Kali Linux 渗透测试

The quieter you become，the more you are able to hear

第十四章 流量操控与隧道

苑房弘 Fanghong.yuan@163.com

流量操控技术

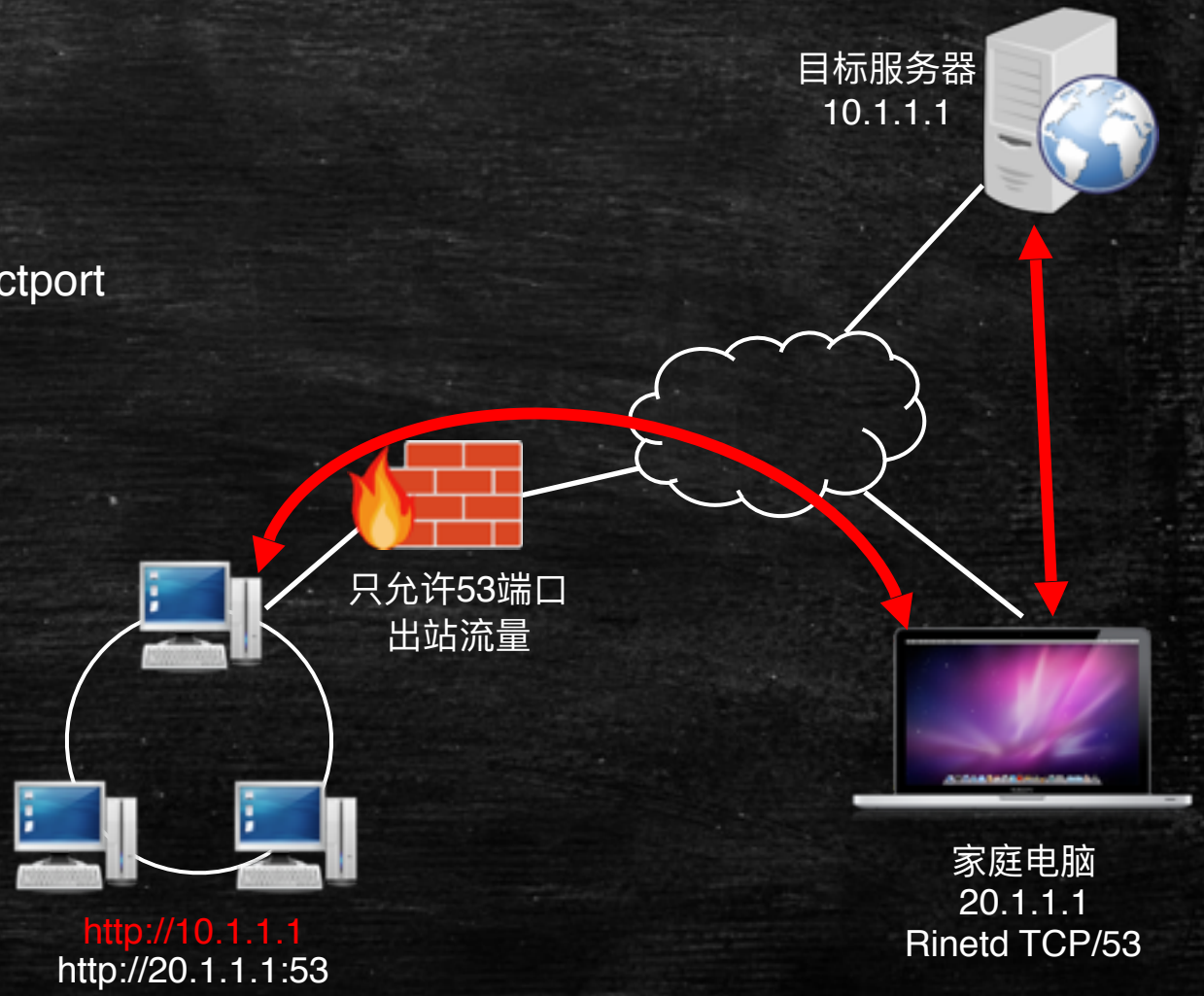
- Traffic manipulation technique
- 渗透测试中经常遇到访问受限的网络环境
- 使用隐蔽的手段逃避安全检查措施和溯源追踪
- 证明看似严格的访问控制仍然存在弱点
- 在非受信任的网络中实现安全的数据传输
- 部分概念的实现过程略有烧脑

流量操控技术

- 重定向（Redirection）
  - IP、Port
- 隧道（Tunneling）
  - 在不受信任的网络环境中实现安全的通信
  - 通常使用多种加密技术建立通信隧道
  - 点到点（IP2IP）、端到端（Port2Port）隧道
  - VPN：pptp、l2tp、IPSec、SSL vpn
- 封装（encapsulation）
  - 通常结合在隧道中使用，使用一种协议封装一种协议（RPC o http、VoIP）
  - 使用网关设备实现不同类型网络的互联互通

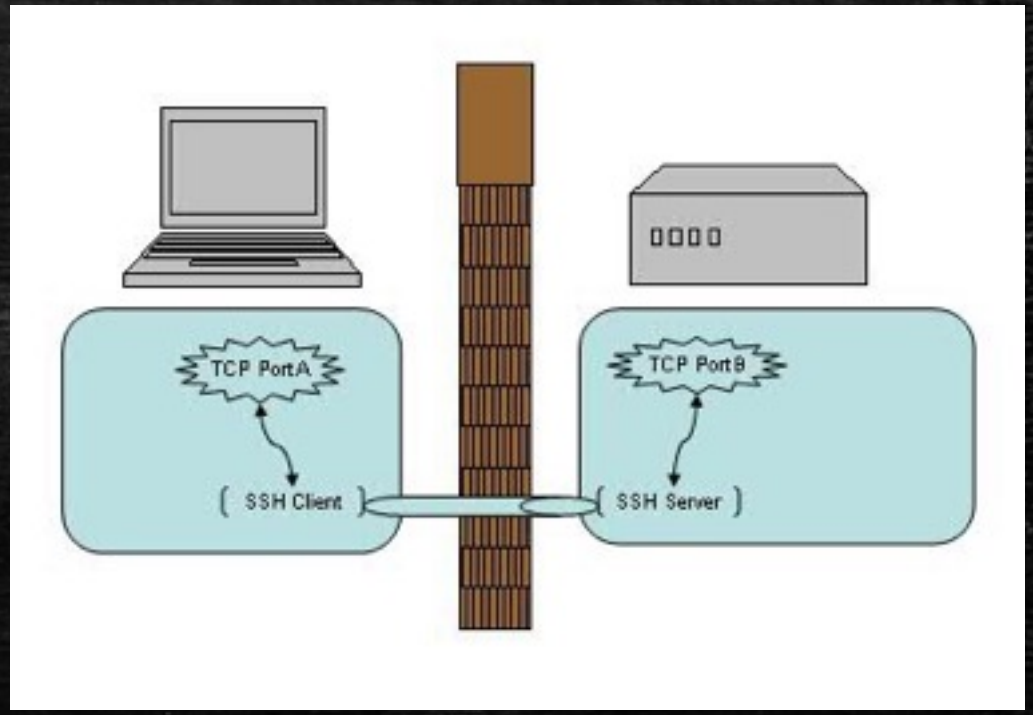
重定向

- Rinetd
  - 安装：apt-get install rinetd
  - 配置：/etc/rinetd.conf
    - bindadd bindport connectadd connectport
  - 运行：rinetd
- 应用场景
  - 重定向web流量，突破上网限制
  - 远程桌面重定向
  - NC重定向获得shell
  - 不兼容FTP等二次连接的协议
- 安装monowall防火墙



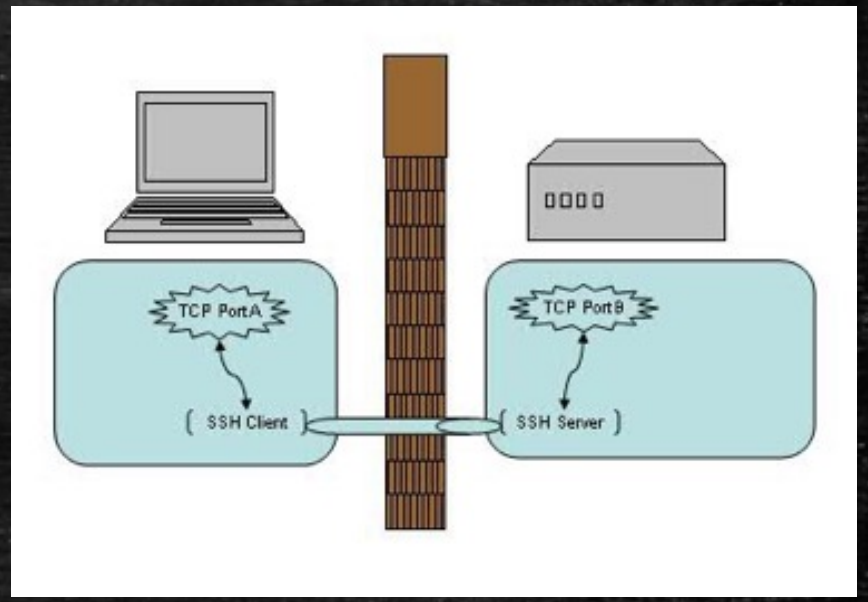
SSH 隧道

- SSH支持双向通信隧道
  - 将其他 TCP 端口的通信通过 SSH 链接来转发
  - 用SSH作为传输层协议，对流量自动加解密
  - 突破防火墙访问规则的限制，可用于翻墙
- SSH 本地端口转发
  - 使效果类似于rinetd
  - 将一本地端口与远程服务器建立隧道



SSH 隧道

- 建立双向安全隧道
  - 将其他 TCP 端口的通信通过 SSH 链接来转发
  - 用SSH作为传输层协议，对流量自动加解密
  - 突破防火墙访问规则的限制，可用于翻墙
- 本地端口转发
  - 本机侦听端口，访问转发到远程主机指定端口
- 远程端口转发
  - 远程侦听端口，访问转发到本机主机指定端口
- 动态隧道模式



SSH 本地端口转发

- 将一本地端口与远程服务器建立隧道
- /etc/ssh/sshd_config
  - PermitRootLogin yes
  - Port 53
  - PasswordAuthentication yes
- service ssh restart

SSH 本地端口转发

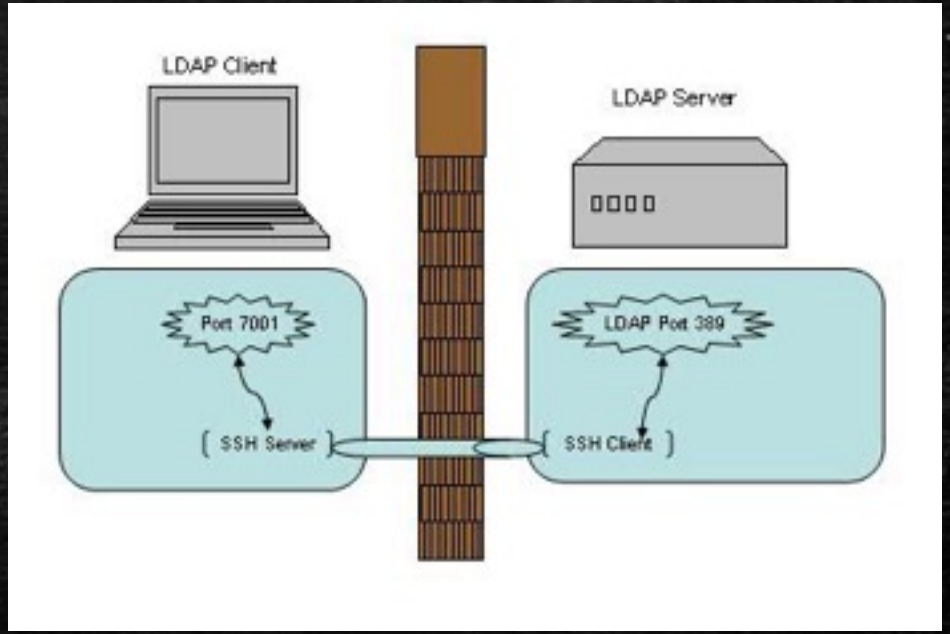
- ssh -fCNg -L <listen port>:<remote ip>:<remote port> user@<ssh server> -p <ssh server port>
- ssh -fCNg -L <listen port>:localhost:<remote port> user@<ssh server> -p <ssh port>
- -f　后台运行进程
- -N　不执行登陆 shell
- -g　复用访问时作为网关，支持多主机访问本地侦听端口
- 网管模式转发RDP、NC shell

SSH 本地端口转发

- 端口转发基于建立起来的SSH隧道，隧道中断则端口转发中断
- 只能在建立隧道时创建转发，不能为已有隧道增加端口转发

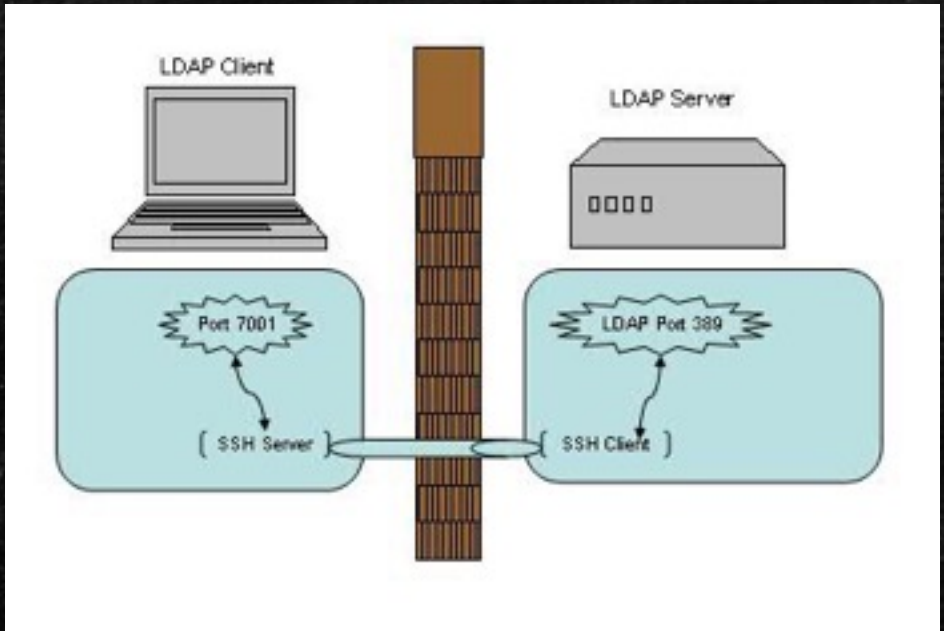
SSH 远程端口转发

- 由于ACL等原因，SSH与应用连接建立方向相反
- 本地端口转发
  - SSH客户端+应用客户端位于FW一端
  - SSH服务端+应用服务端位于另一端
- 远程端口转发
  - SSH客户端、应用客户端位于FW两端
  - SSH服务端、应用服务端位于FW两端



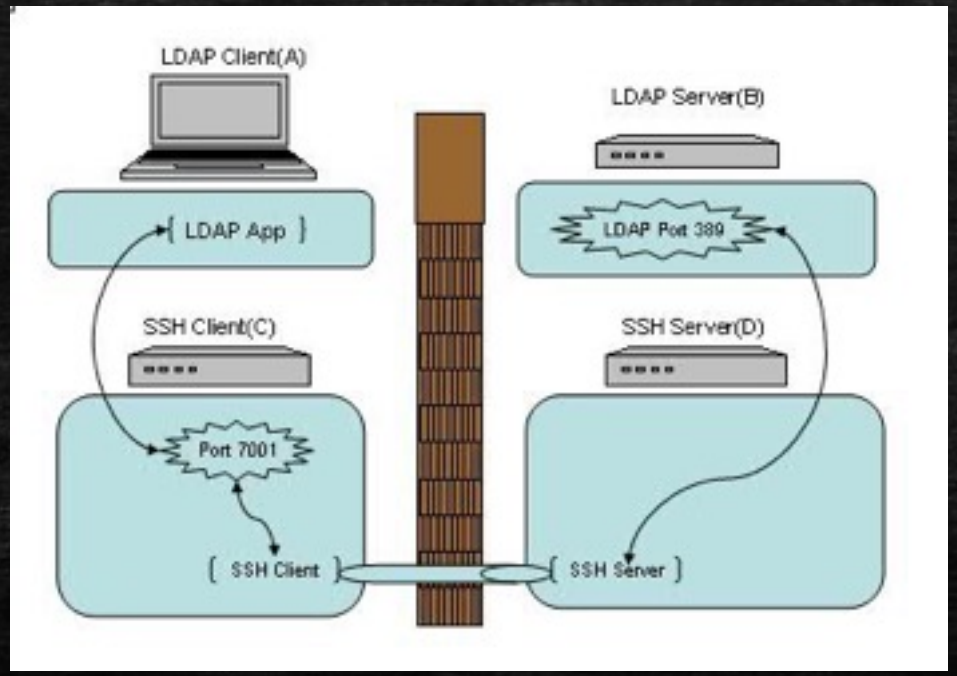
SSH 远程端口转发

- ssh -fNg -R <listen port>:<remote ip>:<remote port> user@<SSH server> -p <ssh server port>
- 之所以称为远程，是因为SSH侦听端口开在远程的SSH Server上
- 侦听端口永远开在应用客户端一方



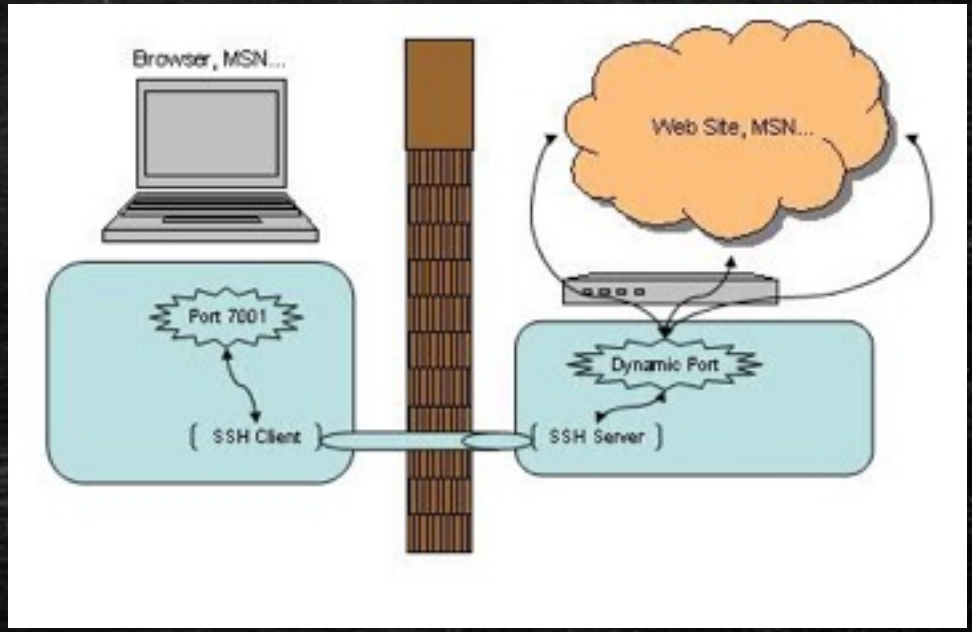
SSH 远程端口转发

- WEB、RDP、NC应用端口转发演示
- （A）<-> (C) 以及 (B)<->(D)之间通信未加密，可嗅探



SSH 动态端口转发

- 本地、远程端口转发都需要固定应用服务器IP、Port
  - 应用端口繁多，逐个转发效率低
  - 某些应用不固定端口
  - 某些网站不支持IP直接访问
  - 使用非受信网络上网时保护流量不被嗅探
- 本地侦听socks4/5代理端口
  - 由SSH server决定如何转发
  - 作为翻墙代理使用
  - 配置客户端代理（浏览器）
  - 使用proxychains支持无代理客户端
  - ssh -CfNg -D 7001 root@1.1.1.1 -p 2121

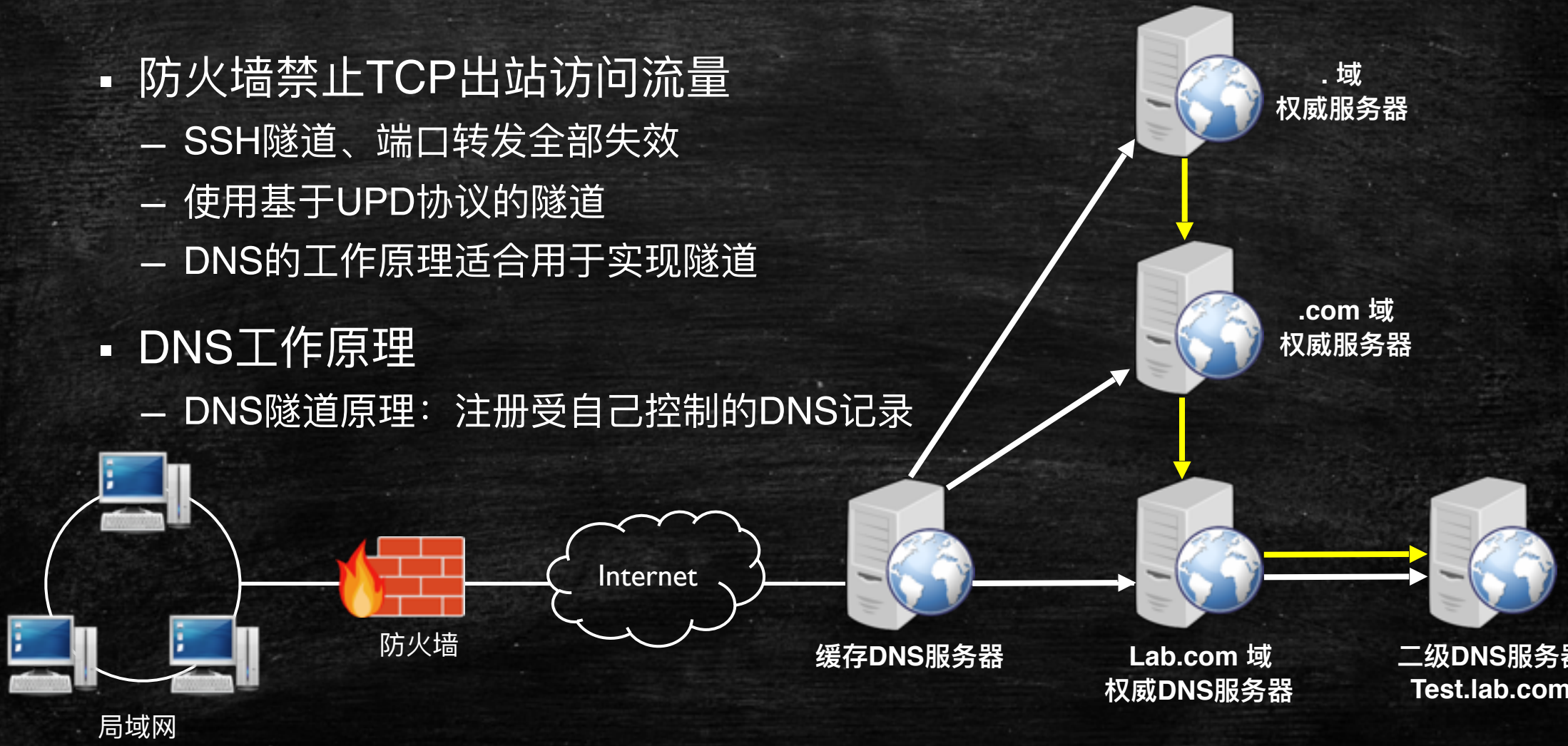


X 协议转发

- 远程登陆Linux GUI运行图形化界面工具
  - VNC
  - X Windows
- 防火墙限制访问时
  - 基于SSH的X转发
  - ssh -X user@1.1.1.1 -p 53

DNS协议隧道

- 防火墙禁止TCP出站访问流量
  - SSH隧道、端口转发全部失效
  - 使用基于UPD协议的隧道
  - DNS的工作原理适合用于实现隧道
- DNS工作原理
  - DNS隧道原理：注册受自己控制的DNS记录



DNS协议隧道——dns2tcp

- Dns2tcp
  - 利用合法DNS服务器实现DNS隧道
  - C/S（dns2tcpc / dns2tcpd）结构
  - 通过TXT记录加密传输数据（A记录长度有限）
  - 隧道建立后保持连接
  - 默认记录生存时间TTL值为 3 秒
- 安装
  - apt-get install dns2tcp
  - Kali 默认安装

DNS协议隧道——dns2tcp

- 服务端配置文件
  - /etc/dns2tcpd.conf
  - .dns2tcprcd
  - 资源可以是其他地址

```
listen = 0.0.0.0
port = 53
user = nobody
chroot = /tmp
key = password123
domain = test.lab.com
resources = ssh:127.0.0.1:22 , smtp:127.0.0.1:25 , socks:127.0.0.1:
1082 , https:127.0.0.1:8087 , http:127.0.0.1:3128
```

- 启动
  - dns2tcpd -F -d 1 -f /etc/dns2tcpd.conf
  - F：前端运行
  - d：debug level 1-3
  - f：指定配置文件

DNS协议隧道——dns2tcp

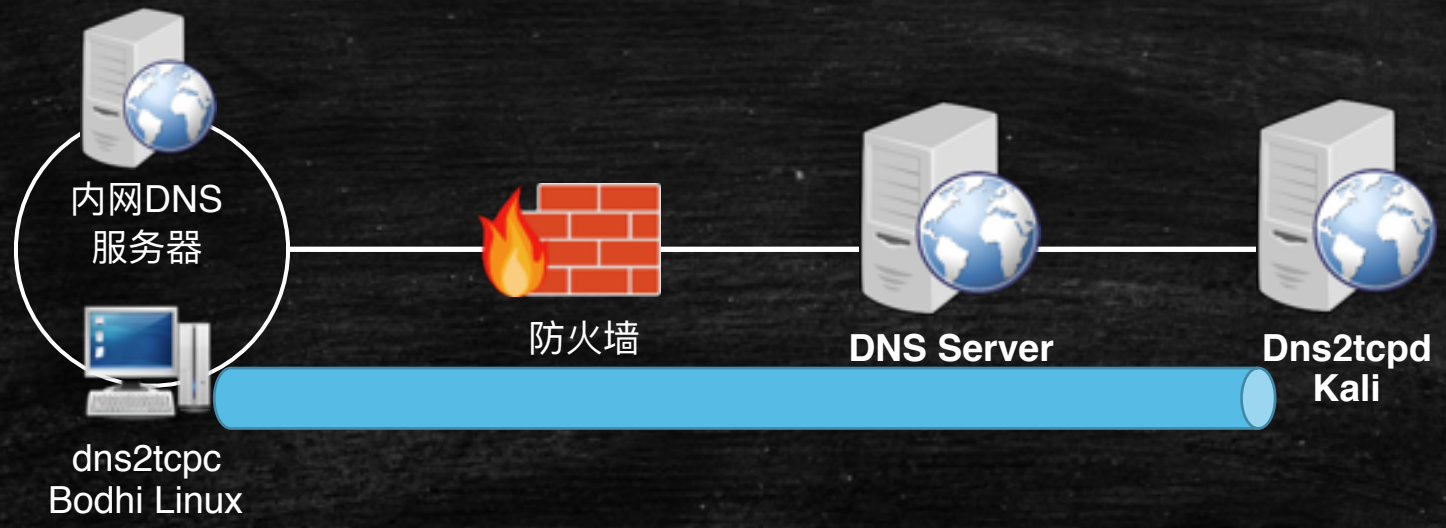
- 演示环境-1
  - Win 2003：安装DNS服务；配置转发器；创建区域lab.com；指派二级域test.lab.com，NS记录指向Kali
  - 防火墙：只允许出站UDP 53端口流量
  - Bodhi Linux：
    - 安装dns2tcp、wireshark、firefox
    - dns2tcpc -c -k pass -d 1 -l 2222 -r ssh -z test.lab.com

DNS协议隧道——dns2tcp

- 资源访问
  - 本地SSH资源
  - 远程http资源
  - http代理（squid）
    - apt-get install squid3
  - https代理
  - 隧道嵌套
    - 基于SSH资源将SSH动态端口转发隧道嵌套于DNS隧道中
    - ssh -CfNg root@127.0.0.1 -p 2222 -D 7001
    - XP IE、Firefox 使用嵌套的Socks代理上网
- 抓包分析DNS隧道通信

DNS协议隧道——dns2tcp

- 演示环境-2
  - FW限制只允许内网DNS服务器访问外网指定DNS服务器UDP 53端口
  - 内网DNS服务器：安装DNS服务，配置转发器到外网DNS服务器z



DNS协议隧道——iodine

- 基于DNS查询的隧道工具
- 与同类工具相比的优点
  - 对下行数据不进行编码，因此性能优
  - 支持多平台：Linux、BSD、Mac OS、Windows
  - 最大16个并发连接
  - 强制密码支持
  - 支持同网段隧道IP（不同于服务器、客户端网段）
  - 主持多种DNS记录类型
  - 丰富的隧道质量检测措施

DNS协议隧道——iodine

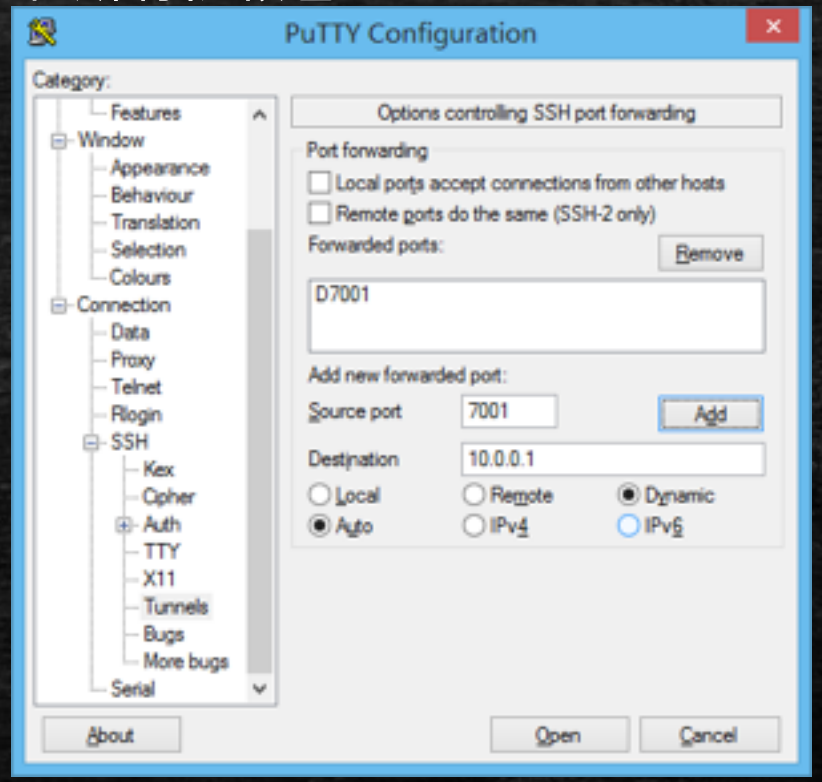
- 运行服务器端
  - iodined -f -c 10.0.0.1 test.lab.com
  - -f：前端显示（可选）
  - -c：不检查客户端IP地址
  - IP：服务器端的隧道IP地址

DNS协议隧道——iodine

- 运行客户端
  - iodine -f test.lab.com
  - curl --socks5-hostname 127.0.0.1:7001 http://www.sina.com
- 隧道网络接口
  - 不基于资源的通用隧道，如同本网段内两台相邻的主机
  - 服务器端和客户端分别生成隧道网络接口dns0
  - 隧道两端接口的IP地址应不同于客户端和服务器端网段
  - 基于此隧道可嵌套其他隧道技术
    - ssh -CfNg -D 7001 root@10.0.0.1

DNS协议隧道——iodine

- 安装TAP网卡驱动
  - https://openvpn.net/index.php/open-source/downloads.html
  - 只安装 TAP Virtual Ethernet Adapter 和所有依赖包
- Windows客户端
  - http://code.kryo.se/iodine/
  - iodine -f test.lab.com
- 建立SSH隧道



NCAT

- 被称为众多NC衍生版软件中最优的选择
- 代理功能
  - ncat -l 8080 --proxy-type http --proxy-auth user:pass
- Broker中介功能
  - AB不同但AC、BC互通
  - 服务器：ncat -l 333 --broker
  - 客户端之间发送任何信息都会被hub到其他客户端
  - 批量执行命令：ncat 1.1.1.1 --sh-exec "echo `pwd`“
  - 批量传文件： ncat --send-only 1.1.1.1 < inputfile

SOCAT

- 被称为nc++（增强增强版的nc）
  - 双向数据流通道工具
- 连接端口
  - Socat - tcp:1.1.1.1:80
- 侦听端口
  - socat - tcp4-listen:22　/　socat - tcp-l:333
- 接收文件
  - socat tcp4-listen:333 open:2.txt,creat,append
- 发送文件
  - cat 1.txt | socat - tcp4:1.1.1.1:333

SOCAT

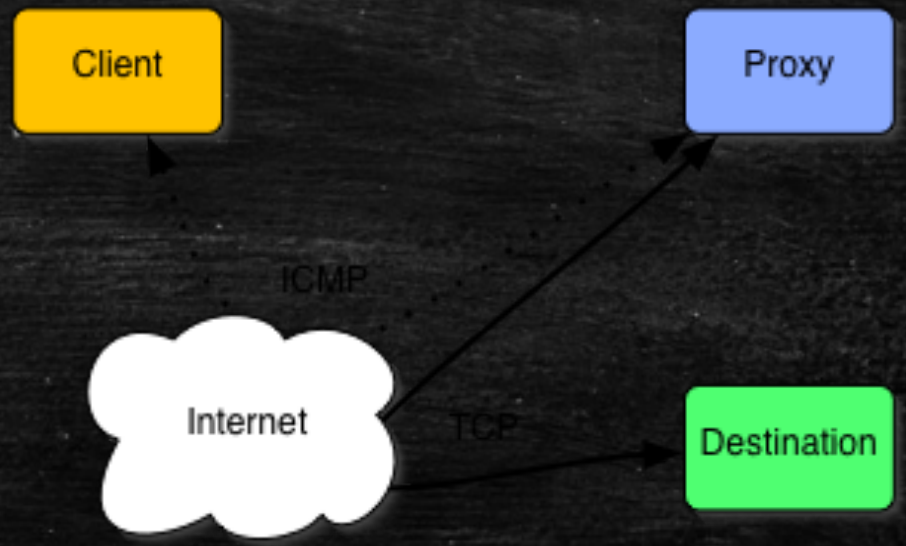
- 远程shell——服务器端
  - socat tcp-l:23 exec:sh,pty,stderr
- 端口转发
  - socat tcp4-listen:22,fork tcp4:1.1.1.1:22
- 远程执行命令
  - 服务器：socat - udp-l:2001
  - 客户端：echo “`id`” | socat - udp4-datagram:1.1.1.1:2001
- UDP 全端口任意内容发包
  - for PORT in {1..65535}; do echo “aaaaa" | socat - UDP4-DATAGRAM:1.1.1.1:$PORT; sleep .1; done

SOCAT

- 二进制编辑器
  - echo -e "\0\14\0\0\c" | socat -u - file:/usr/bin/squid.exe,seek,seek=0x00074420
- 简单的WEB服务器
  - socat -T 1 -d -d TCP-L:10081,reuseaddr,fork,crlf SYSTEM:"echo -e \"\\\"HTTP/1.0 200 OK\\\nDocumentType: text/plain\\\n\\\ndate: \$\(date\)\\\nserver:\$SOCAT_SOCKADDR:\$SOCAT_SOCKPORT\\\nclient: \$SOCAT_PEERADDR:\$SOCAT_PEERPORT\\\n\\\"\"; cat; echo -e \"\\\"\\\n\\\"\""

ptunnle

- Ping tunnel ICMP隧道工具
  - 通过ICMP echo(ping requests)和reply(ping reply) 实现隧道
  - 适用于防火墙只允许ping出站流量的环境
  - 支持多并发连接、性能优
  - 支持身份验证
  - 需要root权限
  - 支持抓包
    - Windows：winpcap
    - Linux：libpcap
  - 工作过程
    - Proxy 、Client 、Destination

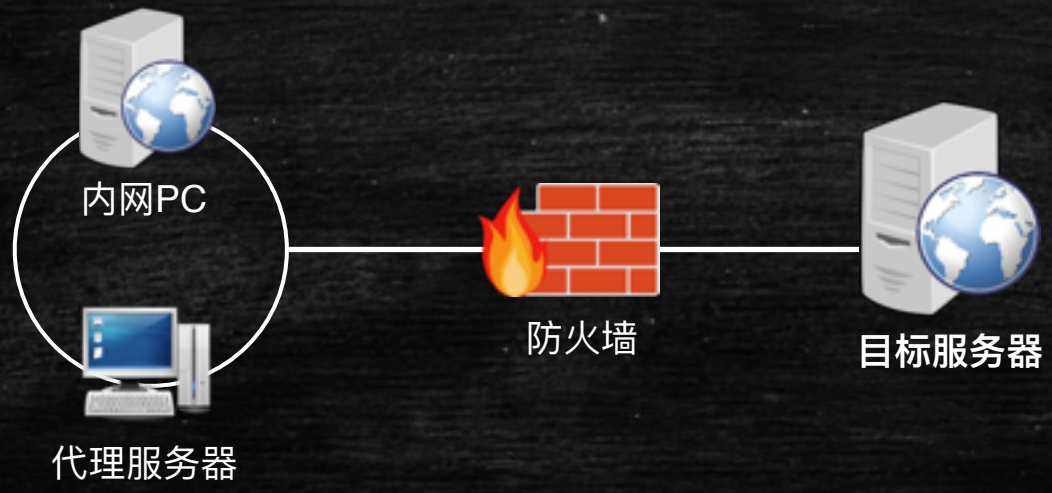


ptunnle

- 服务器
  - ptunnel -x 1234
- 客户端
  - sudo ptunnel -p *proxy* -lp 2222 -da *destination* -dp 22 -x 1234
- 嵌套SSH隧道
  - ssh -CNfg -D 7000 root@127.0.0.1 -p 2222
- ptunnel 直到目前的最新版仍存在拒绝服务漏洞
  - 0.72

proxytunnle

- 通过标准的HTTP / HTTPS代理创建隧道的工具
- 通过HTTP CONNECT 方法封装信息
- 适用于内网使用代理并且防火墙只允许代理服务器上网的场景
- 无法创建DNS隧道和ICMP隧道

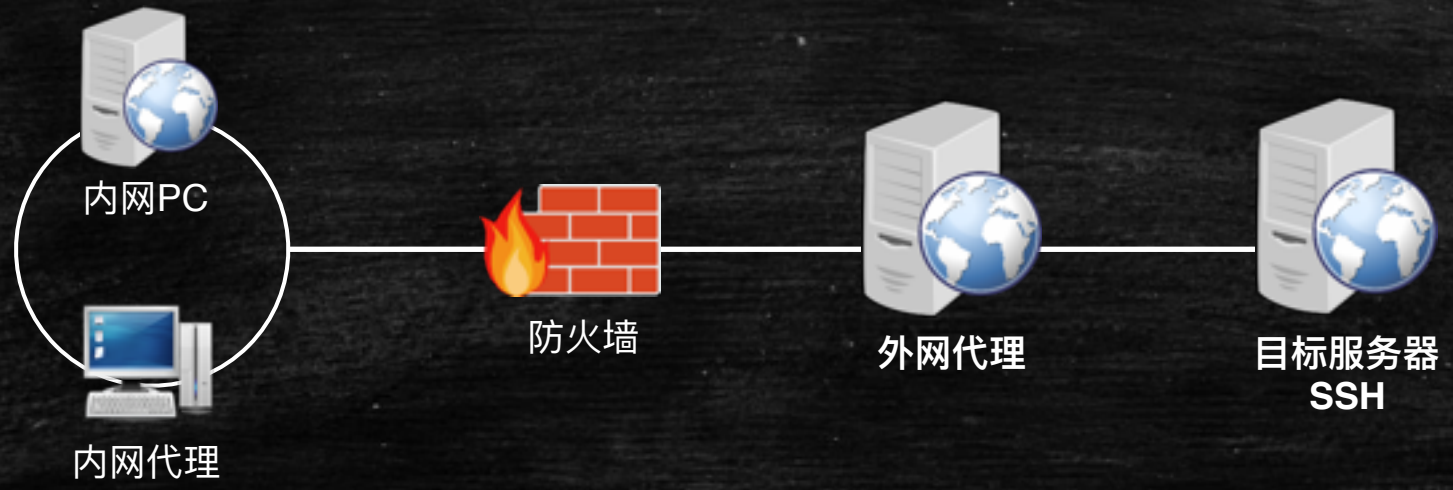


Proxytunnle

- 实验一：将外网资源映射为内网指定端口
  - 内网安装squid3代理、proxytunnle
    - vi /etc/squid3/squid.conf
    - /^http_port
    - /^http_access
    - /^acl
  - 创建隧道：proxytunnle -a 80 -p 127.0.0.1:3128 -d 192.168.1.1:80

Proxytunnle

- 实验二：外网资源非防火墙允许端口
  - 修改目标资源侦听端口可能无法躲避防火墙深层检测
  - 外网安装squid3代理服务器并侦听80端口
  - 创建隧道：proxytunnle -a 80 -p 127.0.0.1:3128 -r 192.168.1.1:80 -d 192.168.1.1:22



Proxytunnel

- 实验三：ssh客户端配置自动创建代理链隧道
  - vi ~/.ssh/config

```
Host 192.168.1.1
    Hostname 192.168.1.1
    ProtocolKeepAlives 30
    ProxyCommand /usr/bin/proxytunnel -p 1.1.1.1:3128 -r 192.168.1.1:80 -d %h:%p
```

sslh

- 端口分配器
  - 根据客户端第一个包检测协议类型
  - 根据协议检测结果将流量转发给不同目标
  - 支持HTTP, HTTPS, SSH, OpenVPN, tinc, XMPP和其他可基于正则表达式判断的人和协议类型
  - 适用于防火墙允许443端口入站访问流量的环境

sslh

- 配置文件
  - /etc/default/sslh

```
root@K:/etc/init.d# cat /etc/default/sslh
# Default options for sslh initscript
# sourced by /etc/init.d/sslh
RUN=yes
DAEMON=/usr/sbin/sslh
DAEMON_OPTS="--user sslh --listen 1.1.1.10:443 --ssh 127.0.0.1:22 --ssl 1.1.1.11:443
 --http 127.0.0.1:80 --pidfile /var/run/sslh/sslh.pid"
```

- 安装HTTPS站点
  - 安装IIS服务、证书服务
  - 部署HTTPS站点
- 启动本地HTTP服务
- 防火墙端口映射TCP/443

stunnel4

- 无需修改原代码的情况下将TCP流量封装于SSL通道内
- 适用于本身不支持加密传输的应用
- 支持openssl安全特性
- 跨平台
- 性能优

stunnel4

- 安装内网Stunnel4服务器
- 服务器端配置
  - 生成证书：openssl req -new –days 365 -nodes -x509 -out /etc/stunnel/stunnel.pem -keyout /etc/stunnel/stunnel.pem
  - 创建配置文件 /etc/stunnel/stunnel.conf
    - cert = /etc/stunnel/stunnel.pem
    - setuid = stunnel4
    - setgid = stunnel4
    - pid = /var/run/stunnel4/stunnel4.pid
    - [mysqls]
    - accept = 0.0.0.0:443
    - connect = 1.1.1.11:3306

stunnel4

- Stunnel4自动启动
  - /etc/default/stunnel4
    - ENABLED=1
- 启动stunnel4服务端
  - service stunnel4 start
- 防火墙规则
  - 端口映射TCP/443端口到stunnel4服务端TCP/443
  - 设置防火墙规则
- Stunnel4客户端

stunnel4

- 安装Stunnel4客户端
  - Kali自带
- 客户端端配置
  - 创建配置文件 /etc/stunnel/stunnel.conf
    - client = yes
    - [mysqls]
    - accept = 3306
    - connect = 192.168.1.11:443
- 客户端自动启动
  - /etc/default/stunnel4
    - ENABLED=1

stunnel4

- 启动客户端服务
  - service stunnel4 stop / start
- Mysql客户端连接服务器
  - mysql -u root -h 127.0.0.1
- 抓包对比隧道前后差异

Thanks!