案也完全谈不上彻底根治。防火墙、IPS、WAF这些安全产品都号称自己有一定的抗DDOS能力，而实际上他们只针对小流量下，应用层的攻击比较有效，对于稍大流量的DDOS攻击则无济于事。

以2015年年中的情况为例，国内的DDOS攻击在一个月内攻击流量达到300G的就将近10-20次，这个数值将随着国内家庭宽带网速提升而进一步放大。对于200~500G的攻击流量该如何防御，下来将展示完整的防御结构，通常可以分为4层。

这4层不是严格意义上的纵深防御关系，也不是在所有的防御中4层都会参与，可能有时候只是其中的2层参与防御。但对于超大流量攻击而言，4层就是防御机制的全部，再没有其他手段了。跟厂商们的市场宣传可能有所不同，大流量攻击的防护并不是像某些厂商宣称的那样靠厂商单方面就能解决的，而是多层共同参与防御的结果。

#### ISP/WAN层

这一层通常对最终用户不可见，如果只是中小企业，那这一层可能真的不会接触到。但如果是大型互联网公司，公有云厂商，甚至是云清洗厂商，这层是必不可少的。因为当流量超过自己能处理的极限时必须要借助电信运营商的能力。大型互联网公司虽然自身储备的带宽比较大，但还没到达轻松抵抗大流量DDOS的程度，毕竟带宽是所有IDC成本中最贵的资源没有之一。目前无论是云计算厂商，大型互联网公司向外宣称的成功抵御200G以上攻击的新闻背后都用到了运营商的抗D能力，另外像第三方的云清洗平台他们实际上或多或少的依赖电信运营商，如果只依靠本身的清洗能力，都是非常有限的。

目前如中国电信的专门做抗DDOS的云堤提供了[近源清洗]和[流量压制]的服务，对于购买其服务的厂商来说可以自定义需要黑洞路由的IP与电信的设备联动，黑洞路由是一种简单粗暴的方法，除了攻击流量，部分真实用户的访问也会被一起黑洞掉，对用户体验是一种打折扣的行为，本质上属于为了保障留给其余用户的链路带宽的弃卒保帅的做法，之所以还会有这种收费服务是因为假如不这么做，全站服务会对所有用户彻底无法访问。对于云清洗厂商而言，实际上也需要借助黑洞路由与电信联动。

相比之下，对攻击流量的牵引，清洗，回注的防御方式对用户体验的挑战没那么大，但是跟黑洞路由比防御方的成本比较高，且触发到响应的延时较大。

在运营商防御这一层的主要的参与者是大型互联网公司，公有云厂商，云清洗厂商，其最大的意义在于应对超过自身带宽储备和自身DDOS防御能力之外的超大攻击流量时作为补充性的缓解措施。

#### CDN/Internet层

CDN并不是一种抗DDOS的产品，但对于web类服务而言，他却正好有一定的抗DDOS能力，以大型电商的抢购为例，这个访问量非常大，从很多指标上看不亚于DDOS的CC，而在平台侧实际上在CDN层面用验证码过滤了绝大多数请求，最后到达数据库的请求只占整体请求量的很小一部分。

对http CC类型的DDOS，不会直接到源站，CDN会先通过自身的带宽硬抗，抗不了的或者穿透CDN的动态请求会到源站，如果源站前端的抗DDOS能力或者源站前的带宽比较有限，就会被彻底DDOS。

云清洗厂商提出的策略是，预先设置好网站的CNAME，将域名指向云清洗厂商的DNS服务器，在一般情况下，云清洗厂商的DNS仍将穿透CDN的回源的请求指向源站，在检测到攻击发生时，域名指向自己的清洗集群，然后再将清洗后的流量回源。

检测方式主要是在客户网站前部署反向代理（nginx），托管所有的并发连接。在对攻击流量进行分流的时候，准备好一个域名到IP的地址池，当IP被攻击时封禁并启用地址池中的下一个IP，如此往复。

以上是类Cloudflare的解决方案，国内云清洗厂商的实现原理都相似。不过这类方案都有一个通病，由于国内环境不支持anycast，通过DNS引流的方式需要比较长的生效时间，这个时间依赖于DNS递归生效的时长，对自身完全不可控。同时CDN仅对web类服务有效，对游戏类TCP直连的服务无效。

网上很多使用此类抗D服务的过程可以概括为一句话：更改CNAME指向，等待DNS递归生效。

#### DC层

目前国内厂商华为的Anti-ddos产品的最高型号支持T级高达1440Gbps带宽的防护。原理大致如下图所示，在DC出口以镜像或分光部署DDOS探针（检测设备），当检测到攻击发生时，将流量牵引到旁路部署的DDOS清洗设备，再将经过清洗的正常流量回注到业务主机，以此完成一轮闭环。

部署设备本身并没有太大的技术含量，有技术含量的部分都已经被当做防御算法封装在产品盒子里了。不过要指出的一点是，目前市场上的ADS盒子既有的算法和学习能力是有限的，他仍然需要人的介入，比如：

对业务流量基线的自适应学习能力是有限的，例如电商行业双11大促日子的流量模型可能就超越了ADS的学习能力，正常流量已经触发攻击判定

自动化触发机制建立在阈值之上，就意味着不是完全的自动化，因为阈值是一个经验和业务场景相关的值

全局策略是通用性策略，不能对每一个子业务起到很好的防御效果，有可能子业务已经被D了，全局策略还没触发

常见的DDOS类型ADS可以自动处理，但变换形式的DDOS类型可能还需要人工识别

默认的模板策略可能不适用于某些业务，需要自定义

DDOS防御除了整体架构设计外，比较需要专业技能的地方就是在上述例子的场景中。目前在ADS设备里覆盖了3-4,7这三层的解决方案。

一般情况下ADS设备可以缓解大部分常见的DDOS攻击类型，相对而言3-4层的攻击手法比较固定，而7层的攻击，由于涉及的协议较多，手法变化层出不穷，所以ADS有时候对7层的防护做不到面面俱到，比如对CC，ADS提供了http 302，验证码等，但还是不能很好的解决这个问题。应用层的防护需要结合业务来做，ADS则在能利用的场景下承担辅助角色，比如对于游戏封包的私有协议，通过给packet添加指纹的方式，ADS在客户端和服务端中间鉴别流入的数据包是否包含指纹。

#### OS/APP层

这是DDOS的最后一道防线。这一层的意义主要在于漏过ADS设备的流量做最后一次过滤和缓解，对ADS防护不了的应用层协议做补充防护。比如NTP反射，可以通过服务器配置禁用monlist，如果不提供基于UDP的服务，可以在边界上直接阻断UDP协议。

互联网在线服务中最大的比重就是web服务，因此有些互联网公司喜欢自己在系统层面去做应用层的DDOS防护，例如对抗CC，有时这些功能也能顺带关联到业务安全上，比如针对黄牛抢单等。

实现方式可以是web服务器模块，也可以是独立部署的旁路系统，反向代理将完整的http请求转发给检测服务器，检测服务器根据几方面的信息：

来自相同IP的并发请求

相同ip+cookie的并发请求

相同http头部可设置字段

相同的request URL

然后保存客户端的连接信息计数表，例如单位时间内相同IP+cookie再次发起连接请求则此客户端IP的计数器+1，直到触发阈值，然后服务器会进行阻断，为了避免误杀，也可以选择性的弹出验证码。

以上是拿CC防御举了个例子，ADS设备本身提供了http 302跳转，验证码，Javascript解析等防御方式，尽管OS和应用层可以做更多的事情，但毕竟有自己去开发和长期维护的代价，这个收益要看具体情况。

### 链路带宽

增加带宽，大部分介绍DDOS防御策略的文章里似乎很少提及这一点，但却是以上所有防御的基础，例如第二层次的CDN实际上就是拼带宽，很多大型企业选择自建CDN，本质上就是自己增加带宽的行为。除了CDN之外，要保障DC出口的多ISP链路、备份链路，到下层机柜交换机的带宽都不存在明显的单点瓶颈，否则抗DDOS的处理性能够了，但是流量流经某个节点时突然把一个杂牌交换机压垮了，最后的结果还是表现为有问题。

对运维经验成熟的互联网公司而言，一般都有能容管理，对于促销活动，高峰时段的带宽，IDC资源的波峰波谷都有预先的准备，DDOS防御主要是消除这些网络方案中内在的单点瓶颈

## 不同类型的企业

DDOS的防御本质上属于资源的对抗，完整的4层防御效果虽好，但有一个明显问题就是TCO[总所有成本TCO (Total cost of ownership)]，这种成本开销除了互联网行业排名TOP10以外的公司基本都吃不消。或者就算付得起这钱，在IT整体投资的占比会显得过高，付得起不代表这种投资是正确的。所以针对不同的企业分别描述DDOS缓解方案的倾向和选择性。

大型平台

这里的“大”有几层意思：

公司很有钱，可以在补贴具体的业务上不“太”计较投入，对土豪来说只选效果最优方案

公司不一定处在很赚钱的阶段，但IDC(（互联网数据中心）)投资规模足够大，这样为了保障既有的投入，安全的总投资维持一个固定百分比也是非常有必要的，在IDC盘子很大的时候没必要省“小钱”,与潜在的由于服务中断造成的损失比，DDOS防御的投资可以忽略不计

映射到现实中与这几条相关的公司：

市值100亿美元以上互联网公司

大型公有云厂商，IaaS、PaaS平台

IDC规模多少算大呢，这个问题其实很难判断，1w台物理服务器算多么，现在应该只能算中等吧

利润比较高的业务，比如游戏、在线支付假如被DDOS的频率较高

对于IDC规模比较大又有钱的公司来说，防DDOS的口诀就是“背靠运营商，大力建机房”，在拥有全部的DDOS防御机制的前提下，不断的提高IDC基础设施的壁垒给攻击者制造更高的门槛，对于网络流量比较高的公司而言，抗DDOS是有先天优势的，因为业务急速增长而带来的基础设施的扩容无意识间就成了一种防御能力，但对于没有那么大业务流量的公司，空买一堆带宽烧钱恐怕也没人愿意。

对于比较有钱，但没那么多线上服务器的公司而言，自己投入太多IDC建设可能是没必要的，此时应该转向通过购买的手段尽可能的获得全部的DDOS防御机制。

中小企业

资源的对抗肯定不是中小企业的强项，所以追求ROI(投资回报率)是主要的抗DDOS策略。第一种极度省钱模式，平时裸奔，直到受攻击才找抗DDOS厂商临时引流，这种方案效果差一点，绝大多数企业应该都是这种心理，得过且过，能省则省，也无可厚非，不过关键时间知道上哪儿去找谁，知道做哪些事。

第二种追求效果，希望有性价比。如果本身业务运行于公有云平台，可以直接使用云平台提供的抗DDOS能力，对于web类可以考虑提前购买云清洗厂商的服务。

## 不同类型的业务

不同的类型的服务所需要的DDOS防御机制不完全相同，所以不能直接拿前述4层生搬硬套。

Web

对于web类服务，攻击发生时终端用户可以有一定的延迟容忍，在防御机制上4层全部适用，大型平台的一般都是4层全部拥有，规模小一点的企业一般购买云清洗，cloudflare类的服务即可。

游戏类

Web api形式的轻游戏跟web服务类似，而对于TCP socket类型的大型网游，稍有延迟很影响用户体验，甚至很容易掉线。云WAF、CDN等措施因为是针对web的所在在该场景下无效，只有通过DNS引流和ADS来清洗，ADS不能完美防御的部分可以通过修改客户端、服务端的通信协议做一些辅助性的小动作，例如封包加tag标签，检测到没有tag的包一律丢弃，防御机制基本都是依赖于信息不对称的小技巧。DNS引流的部分对于有httpdns的厂商可以借助其缓解DNS递归生效的时间。

服务策略

分级策略

对于平台而言，有些服务被DDOS会导致全站服务不可用，例如DNS不可用则相当于全线服务不可用，对于强账号体系应用例如电商、游戏等如果SSO登陆不可用则全线服务不可用，攻击者只要击垮这些服务就能“擒贼擒王”，所以安全上也需要考虑针对不同的资产使用不同等级的保护策略，根据BCM的要求，先将资产分类分级，划分出不同的可用性SLA要求，然后根据不同的SLA实施不同级别的防护，在具体防护策略上，能造成平台级SPOF（单点故障）的服务或功能应投入更高成本的防御措施，所谓更高成本不仅指购买更多的ADS设备，同时可能建立多灾备节点，并且在监控和响应优先级上应该更高。

Failover机制

DDOS防御不只是依赖于DDOS防御的那4层手段，同时依赖于基础设施的强大，例如做分流，就需要多点异地灾备，mirror site & hot site & standby system，云上的系统需要跨AZ部署，这些可以随时切换的基础。把鸡蛋放在一个篮子里会导致没什么选择。

基础设施和应用层面建立冗余是技术形式上的基础，光有这些还远远不够，需要有与之配套的DRP&BCP策略集，并且需要真实的周期性的演练，意在遇到超大流量攻击时能够从容应对。

有损服务

在应用服务设计的时候，应该尽量避免“单点瓶颈”，避免一个点被DDOS了整个产品就不好用了，而是希望做到某些服务即使关闭或下线了仍然不影响其他在线的服务（或功能），能在遇到需要弃卒保帅的场景时有可以“割肉”的选择，不要除了“0”就是“1”，还是要存在灰度，比如原来10个服务在线，遇到攻击时我只要保底重要的3个服务在线即可。

补充手段

DDOS攻击的目的不一定完全是出于想打垮服务，比如以前在做游戏时遇到玩家DDOS服务器的目的竟然是因为没抢到排在第一的房间，这种因素通过产品设计就可以根治，而有很多应用层DDOS只是为了达成另外一种目的，都跟上述4层防御机制无关，而跟产品设计有关。所以防御DDOS这事得看一下动因，不能盲目应对。