TOR&互联网 B 面

（一）简介：

Tor 的全称是“The Onion Router”，号称是“An anonymous Internet communicaton system”。它针对现阶段大量存在的流量过滤、嗅探分析等工具，在 JAP 之类软件基础上改进的，支持 Socks5，并且支持动态代理链（通过 Tor 访问一个地址时，所经过的节点在 Tor 节点群中随机挑选，动态变化，由于兼顾速度与安全性，节点数目通常为 2-5 个），因此难于追踪，有效地保证了安全性。另一方面，Tor 的分布式服务器可以自动获取，因此省却了搜寻代理服务器的精力。

Tor 最初设计、实施和部署为海军研究实验室的第三代洋葱路由项目。 它最初是与美国海军开发的，主要目的是保护政府通信。 今天，它被军事，记者，执法官员，活动家和许多其他人每天用于各种各样的目的。

（二）最新版本：Tor6.0:

Tor 团队还特别针对 Mac OS X 系统进行了漏洞补丁，更新后，Tor 网络在 Mac OS X 系统内运行时，将使用代码签名来避免被 Mac OS X 系统自带的安全软件封锁。

现在，不论是 Linux、Windos 还是 Max 系统的用户，现在都可以通过 Softpedia 来下载 Tor 网络 6.0 版本。对于已经下载过 Tor 网络的用户，通过内置的升级更新程序实升级就可以了，Tor 将会向用户提供 6.0 版本的完整更新日志

Tor 团队还说明了一下，由于他们的老合作伙伴，Disconnect 搜索引擎与 Google 之间的合作情况有变，Tor 现在是通过 DuckDuckGo 的 API 显示搜索结果，而不再是 Google。

Tor 新版本最大的变化，就是对浏览器加密层所做的修改，因为 Tor 是运行在一个最好的首层加密协议上的，所以对现代密码学的支持必须做到位才能不负盛誉.

SHA- 1 是由美国国家安全局（NSA）设计，美国国家标准与技术研究院（NIST）发布的一系列密码散列函数，SHA-1 散列函数加密算法输出的散列值为 40 位十六进制数字串，用于验证信息的一致性，保护敏感的为保密资料，防止被篡改。

随着黑客技术的提高，SHA-1 已变得很容易被攻破了。安全专家们在 1 年前就建议使用 SHA-1 散列的网站，尽快更新到 SHA-2 或 SHA-3 版本。

为此，Tor 网络 6.0 版本移除了对 SHA-1 的支持。

而使用 SHA-1 的一些“爸爸们”，如火狐、谷歌以及 Edge 也都已经在去年冬季时宣布要淘汰对它的 支持。这些浏览器动作比较迟缓的原因大概是因为他们的用户量太大，只能先在今年 7 月份时移除 SHA-1，然后在 2017 年初时才能实现彻底淘汰。

（三）、Tor 保护举例：

1、梗概：

Tor 网络是一组志愿者操作的服务器，允许人们改善他们在互联网上的隐私和安全。Tor 的用户通过一系列虚拟隧道连接而不是直接连接，从而允许组织和个人通过公共网络共享信息，而不会影响他们的隐私。 沿着同一条路线，Tor 是一个有效的审查制裁工具，允许其用户访问其他被阻止的目的地或内容（翻墙）。 Tor 还可以用作软件开发人员创建具有内置隐私功能的新通信工具的构建块。

Tor 网络是基于通道交换的低延迟匿名通信服务,用户需要构建相应的匿名传输通道,然后才能进行应用数据的匿名通信。因此通道构建是整个 Tor 网络的核心内容之一

相比于当前的通道构建协议,基于随机游走的 Tor 网络通道构建协议为用户提供了更好的传输匿名度,处于同一水平的传输性能和更好的用户体验

使用 Tor 的人的种类实际上是使它如此安全的一部分。 Tor 隐藏你在网络上的其他用户，所以更多的人口和多样化的用户群体 Tor，是你的匿名将被保护得越多。

2、常见攻击举例

（1）攻击——**流量分析：**

流量分析可用于推断谁通过公共网络与谁通话。了解用户互联网流量的来源和目的地，允许其他人跟踪您的行为和兴趣。 这可能会影响您的支票簿，例如，如果电子商务网站使用基于您所在国家/地区或原产地的价格歧视。 它甚至可能威胁你的工作和身体安全，揭示你是谁和你在哪里。 例如，如果您在国外旅行，并且连接到您的雇主的计算机检查或发送邮件，您可能无意中泄露您的国籍和专业联系网络任何人，即使连接是加密的。

流量分析的攻击方式：

因特网数据分组具有两个部分：**数据有效载荷和用于路由的报头。**

数据有效载荷是被发送的，无论是电子邮件，网页还是音频文件。 即使你加密你的通信的数据有效载荷，流量分析仍然揭示了很多关于你在做什么，可能，你在说什么。 这是因为它专注于头部，其中公开源，目的地，大小，时间等 。

隐私的一个基本问题是，您的通信的收件人可以看到您通过查看标头发送它，所以可以授权中介如互联网服务提供商查看标头，有时即使是未经授权的中介。 一种非常简单的流量分析形式是监控者位于网络上的发送者和接收者之间的某处，只是监控报文的头部。

但也有更强大的交通分析类型。 一些攻击者监视互联网的多个部分，并使用复杂的统计技术跟踪许多不同组织和个人的通信模式。 加密对这些攻击者没有帮助，因为它只隐藏 Internet 流量的内容，而不是头。

（2）Tor 的解决方案：一个分布式，匿名网络

Tor 通过在因特网上的多个地方分发您的交易，帮助减少简单和复杂的流量分析的风险，所以没有一个点可以链接到您的目的地。 这个想法类似于使用一个扭曲、难以跟随的路线，以抛弃一个拖尾你的节点，然后定期擦除你的脚印。而不是采取从源到目的地的直接路由，Tor 网络上的数据包采取随机途径通过几个继电器覆盖您的轨道，所以没有观察员在任何一个点可以告诉数据来自哪里或它去哪里。

为了创建与 Tor 的专用网络路径，用户的软件或客户端通过网络上的继电器递增地建立加密连接的电路。电路每次扩展一跳，并且每个继电器只知道哪个继电器给它数据和哪个继电器给数据。没有单个中继器知道数据分组已经采取的完整路径。客户端为沿着电路的每一跳协商单独的一组加密密钥，以确保每个跳不能在它们通过时跟踪这些连接。

一旦建立了电路，可以交换许多种类的数据，并且可以通过 Tor 网络部署多种不同类型的软件应用。因为每个中继在电路中看不到一跳，所以窃听者或被泄漏的中继都不能使用业务分析来链接连接的源和目的地。Tor 仅适用于 TCP 流，并且可以由具有 SOCKS 支持的任何应用程序使用。

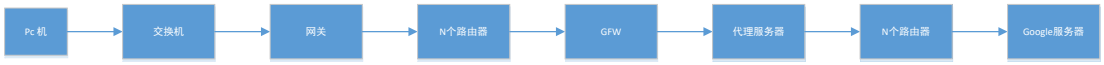
为了效率，Tor 软件只在十分钟内使用相同的电路，十分钟左右内发生的重新连接。后来的请求给了一个新的电路，以防止人们链接你的早期行动与新的电路。

（四）Tor 保护原理：

1、匿名：

先构建一个模型（大部分翻墙党都符合这个模型）：

从本地 PC 开始，经过交换机（平时很多人说的"家庭路由器"真正的名字叫交换机，真正的路由器不是个人能买得起的），网关（如果是局域网），N 个路由器，GFW ，代理服务器，然后又是 N 个路由器，最终到达 google 服务器。如果是改 hosts 来的，那么就没有代理服务器这一步了。



这其中只要数据是被强加密的，那些交换机路由器网关以及 GFW 就都不是问题。当然，在代理服务器之前的交换机网关路由器以及 GFW 是知道你的真实 IP 的（交换机还知道你的真实 MAC 地址呢），但在强加密的情况下，他们不知道你干了些什么（"不知道你具体干了什么"不等于"不知道你在翻墙"，ISP 和 GFW 想要知道你在翻墙不是一件难事）；

代理服务器负责处理数据包，对于用户而言，他是服务器；对于 google 服务器而言，他是客户端。又当服务器又当客户端，这就是代理服务器的特别之处。代理服务器之后的路由器以及最终的 google 服务器看到的都是代理服务器的 IP 了，用户的真实 IP 就被隐藏起来了，所以说"有一定的匿名性。"但问题在于代理服务器知道你的真实 IP 和你干了些什么（HTTP），所以一旦代理服务器出事，后果不堪设想。

那么，对于匿名这一步而言，关键点如下：

（1）各个环节除了必要的信息之外，其他什么都不该知道，这样当某一环节被攻破时，用户匿名失效的可能性最小；

（2）用户通信时跨越适当数量的跳板（hop ，或者叫中继节点），而且最好是不同国家的 hop ，这样流量分析就难以进行，逆向追踪的难度也大大增加了，这些 hop 也只是知道他们正常工作必须知道的信息而已，除此之外什么也不知道.

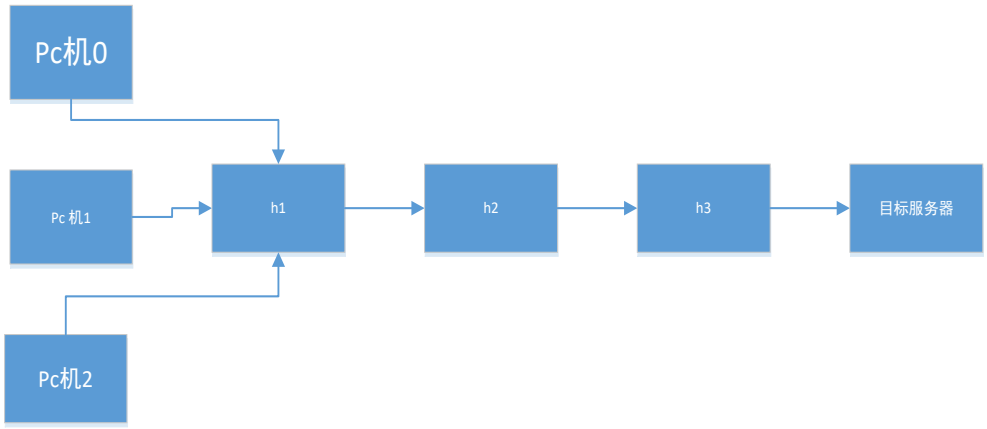
例如：有三个 hop ：h1，h2，h3. 用户首先连到 h1，然后 h1 连接到 h2， h2 连接到 h3，h3 最终连接到目标网站服务器上。在这一过程中，

h1 知道用户的真实 IP，

h2 知道它从 h1 接收用户数据以及要把数据送到 h3，

h3 知道用户想干什么（HTTP）或者用户要去哪里（HTTPS ），

除此之外这些 hop 不应该知道其他任何有关用户的信息，这样的话即使三个 hop 中有一个是攻击者的蜜罐，用户身份也不会暴露（不过如果三个 hop 都是蜜罐，用户的身份还是会暴露的，但当三个 hop 分别位于三个不同的国家时，这种可能性很小）；



其次就是才用“藏叶于林”的方案，即如果有很多人同时使用那三个 hop ，那么监控者想要找出其中某个具体用户就很困难了。同样道理，每次连接建立时都应该随机选择 hop ，这样监控者都不知道该如何进行有效的监视（TOR 的策略是每 10 分钟就重新随机选择三个 hop，每次启动 TOR 时也是随机选择 hop 的）。

（3）最后就是中间人攻击的问题了。

为了防止监控者进行中间人攻击破坏用户匿名，Tor 有着自己的一套基于数字证书的身份认证机制，每个 hop 都有一个自签发的数字证书，Tor 客户端在连接建立时会进行严格的身份认证。

补充说明：

这套证书系统是 Tor 独有的，跟操作系统和浏览器的证书系统完全没有关联，所以 GFW 即使利用流氓证书 CNNIC ROOT 进行中间人攻击也无法影响到 Tor,但 Tor 出口节点到目标网站服务器这段路还是需要防备中间人攻击的。

综合上述分析，在匿名系统方面，和之前的相比，tor 有如下三点最为特别：

（1） 一个 Tor 环路可以被很多人同时使用，说的具体一点，hop 与 hop 之间的每个 TLS 连接都包括了很多不同用户的 TCP 数据流；

（2） Tor 客户端随机选择了入口节点，入口节点随机选择了中间节点，中间节点又随机选择了出口节点。对节点的选择，是由精心设计过的算法保证随机选择的，没有规律可循（不过后来发现这算法造成了很搞笑的情景：某一条环路上的三个 hop 有很多用户，都塞车了，但另一条环路上的三个 hop 用户数少得可怜，所以后来又基于 hop 的用户数对算法进行了改进）；

（3） Tor 拥有独立于操作系统和浏览器的严格的基于数字证书的身份认证机制。这一点很重要，如果 Tor 的身份认证机制和特定的操作系统或浏览器有了关联，那就意味着用户将不得不为了使用 Tor 而更换浏览器或操作系统。

2、Tor 的具体工作机制：

官网介绍：

TOR 是一个三重代理，TOR 客户端先与目录服务器通信获得全球活动中继节点信息，然后再随机选择三个节点组成 circuit （电路），用户流量跳跃这三个节点（hop）之后最终到达目标网站服务器，每隔 10 分钟左右就会再重新选择三个节点以规避流量分析和蜜罐节点。

如果真如官网所言，Tor 和其它三重代理相比，除了每隔十分钟更换节点之外没有任何区别。但是真实情况远非如此。

查阅分析，具体的工作机制如下：

1、当用户启动 Tor 后，Tor 客户端就会在本机 PC 上运行一个 onion proxy ，同

时开始监听本机的 9150 端口（Tor Browser 里的 Tor 监听端口），所有经过这一端口的流量都会在经过 onion proxy 的处理之后进入 Tor 电路中；

注：对于服务器，只要服务器开机，端口就一直存在；

但对于本机，只有相应的应用程序试图与远程服务器通过特定端口通信时，端口才会出现，一旦通信结束端口就会立即消失。

也就是说，这个 9150 端口只有在某一应用程序（例如被设置过的浏览器,比如 Tor ）试图通过它与远程主机通信时才会出现，并不是个物理存在。

2、然后是与存有全球中继节点信息的目录服务器取得联系。一开始这一步是明文 HTTP 通信，有着增加流量指纹的风险。

后来 Tor 开发者进行了改进，让 Tor 客户端第一步先与入口节点通信（当然第一次连接是做不到的，以后的连接都可以这么做了）再与目录服务器通信更新节点信息，全过程都是 TLS 的，这样做不仅保证了安全，还避免了单独与目录服务器建立一次性的 TLS 连接，提高了效率（因为建立 TLS 连接是很消耗资源的，生成随机参数进行密钥交换强加密传输数据都是要进行大量运算的，如果只是为了短暂的一次性连接，那么就有些太浪费了）；当完成更新中继节点信息之后，客户端不必切断连接，而是可以直接把原来的 TLS 连接拔出来再继续组成电路（意思是原来的 TLS 连接的参数还可以继续使用，而不用又重新计算生成）得到中继节点信息之后，电路构造过程就正式开始；

3、Tor 环路的核心思想：哪个节点都不可信，像洋葱一样。

基于这种理论，和普通的 TCP 流不同（Tor 工作在 TCP 流之上），Tor 协议把通信数据打包为了一个个特殊的 cell：一开始建立 Tor 电路时，本机上的 onion proxy 向入口节点发送 create cell 进行 TLS handshake，这一过程的身份认证过程是基于数字证书的：Tor 有着自己的一套数字证书系统，每一个洋葱路由（就是节点）都有一个用于签发证书的身份密钥和用于解密用户的电路建立请求以及协商出一个用于后续通信的（使用时间）短暂的密钥。当节点之间进行通信时，这个特制的 TLS 协议还会建立短期连接密钥，而且这一密钥会周期性的独立发生变化以最大限度降低密钥泄露带来的风险；

Tor 电路默认由客户端和三个节点组成，在这种情况下 Tor 电路建立方式如下：

（1） 客户端最先发过去的 relay cell 1 是入口节点与中间节点建立 TLS 连接时所需要的参数，

（2） 接着发过去的 relaycell 2 是中间节点与出口节点建立 TLS 连接时所需要的参数，

（3） 接下来是只有目标网站和对应端口信息的 relay cell 3，在返回表明已成功建立环路的 relay cell 4 之后才会真正开始发送包含用户信息的 relay cell 5.

这一过程中 relay cell 1 被一重加密，到了入口节点之后就被解密，再用来与中间节点完成握手；

relay cell 2 被两重加密，到了入口节点时第一重加密解除，到了中间节点时第二重加密解除，中间节点可以看到明文，用来与出口节点完成握手；

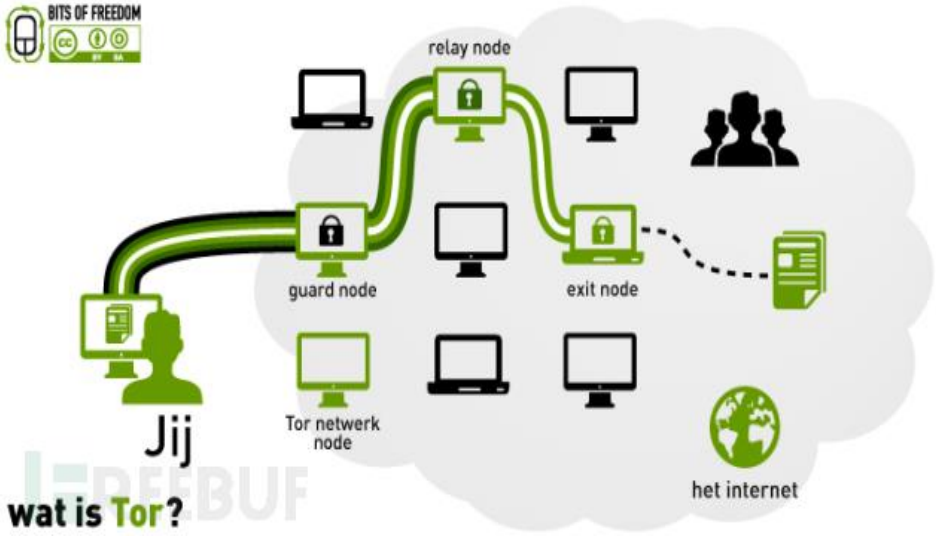
relay cell 3 被三重加密，只有出口节点能看到明文，被用来与目标网站建立连接。同样 relay cell 5 也是三重加密的，只有出口节点能看到明文（HTTP）。用户数据就像洋葱一样，被层层包裹着，只有到了终点包裹才会解开。

在这一过程中，只有入口节点知道用户的真实 IP 地址，出口节点知道用户的目

的地和传输内容（HTTP），Tor 电路的 cell 里没有其他任何关于用户真实身份的信息（不考虑 cookie 和 flash 插件等应用层协议和程序带来的隐私）。

注：

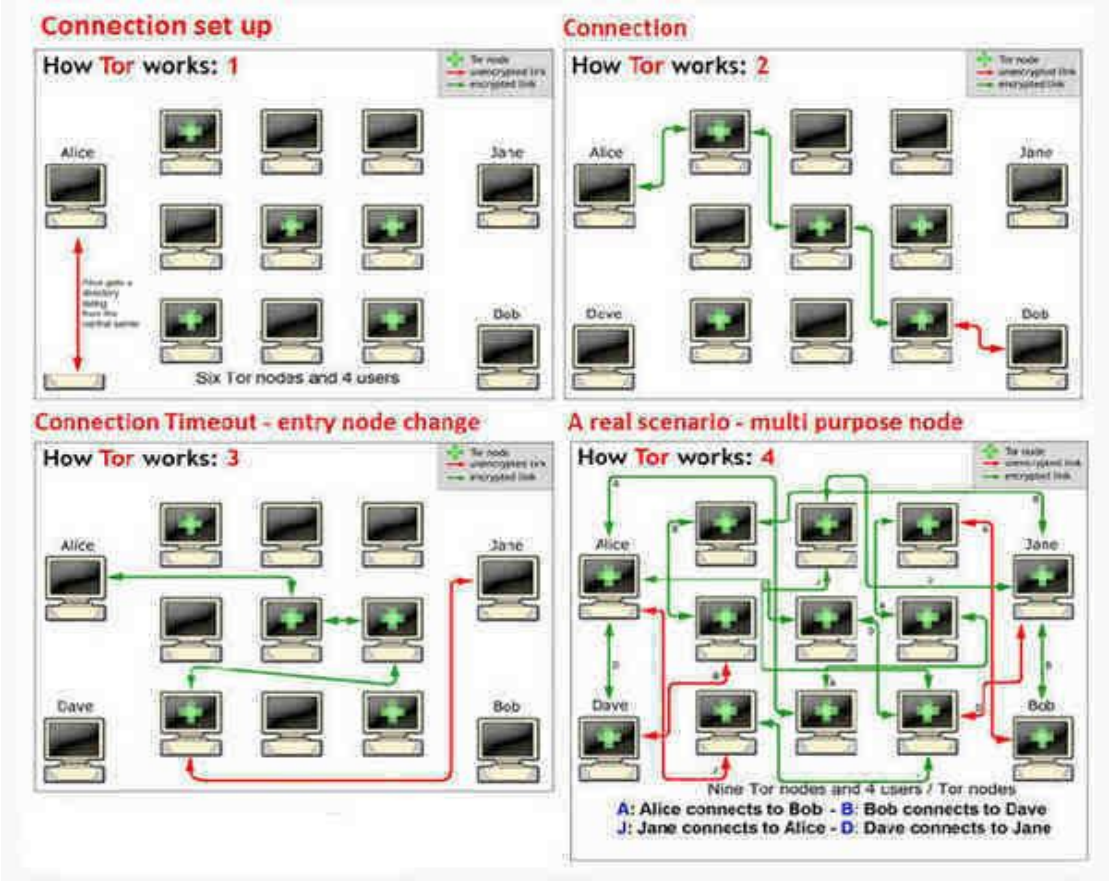
Tor 电路中的节点们并没有预设参数，而是采取了“半握手”的方法：在 diffie-hellman 算法的帮助之下，onion proxy 和中间节点通过入口节点的中介交换了参数，然后就能各自算出私钥用于后续通信而不用担心有人监听，入口节点虽然知道这两个参数，但也根本没办法算出私钥来，这是 diffie-hellman 算法设计时所保证的。同理也可以与出口节点安全通信，客户端最里面一层的加密只有出口节点能够解开，入口节点和中间节点都看不到只有出口节点才能看到的内容。



电路建立

4、Tor 在建立电路时选择节点的方式：

Tor 客户端先随机选择一个入口节点，然后入口节点再随机选择一个中间节点，中间节点又随机选择一个出口节点，这样来最大限度实现随机建立电路。而且 Tor 还有一个更特别的地方：很多用户的流量可以被整合到一个 TLS 连接里同时传输！看起来就像一个用户一样！这样一来，匿名程度更高了。

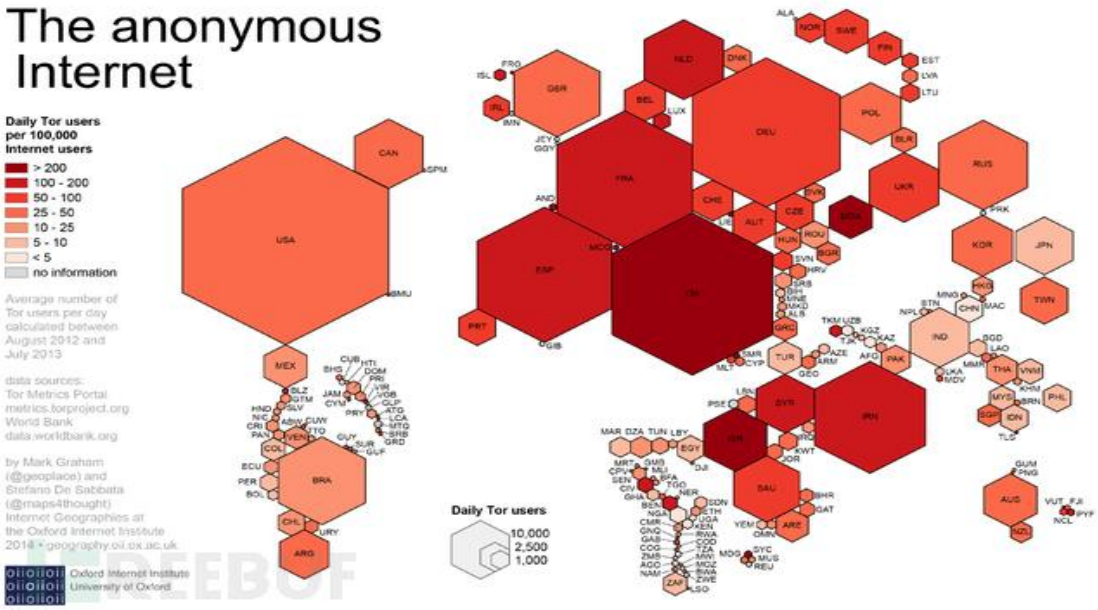


节点选择方式

注：

带来的小问题：很多用户的来自同一个 IP 的大量流量都同时指向同一个网站（例如 google），看上去非常像是 DoS （拒绝服务）攻击，此时就会触发网站的防御机制，网站会要求用户输入验证码。

5、暗网（deep web）：



暗网

（1）对于一般的网站而言，访问过程如下：

首先，用户在浏览器地址栏中输入 URL ，然后回车；紧接着域名解析就开始了，浏览器通过查询从 DNS 服务器得知目标网站的真实 IP 地址，然后开始发起连接……

可是对于暗网，ip 地址被隐藏起来了，根本无法得知 ……

（2）暗网之内建网站与访问：

①要想 deep we 里建立自己的网站，首先要随机选择几个″介绍点″（introduction point ）与之建立电路。（也就是说在 hidden service 和介绍点之间有三个中继节点作为跳板），介绍点不知道 hidden service 的真实 IP；

然后 hidden service 组合起一个描述符，里面包括了公钥和各个介绍点的摘要，然后用私钥签名，最后把描述符上传到数据库（分布式散列表）中。这样网站就建立好了！

这个网站的域名：一个从公钥派生出的 16 位字符。网址和描述符是一一对应的。

②客户端：

i 开启 Tor ，还要设置好浏览器代理，接着输入网址，回车；

ii 通过 Tor 电路与数据库建立连接（这里要注意一下，因为始终都不会访问明网，所以第三个跳板节点在这里就是数据库了，接下来建立连接时第三个跳板节点会变成相应的节点），开始查询（对比是否存在和目标网站网址对应的描述符），同时随机选择一个节点作为″会合点″（rendezvous point）。

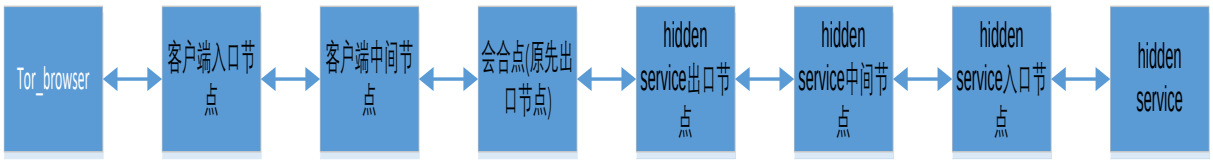
iii 查到了，那么接下来就产生一个随机的一次性 rendezvous cookie 作为一次性的 secret。然后用刚刚得知的公钥加密 cookie 和会合点 IP，再把密文发送给介绍点。

iv 介绍点接到数据之后，就传回给 hidden service（当然始终是通过 Tor 电路间接传递的）。hidden service 用私钥解密，匹配正确。然后就与会合点建立连接（当然也是经过三次跳板中转的完整 Tor 电路），同时把之前的 cookie（也就是一次性 secret）发送回去。

v 客户端接到数据，解密得到之前的那个 cookie，意味着我和 hidden service 之间已经成功建立连接。

最终在客户端和 hidden service 之间有六个中继节点：客户端入口节点，客户端中间节点，会合点（原先出口节点的位置），hidden service 出口节点，hidden service 中间节点，hidden service 入口节点。

从头到尾客户端和 hidden service 的通信都是被 TLS 强加密的（尽管浏览器不这么认为，在访问 deep web 网站的时候浏览器显示客户端和网站之间的连接是未加密的，但实际上刚好相反），Tor 电路本身的证书认证机制也防止了中间人攻击。所以，hidden service 本身再去搞一个证书支持 TLS 连接（让浏览器认为建立了加密连接）其实没有意义了。



（五）漏洞：

既然 guard 被随机选择，那么如果攻击者连接到 Tor 网络上面足够多的电脑，就有很高的几率，在某些情况下，能够成功的窥探到他们其中的一个或者几个。

一个电路的建立过程中，Tor 网络上的计算机们将大量的来回的传递数据。研究人员发现，通过一个 guard 可以很简单的找到在各个方向上传递的数据包流量。利用机器学习算法，就能以 99%的准确率分辨出这是一个普通的网页环路、introduction-point circuit（引入点电路）还是一个 rendezvous-point（会合点）。因此并不需要打破 Tor 的加密。

此外通过使用 Tor 的电脑连接到一系列不同的隐藏服务，他们表明，类似于流量分析模式可以以 88%的准确率确定这些服务。这意味着一个幸运的攻击者进入到了隐藏服务的 guard 的位置时，它将有 88%的把握，确定它就是该隐藏服务的主机。

同样，一个幸运的间谍进入到 guard 位置，那么他有百分之 88 的准确率，发现哪些网站被用户访问过。因此环路（circuit）的指纹是 Tor 隐藏服务的一大隐患。

在用 tor 传递消息的时候，如果它要到一个直接连接到 Tor 网络的服务器上“Tor 隐藏服务”，没有任何问题。但是，如果你只是使用 Tor 作为代理来访问你经常上的网络，就有点复杂。因为在某些时候，你的流量需要经过一个 Tor “出口节点”， 该节点负责把你的数据包传送到网络上。流量很容易在这些出口节点被窥探。

当我们用 tor 浏览网页的时候，发送的任何邮件都会很容易在出口节点窥探。

实例：

瑞典安全研究人员“Chloe”制定了巧妙的技术来欺骗被监听的节点。她建立了一个蜜罐网站，并使用了一个貌似合法的域名并进行网页设计。作为特定测试，她以比特币为主题。然后使用不同的账户密码组合通过不同的 Tor 出口节点登陆这个蜜罐网站。然后，她测试了一个月。发现任何被监视的节点都窃取了她的用户名和密码，并尝试使用它。她记录下蜜罐网站上出现的很多登录尝试。由于每个节点的密码是唯一的，因此 Chloe 可以找到到底哪些节点上钩了。

该实验的结果很有趣。约 1400 个退出记录， 16 个尝试窃取密码和登录。这个数字在表面看起来并不多，但是却让人不得不注意。

（六）如何安全的使用 Tor:

1.使用暗网。

与出口节点保持安全距离的最简单的方法就是不使用它们：坚持使用 Tor 本身的隐匿服务，你可以确保所有的通信都是加密的，无需跨越更多的互联网。但是这种方式有时很有效。暗网只是互联网中众多网站的一小部分。

2.使用 https。

另一种方式使 Tor 的更安全的方法是增强终端到终端的加密协议。其中最有用的一般是 HTTPS，允许你在加密模式下浏览网站。Tor 网站默认支持 HTTPS 的功能。在你发送任何敏感信息之前检查一下 HTTPS 按钮是否为绿色。

3、使用匿名服务。

也可以使用不会记录活动的网站和服务提高你的安全。例如，虽然谷歌通过你平时的搜索活动找出你是谁。但是他们不是用于任何恶意目的，仅仅只是作为其业务的一部分。因此，您需要使用像 Duck Duck Go 浏览器之类的，它有一个服务功能就是不会保留任何关于你信息。你也可以结合 Cryptocat 加密聊天功能进行私人会话。

4、避免个人信息。

避免个人信息泄漏的最安全的方式就是在起先时候就避免发送信息。使用 Tor 浏览固然不错，但也要最大程度地避免信息上传。只能尽可能避免聊天、发送邮件和上论坛。

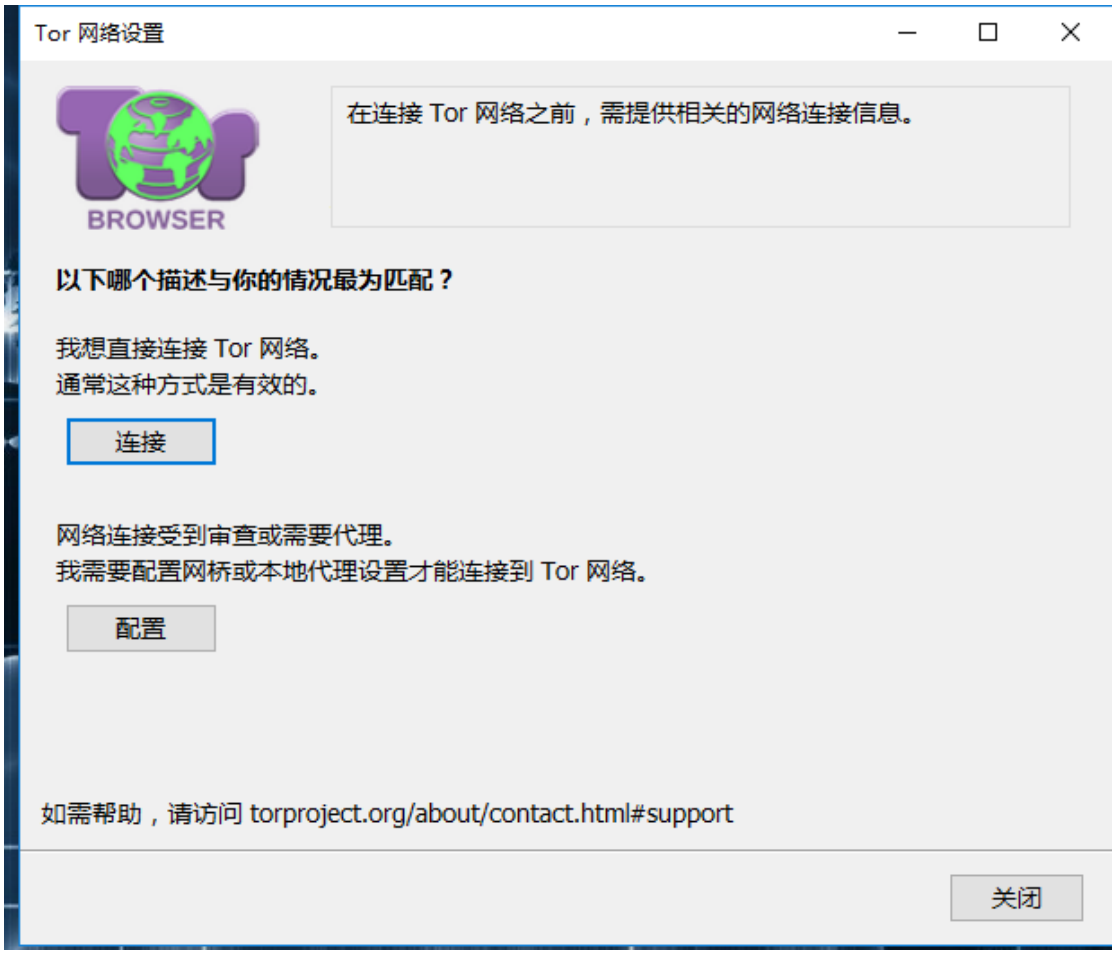
5、避免登陆。

最后，避免浏览需要你登录的网站。通过 Tor 浏览 reddit 就存在这潜在的危险，因为它包括了浏览、发布和评论，让攻击者很容易获取到个人信息。你也应该注意避免像 Facebook 知道你的身份后把它买给广告商理事之类的事。 Tor 不是魔术，并不能保护你，除非你不在乎你的信息被别人知道。

（七）使用 Tor：

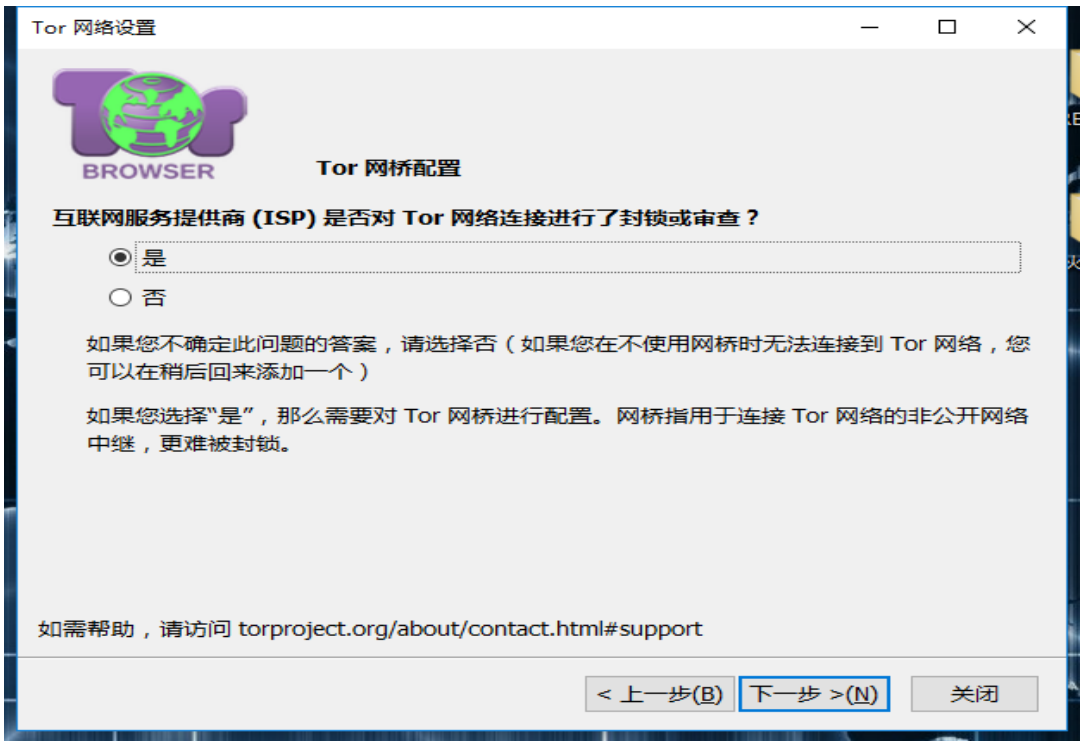
1、下载安装：

在 tor 的官网或者一些开源网站，下载 tor 的安装包，安装。

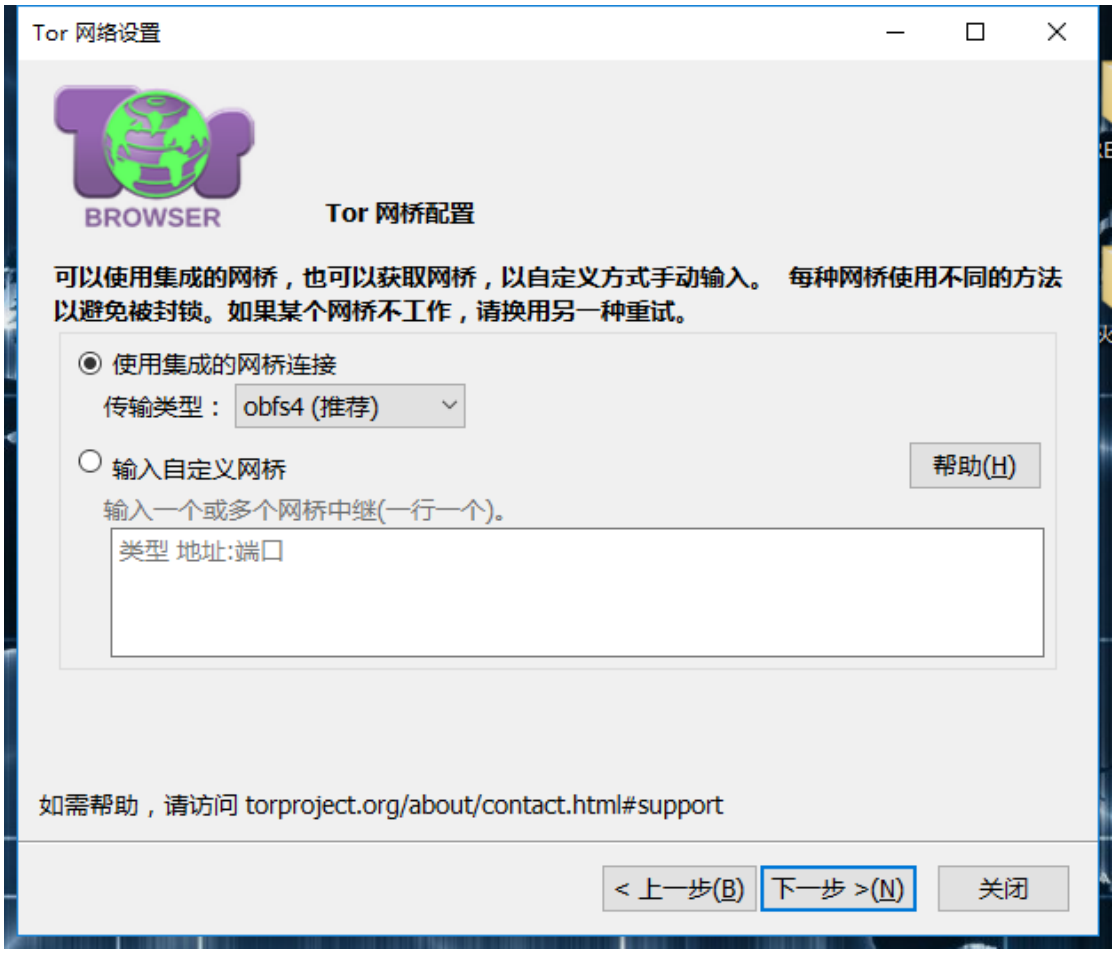


不要通过直接点击来连接，因为 GFW 的存在，会封锁和审查我们的线路。这种办法通常不能直接连接暗网成功。

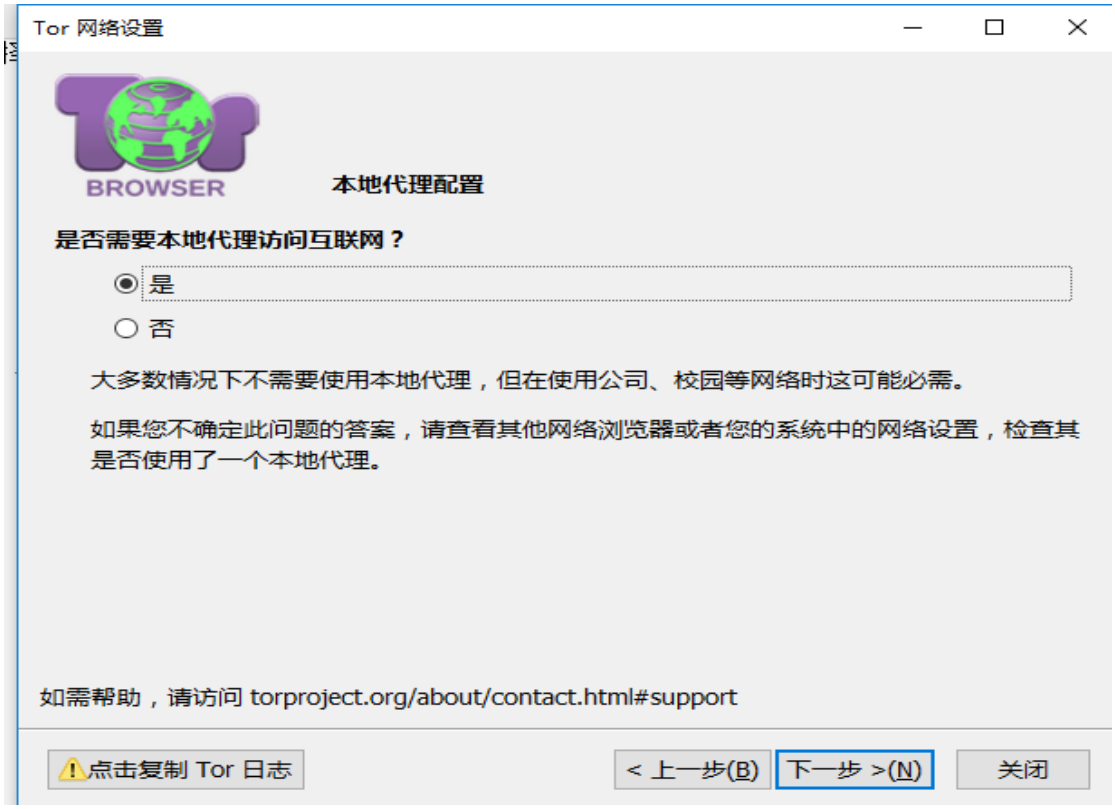
2、配置网桥和端口：
按照图片的提示，一直操作到输入网桥这一步。



注：网桥即 Tor 中继节点，用于帮助用户绕过审查或封锁。
在这一步可以选择“使用集成的网桥连接”，也可以选择“输入自定义的网桥”连接。
（1）使用集成的网桥：
Tor 内部集成了几种网桥，可以选择不同的网桥试验；
（2）手动输入网桥：
获得网桥的方法：
①网页方式
使用浏览器访问 https://bridges.torproject.org
②电子邮件
自动回复方式发送电子邮件至 bridges@torproject.org，并且正文中需包含“get bridges” 这两个单词（如需获取 obfs3 网桥，请写“get transport obfs3”）。为了防止封锁者大量获取网桥地址，发送网桥请求邮件必须使用以下网站的电子邮箱（按推荐度由高到低排列）。https://www.riseup.net，https://mail.google.com 或者 https://mail.yahoo.com。
当获得自己的网桥之后，可以在下图所示的界面中输入网桥。
网桥.txt 中有关于网桥的总结。

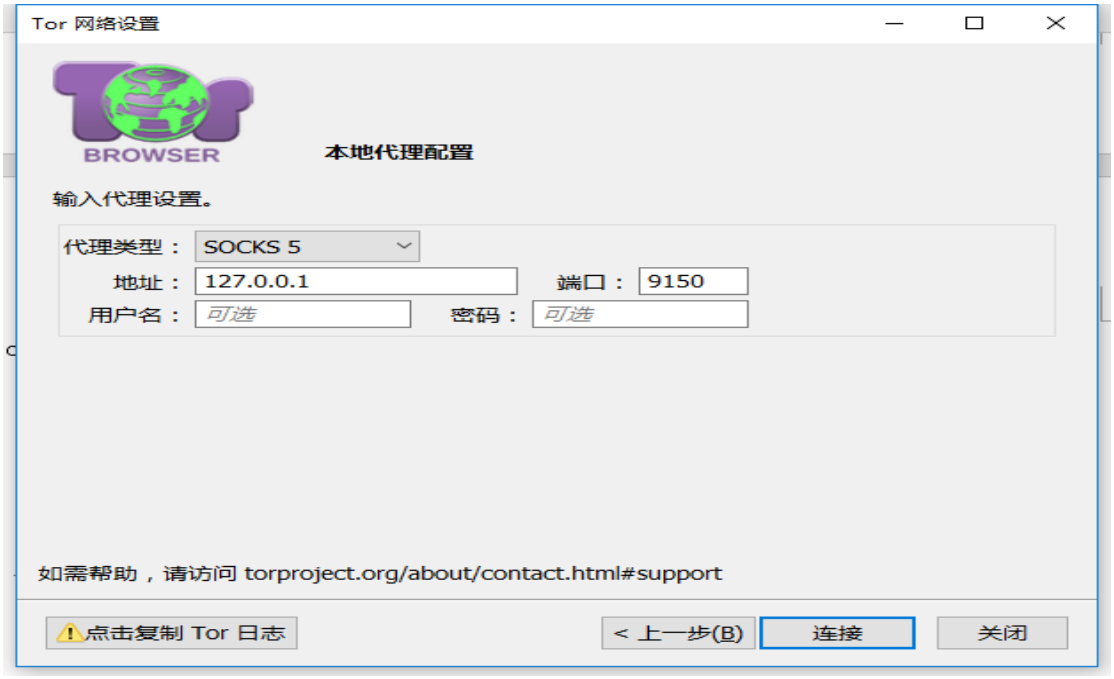

Tor 网络设置
BROWSER
Tor 网桥配置
可以使用集成的网桥，也可以获取网桥，以自定义方式手动输入。 每种网桥使用不同的方法以避免被封锁。如果某个网桥不工作，请换用另一种重试。
使用集成的网桥连接
传输类型： obfs4 (推荐)
输入自定义网桥
帮助(H)
输入一个或多个网桥中继(一行一个)。
类型 地址:端口
如需帮助，请访问 torproject.org/about/contact.html#support
< 上一步(B)
下一步 >(N)
关闭

3、选择代理：

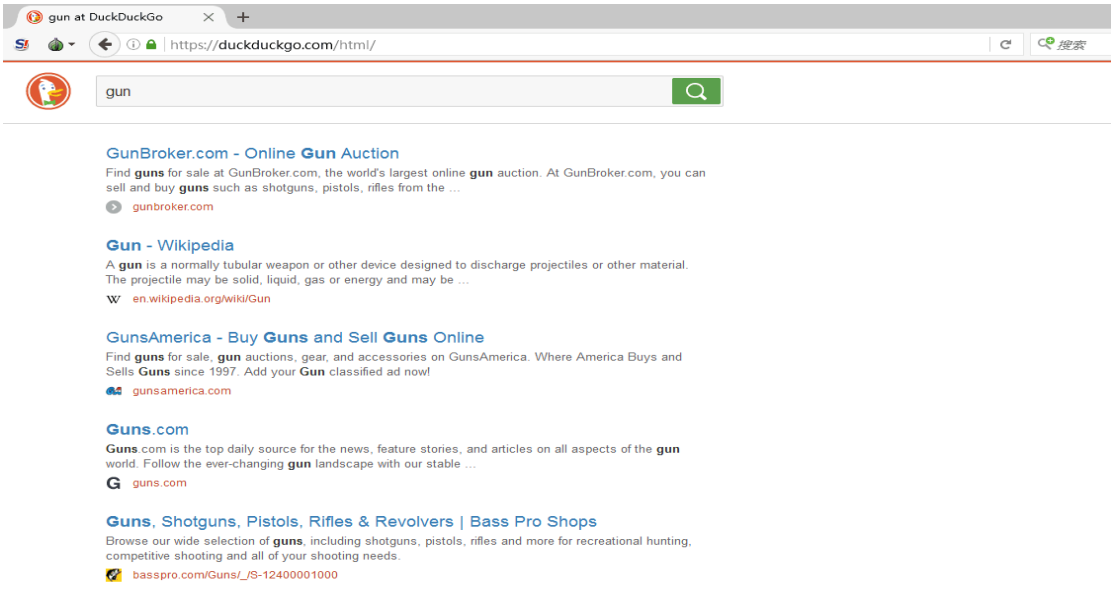

Tor 网络设置
BROWSER
本地代理配置
是否需要本地代理访问互联网？
是
否
大多数情况下不需要使用本地代理，但在使用公司、校园等网络时这可能必需。
如果您不确定此问题的答案，请查看其他网络浏览器或者您的系统中的网络设置，检查其是否使用了一个本地代理。
如需帮助，请访问 torproject.org/about/contact.html#support
点击复制 Tor 日志
< 上一步(B)
下一步 >(N)
关闭

按照 tor 的提示，在校时应该选择“是”。点击下一步。

4、配置协议、地址和端口



可以选择不同的协议：SOCKS4,SOCKS5,HTTP/HTTPS,地址不变，但是端口需要改变，具体情况还要参考 tor 的版本，一般是 9050、9150。
友情提示：接下来连接即可，网速比较慢，需要耐性。
连接成功打开页面如下：



(八)参考资料：

1、维基百科——Tor https://zh.wikipedia.org/wiki/Tor

2、Tor online——https://www.torproject.org/

3、Tor，让你变身互联网“黑影人”——http://www.guokr.com/article/438689/

4、新一代 Tor 发布，它牛在哪里？——http://www.oschina.net/news/74035/tor-release-it-cattle