Hosting a Hacking Challenge - CTF-style

Lexi Pimenidis

Lehrstuhl für Informatik IV
RWTH Aachen

21. Januar 2006

Hacking Challenge – CTF-style

- "Hacking" like in "Hacking"
- CTF is for *Capture the Flag*

- There are several "teams", each protecting a "server" containing "services" (identical for each team)
- Like socker: keep your own box clean while hitting the other boxes
- Thus analyse the services, correct them and sploit the vulns on the *other* team's boxes

How does a CTF work?

- Services are tested by a Scorebot for functionality
- Small pieces of information (“*flags*”) are left in secured places and retrieved later on
- Protect own, hack other teams to get their flags
- Scores are awarded for protected flags and captured flags
- **New:** Advisories are also scored

Wasting time to play?

- To Win
- To learn
- To have fun

Wasting time to play?

- ~~To Win~~
- To learn
- To have fun

What do you learn?

- Training of programming and defense skills

- Prepare for the worst in real life
- Work under pressure
- Where software breaks in real scenarios

- Got to see all kind of exploits
- Getting hit by all kind of exploits
- Inventing innovative defense techniques

What do you learn **not**?

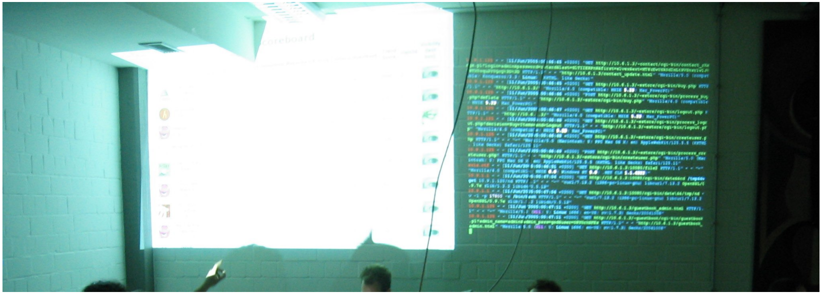
- Real life pentesting

Organisorz

- Need to have a lot(!) of spare time
- Expertise is a plus
- Accurate, careful and vicious
- Be responsive and always online (email, irc, jabber, phone, . . .)

Pl4y0rz

- Need to know at least basic exploitation and defense techniques
- Work under pressure
- Good communication and team work required
- Size of teams is crucial, 5 to 10 seems optimal
- (Need to be able to read and understand texts)



The Rules of the Game

- Are rules necessary? Yes!

- Fun is a crucial part: destructive behaviour is forbidden
- Fairness: rules treat all equal (as in all competive sports)
- Good experience with known and open rules
- Fair play is suggested
- Enforcement is typically impossible, rely on unfair behaviour to be reported
- $\Rightarrow$ Logging is suggested

The Rules of the Game

- Are rules necessary? Yes!

- Fun is a crucial part: destructive behaviour is forbidden
- Fairness: rules treat all equal (as in all competive sports)
- Good experience with known and open rules
- Fair play is suggested
- Enforcement is typically impossible, rely on unfair behaviour to be reported
- $\Rightarrow$ Logging is suggested

The Rules

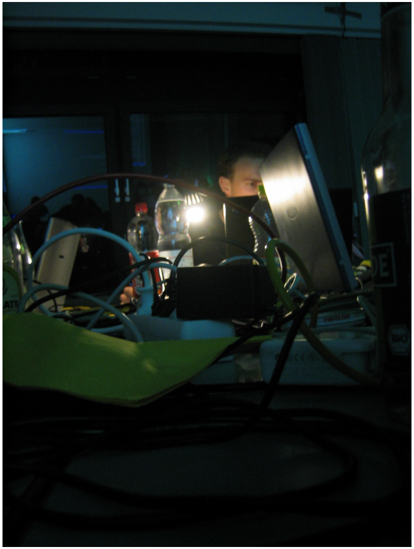
- Cheating on the team size (if restrictions are given)
- Intentionaly supporting other teams
- All computers not from the organizors are legible targets
- No filtering based on IPs (traffic anonymization)
- No unfair filtering on secondary information (HTTP user-agent)
- No automated stack protection techniques, or stuff like grsec, or SELinux
- No (unsophisticated) (D)DoS
- Automated scanning

The OS

- Linux is seen most often
  - No licenses
  - Known to most players
  - Easiest to do well known manipulations
- Others not seen yet (personally)

How much?

- At least one player per "service"
- Allows dedicated analysis and observation
    - op3n kernel-level sploit

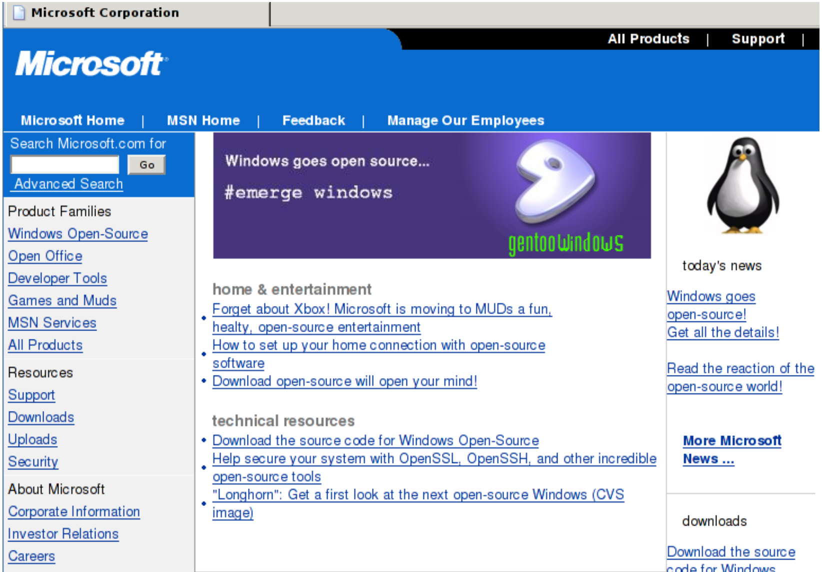


Themes

- Services are typically bound to a theme
- Technically unnecessary, but fun
- Some examples . . .

Themes

CTF: M$ gentoo



Microsoft Corporation

All Products | Support |

Microsoft Home | MSN Home | Feedback | Manage Our Employees

Search Microsoft.com for

Go

Advanced Search

Product Families

Windows Open-Source

Open Office

Developer Tools

Games and Muds

MSN Services

All Products

Resources

Support

Downloads

Uploads

Security

About Microsoft

Corporate Information

Investor Relations

Careers

Windows goes open source...

#emerge windows

home & entertainment

- Forget about Xbox! Microsoft is moving to MUDs a fun, healty, open-source entertainment
- How to set up your home connection with open-source software
- Download open-source will open your mind!

technical resources

- Download the source code for Windows Open-Source
- Help secure your system with OpenSSL, OpenSSH, and other incredible open-source tools
- "Longhorn": Get a first look at the next open-source Windows (CVS image)

today's news

Windows goes open-source! Get all the details!

Read the reaction of the open-source world!

More Microsoft News ...

downloads

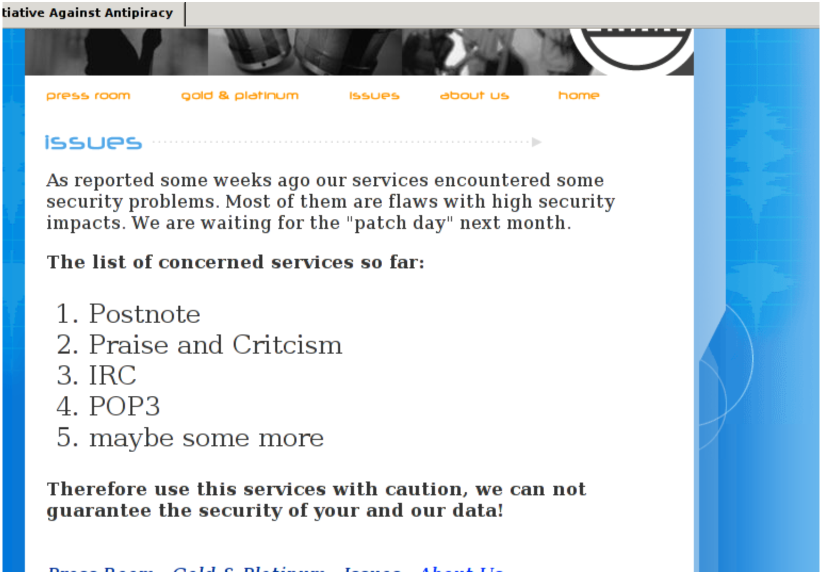
Download the source code for Windows

Themes

iCTF: World Elections

Themes

Cipher: RIAA



issues

As reported some weeks ago our services encountered some security problems. Most of them are flaws with high security impacts. We are waiting for the "patch day" next month.

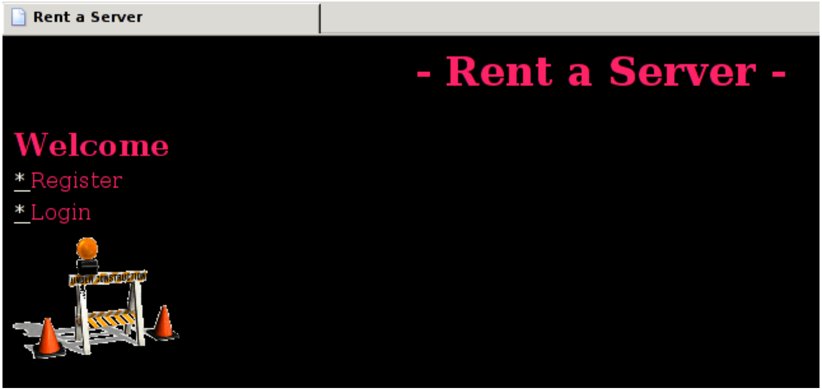
The list of concerned services so far:

1. Postnote
2. Praise and Critcism
3. IRC
4. POP3
5. maybe some more

Therefore use this services with caution, we can not guarantee the security of your and our data!

Themes

op3n: Some Provider



Rent a Server

- Rent a Server -

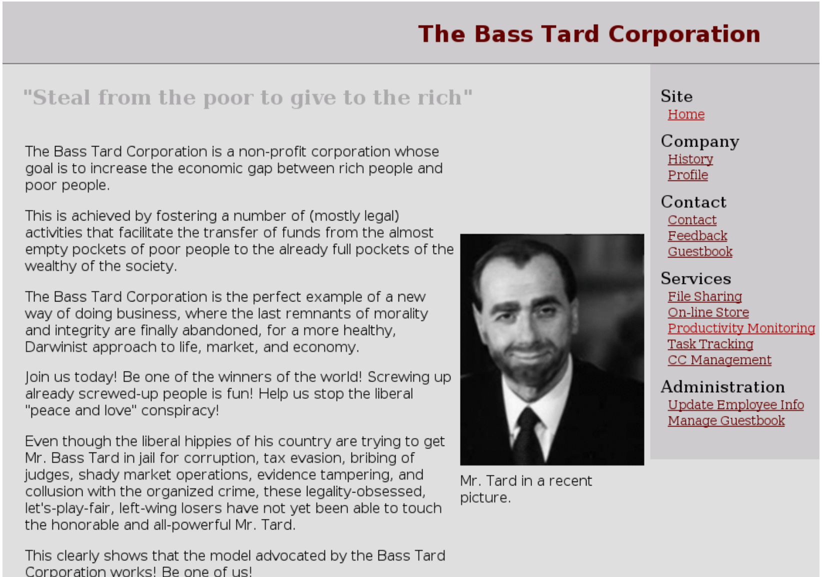
Welcome

* Register

* Login

Themes

iCTF2: Basstard



The Bass Tard Corporation

"Steal from the poor to give to the rich"

The Bass Tard Corporation is a non-profit corporation whose goal is to increase the economic gap between rich people and poor people.

This is achieved by fostering a number of (mostly legal) activities that facilitate the transfer of funds from the almost empty pockets of poor people to the already full pockets of the wealthy of the society.

The Bass Tard Corporation is the perfect example of a new way of doing business, where the last remnants of morality and integrity are finally abandoned, for a more healthy, Darwinist approach to life, market, and economy.

Join us today! Be one of the winners of the world! Screwing up already screwed-up people is fun! Help us stop the liberal "peace and love" conspiracy!

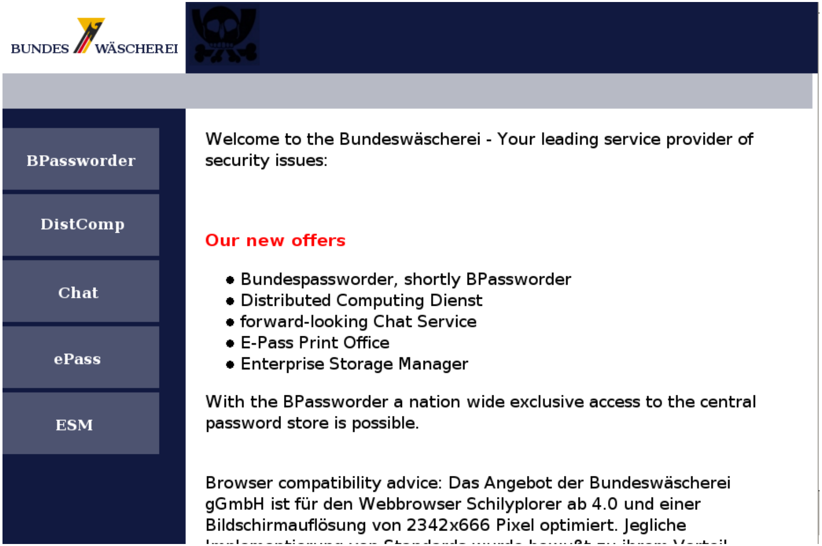
Even though the liberal hippies of his country are trying to get Mr. Bass Tard in jail for corruption, tax evasion, bribing of judges, shady market operations, evidence tampering, and collusion with the organized crime, these legality-obsessed, let's-play-fair, left-wing losers have not yet been able to touch the honorable and all-powerful Mr. Tard.

This clearly shows that the model advocated by the Bass Tard Corporation works! Be one of us!

Mr. Tard in a recent picture.

Themes

22c3: Bundeswäscherei



Welcome to the Bundeswäscherei - Your leading service provider of security issues:

Our new offers

- Bundespassworder, shortly BPassworder
- Distributed Computing Dienst
- forward-looking Chat Service
- E-Pass Print Office
- Enterprise Storage Manager

With the BPassworder a nation wide exclusive access to the central password store is possible.

Browser compatibility advice: Das Angebot der Bundeswäscherei gGmbH ist für den Webbrowser Schilyplorer ab 4.0 und einer Bildschirmauflösung von 2342x666 Pixel optimiert. Jegliche

The Challenges

- Each challenge is typically a single service ⇒ Avoid r00t-xploit
  - But this never works anyway
- Services are connected to the network
- Service type: Out-of-the-Box
  - Problem: Vulns known by Google or experienced players
- Service type: Modified Out-of-the-Box
  - Easy to create diffs and look for changes
- Service type: Custom service
  - Custom protocols
  - Standard protocols

Programming Languages

- The more progamming languages, the better
- Must: PHP, Perl, C, Bash, SQL, Assembler(i.e. reverse engineering)
- Possible: Java, Python, C++
- Unusual: Basic, Pascal

- Unusual combinations can be fun (Webserver with dynamic content in BASH-scripts)

Difficulty of Challenges

- Need to have easy and difficult tasks
  - Pedagogic mission: all players would like to be important
  - (unlike e.g. north american team in CIPHER)
- Avoid too difficult tasks at any costs
  - op3n and Italian CTF had a number of services unexploited $\Rightarrow$ Waste of time for organizors and players
  - e.g. No fun to learn and exploit custom binary protocols
  - Too much source with unclear structure
  - Sources hidden in strange places
- There aren't too easy tasks (unfortunately?)
  - There's always at least one team not fixing the easiest bugs until the end

Designing Challenges (Cook Book, Part 1)

- Start with a clear idea on programming language and service type
- Design service such that storing and retrieving flags can be incorporated
  - Storage and retrievel *can* be done over backd00rs, but that's kind of cheating
  - Best, if service type inherently should protect information
  - $\Rightarrow$ Vulns to circumvent protection
- Create a flawless version of the service, well documented and understood

Designing Challenges (Cook Book, Part2)

- Insert typical vulns
  - varying types and difficulty
  - some well hidden, some obvious (avoid commenting them, though (hinting is OK))
- Insert atypical vulns, if possible
- Vulns should only allow access to flags of this single service
  - Don't destroy flags
  - Don't access other service's flags
  - **NEVER** give r00t-access
- Write RC-script for gameserver control
- Test
- Test
- Test

Perl-Part (Guestbook) (iCTF2)

Command Injection

```
# copy values into paramater names
foreach $para ($query->param) {
  $$para = $query->param($para);
}
if (${user}) {
  $command = "grep ${user} ${guestbook}";
  $usercomments = `${command}`;
}
```

Acquire (iCTF05)

backd00r in redundant files

```
<?php
$from = $_COOKIE['cust_id'];
if (!is_null($from))
{
  if ($from == ip2long($_SERVER['SERVER_NAME']))
  {
    # ... print out all flags
```

SMTP-server (CIPHER)

- Python
- `args[1]` is the part after `RCPT TO:`

Command Injection and dir traversal

```
self.rcpt_to = args[1]
self.filename = PATH+'/'+args[1]
fa = os.popen('ls -d %s.*' % self.filename).readlines()
```

SMTP-server (CIPHER)

- The pain continues

Command Injection (ouch..)

```
def write2file(self):
  if self.data != '':
    print 'saving data to ', self.filename
    f=os.popen('cat > '+self.filename,'w')
    print >>f, '%s' % self.data
    f.flush()
    f.close()
```

feedback (iCTF1)

- PHP (and some Perl)

Uninitialized variables

```
<?php
$name = $_POST['name'];
$comment = $_POST['comment'];
$candidate = $_POST['candidate'];
if (!file_exists("feedback")) {
  [[ ... some more code here ... ]]
}
else if ($admin)
{
  system("ls feedback/*");
```

Chatserver (CIPHER)

- Perl
- Simple Chatserver on port 2323
- Multi channel with IRC-style commands
- Flags are posted as topics of channels with a random name

Sending data between clients

```
foreach my $peer (keys %child) {
  if (($channel eq "") ||
      ($channel =~ /$child{$peer}{"CHANNEL"}/)) {
        $child{$peer}{"BUFFER"} .= "$line";
  }
};
```

register (iCTF1)

- Perl
- update.pl contains trivialy exploitable SQL-injections (no checks at all)

SQL-Injection in review.pl

```
$password = param('password');
$password =~ s/[^a-zA-Z0-9']//g;
[[ ... ]]
my $query = "SELECT * FROM registration.voters WHERE
        wwid = '$wwid' AND password = '$password';";
[[ ... ]]
if ($sth->rows != 1) {
  [[ error ]]
} else {
  [[ you win! ]]
```

K3rn3l (op3n)

- C-Frontend for network and Kernel-level backend
- Bug 1: corruption of data (in kernel code)

Bug 2: read data through backdoor (in kernel code)

```
static int mod_get_flag (char *buf, int *len){
  int r;
// [[ length checks omitted ]]
  if (*len > 0) {
    r = copy_to_user(buf, FLAG, FLAG_SIZE);
    *len = FLAG_SIZE;
  } else {
    r = copy_to_user(buf, FLAG_PLAIN, flag_plainsize);
    *len = flag_plainsize;
  }
return r;
```

distcomp (22c3)

Remote code execution by design

```
len = read(c_sock, buffer, 256);
if(strncmp(buffer, "CODE ", 5) == 0) {
  ticket = buffer + 5;
  ptr = strchr(ticket, ' ');
  *ptr = '\0';
  ptr++;
  code = ptr;
  buffer[len] = '\0';
  // [[ some file action ]]
    ret = exec_code(code, len - (buffer - code));
```

electd (iCTF1)

- Reverse-engineering, SSL-layer
- Was done offline

Host-level security

- Additionaly challenges are weak configured services
- . . . weak passwords
- . . . backd00rs
- Possible, but take into account as a service (announce?)

Where to host a challenge?

- Local play is easiest for organizers, most difficult for other teams
    - Allows more interesting variants, e.g. real hardware
    - Easier enforcement of rules
- Remote play is hard for organizers, easier for teams
    - Deployment of services is difficult
    - $\Rightarrow$ Virtualization with VMware
    - Teams drop out due to connectivity (ping of ¿2000ms)

When to host a challenge?

- Take care of exams and holidays
- Timezone! (sunlight saving?)

Common pitfalls – Non technical

- Nigeria
- Bologna
- “Collateral damage”
- Too much or too less advertisement (number of teams)

Common pitfalls – Technical

- Drop out due to hardware failures (iCTF1)
- Drop out due to software failures (iCTF1)
- Avoid side channels on flags
    - Computationally created flags
    - Predictable secrets
    - Sniff traffic
- Keyboard layout and fsck
- .bash_history
- Deleted files?

Fin

Questions?