Hack the Puffy

OpenBSD advocacy through CTF

Jason Testart

February 11, 2019

About the Speaker

- 20 years in IT (higher education)
  - Unix system/security administration (4 years)
  - Technical management (5 years)
  - Cybersecurity leadership/CISO (11 years)
- 6 years of volunteering (~300 hours per year)
- **Creator of dad jokes**

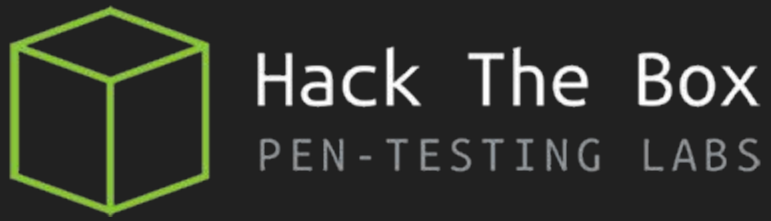
PostgreSQL’s fsync() surprise

- https://lwn.net/Articles/752093/
- https://lwn.net/Articles/752063/
- A problem involving assumptions
- Affects Linux/{Open,Net}BSD but not FreeBSD

My introduction to CTF

https://www.holidayhackchallenge.com/

My Spring (2018) Fling



A little about Hack the Box

- Need to “hack” in invite code to create an account.
- Private network of virtual machines.
  - 20 “Active” at once.
  - Mostly Linux, some Windows, rarely others.
  - VMs are submitted by community members.
  - Moderators review submissions then approve/decline.
  - Typically once a week, a VM is retired and a new VM is made “Active”.
  - Two Flags:
    i. File in user’s home directory
    ii. File in root’s home directory
- Other CTF challenges
  - crypto, forensics, etc...

Contributing back...

- Observed a lack of LDAP services on HTB
- Wanted to see a PASS-THE-HASH technique with SAMBA on *nix
- Do people **really** know OpenSSH?
  - Test my own understanding of `AuthorizedKeysCommand` and `AuthorizedPrincipalsCommand` by building something that uses these features

Learning Objectives

1. Understand how to enumerate an LDAP directory, including determining the root DSE.
2. Understand how to use the PASS-THE-HASH technique with SAMBA on *nix
3. Understand how SSH Certificate Authorities work.

Why OpenBSD?

1. Straightforward operation. Minimal userland processes out of the box.

   Linux has become a complex beast.

2. Good security reputation.

   I don’t want an unintended path to root.

3. Excellent documentation.

   man pages FTW

4. It’s different.

   ‘*doas’ anyone?*

5. I am familiar with it!

Learning Objectives

1. Understand how to enumerate an LDAP directory, including determining the root DSE.
2. Understand how to use the PASS-THE-HASH technique with SAMBA on *nix
3. Understand how SSH Certificate Authorities work.
4. Reenforce system enumeration as a part of the pen-testing process.
5. Highlight some of the features of OpenBSD.
6. Get people to RTFM!

Building the challenge

- Start with a standard install of OpenBSD 6.3 (“dev box”)
- Build and test each technology separately
  - Does it work like I think it does?
  - Does documentation exist to give the participant a chance in succeeding?
  - Do I think learning objectives will be achieved?
- KISS: Install minimal number of packages. Dependencies are awesome!
- Take notes
- (Educated) guess about resource requirements

Samba has changed!

- Active Directory domain controller (amazing!)
- PAM support (wow!)
- PASS-THE-HASH: The old way
  - Fetch a patch (from foofus.net) and build your own.
  - Set `SMBHASH` environment variable containing user and hash before running `smbmount`
- PASS-THE-HASH: The new way
  - Use the `--pw-nt-hash` argument to `smbclient`

My (Modest) bucket list

- Visit Las Vegas (complete)
- Shoot a gun (complete, in Vegas!)
- In a support scenario, ask Ian Goldberg (noted Cryptographer and Computer Scientist) if he read the man page, and get “no” as a response. (complete)
- Throw wooden sticks at riot police (complete, in a training scenario)
- Give an algebra lesson to a Grade 7 Math class. (complete)
- **Set-up the YP service on *nix.** (complete)

Requirements for submitted boxes

1. I confirm that the machine does not contain any software requiring licensing.
2. I confirm that I have secured properly the root.txt file (chmod 600 or less).
3. I confirm that ping (icmp) is allowed on the machine's firewall.
4. I confirm that the machine is original, made by me and not published anywhere else.
5. I confirm that I will not publish the machine anywhere else until it is decommissioned from HTB.
6. I confirm that I will not publish solutions and write-ups for the machine until it is decommissioned from HTB.
7. I give full consent to publish the machine on HTB and mark me as "maker".
8. I confirm that the challenge does not contain malware or other software designed to harm other members or HTB itself.
9. I confirm that I added instructions to prepare the machine including where to change the IP address or what hosts files/vhosts/dns zones to alter.

2019 - More rules!

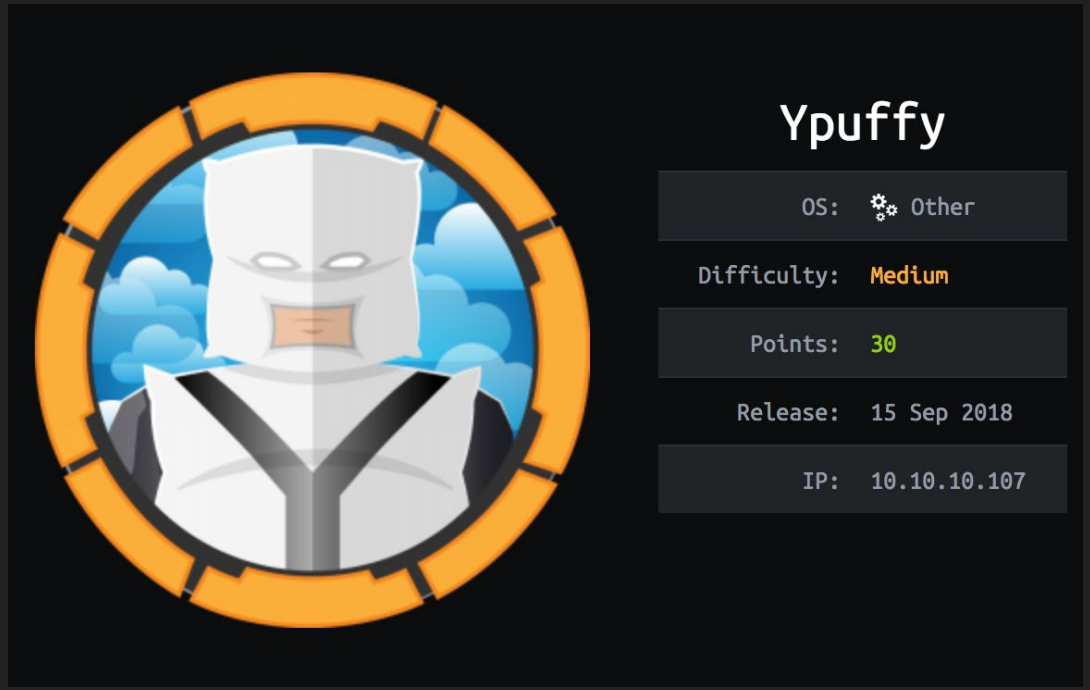
1. Multiple exploitation Vulnerability
2. Vulnerability cannot crash a service or the system
3. Realistic scenarios, please
4. No heavy bruteforcing/fuzzing/directory discovery
5. CTF ONLY within the HackTheBox VPN
6. Users Passwords cannot expire
7. Test your CTF before submitting it
8. Write a Writeup
9. Hint where is {user,root}.txt
10. Once it’s published, it’s published

The Process

- Development and testing - July 2018
- Submission - Early August 2018
- Release - Mid-September 2018
- Retirement - Mid-February 2019

The Box

- VMWare Fusion VM
  - 1 GB RAM
  - 4 GB disk
- Exposed services:
  - sshd (port 22)
  - smbd (port 139/445)
  - OpenBSD ldapd (port 389)
  - OpenBSD httpd (port 80)
- Other services
  - PostgreSQL server containing database of SSH public keys and principals
  - UWSGI/Python/Flask “web service” to PostgreSQL database



Walkthrough

Frustrating the recon step

```
server "ypuffy.hackthebox.htb" {
        listen on * port 80

    ...

        location "/sshauth*" {
                fastcgi socket "/run/wsgi/sshauthd.socket"
        }

        location * {
                block drop
        }
}
```

Getting the first flag

- LDAP enumeration reveals two users:
  - alice1978
    - has additional objectclass `sambaSamAccount` with attribute `sambaNTPassword`
  - bob8791
- Use `sambNTPassword` value with `smbclient` to access a share containing an SSH keypair located in a `ppk` file (PuTTY)
- Keypair can be used to get interactive shell for the “alice1978” account

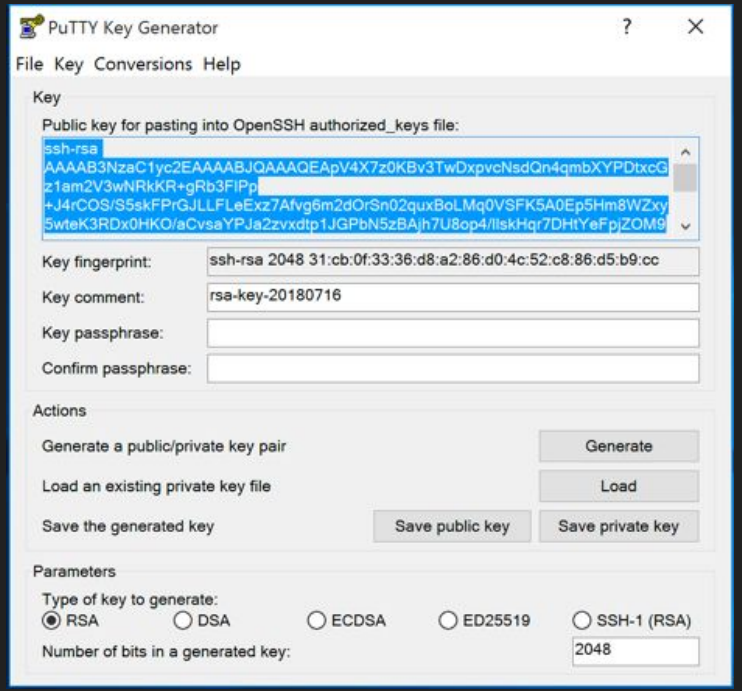
alice1978, passwd, hackthebox.htb
dn: uid=alice1978,ou=passwd,dc=hackthebox,dc=htb
uid: alice1978
cn: Alice
objectClass: account
objectClass: posixAccount
objectClass: top
objectClass: sambaSamAccount
userPassword:: e0JTREFVVEh9YWxpY2UxOTc4
uidNumber: 5000
gidNumber: 5000
gecos: Alice
homeDirectory: /home/alice1978
loginShell: /bin/ksh
sambaSID: S-1-5-21-3933741069-3307154301-3557023464-1001
displayName: Alice
sambaAcctFlags: [U]
sambaPasswordHistory: 0000000000000000000000000000000000000000000000000000000000000000
sambaNTPassword: 0B186E661BBDBDCF6047784DE8B9FD8B
sambaPwdLastSet: 1532916644

Base 64 encoding of:
{BSDAUTH}alice1978

NTLM Hash

Dealing with PPK files

My way



HTB way

```
% puttygen my_private_key.ppk -o alice -O private-openssh-new
```

sshd_config

```
PermitRootLogin prohibit-password

AuthorizedKeysCommand /usr/local/bin/curl
http://127.0.0.1/sshauth?type=keys&username=%u

AuthorizedKeysCommandUser nobody

TrustedUserCAKeys /home/userca/ca.pub

AuthorizedPrincipalsCommand /usr/local/bin/curl
http://127.0.0.1/sshauth?type=principals&username=%u

AuthorizedPrincipalsCommandUser nobody
```

Hints for second flag

There is no `authorized_keys` file in Alice's home directory. In fact, there is no .ssh directory at all in Alice's home directory.

There's a file named `sshauth.sql` in Bob's home directory containing the following:

```
CREATE TABLE principals (
uid text,
client cidr,
principal text,
PRIMARY KEY (uid,client,principal)
);

CREATE TABLE keys (
uid text,
key text,
PRIMARY KEY (uid,key)
);
grant select on principals,keys to appsrv;
```

Frustrating the escalation step (1/2)

```
@app.route('/sshauth', methods=['GET'])
def sshauth():
    return_data = ''
    params = []
    query_type = request.args.get('type')
    uid = request.args.get('username')

    if (not uid) or (not query_type):
        abort(400)

    if query_type == 'principals':
        query_str = 'SELECT principal from principals where client >>= %s and uid = %s;'
        params.append(request.remote_addr)
    elif query_type == 'keys':
        query_str = 'SELECT key from keys where uid = %s;'
    else:
        abort(400)

    if validate_uid(uid):
        params.append(uid)
        return_data = fetch_data(query_str, params)
    return return_data
```

Frustrating the escalation step (2/2)

```
    uid |    client    |    principal
----------+--------------+-------------------
 bob8791    | 10.0.0.0/8    | bob8791
 alice1978 | 10.0.0.0/8    | alice1978
 root     | 127.0.0.1/32 | 3m3rgencyB4ckd00r
 bob8791    | 127.0.0.1/32 | bob8791
 alice1978 | 127.0.0.1/32 | alice1978
```

Getting the second flag

1. Generate a SSH keypair.
2. Sign using the ‘userca’ key with the principal ‘`3m3rgencyB4ckd00r`’
3. SSH directly as root using the signed key

```
ypuffy$ whoami
alice1978
ypuffy$ ls -l /home/userca/
total 8
-r--------  1 userca  userca  1679 Jul 30 21:08 ca
-r--r--r--  1 userca  userca   410 Jul 30 21:08 ca.pub
ypuffy$ cat /etc/doas.conf
permit keepenv :wheel
permit nopass alice1978 as userca cmd /usr/bin/ssh-keygen
```

Remember what I said about minimal install?

```
#!/bin/sh

#
# raptor_xorgasm - xorg-x11-server LPE via OpenBSD's cron
# Copyright (c) 2018 Marco Ivaldi <raptor@0xdeadbeef.info>
#
# A flaw was found in xorg-x11-server before 1.20.3. An incorrect permission
# check for -modulepath and -logfile options when starting Xorg. X server
# allows unprivileged users with the ability to log in to the system via
# physical console to escalate their privileges and run arbitrary code under
# root privileges (CVE-2018-14665).
#
# This exploit targets OpenBSD's cron in order to escalate privileges to
# root on OpenBSD 6.3 and 6.4. You don't need to be connected to a physical
# console, it works perfectly on pseudo-terminals connected via SSH as well.
#
# See also:
# https://lists.x.org/archives/xorg-announce/2018-October/002927.html
# https://www.exploit-db.com/exploits/45697/
# https://gist.github.com/0x27/d8aae5de44ed385ff2a3d80196907850
#
# Usage:
# blobfish$ chmod +x raptor_xorgasm
# blobfish$ ./raptor_xorgasm
# [...]
# Be patient for a couple of minutes...
# [...]
# Don't forget to cleanup and run crontab -e to reload the crontab.
# -rw-r--r--  1 root  wheel  47327 Oct 27 14:48 /etc/crontab
# -rwsrwxrwx  1 root  wheel  7417 Oct 27 14:50 /usr/local/bin/pwned
# blobfish# id
# uid=0(root) gid=0(wheel) groups=1000(raptor), 0(wheel)
```

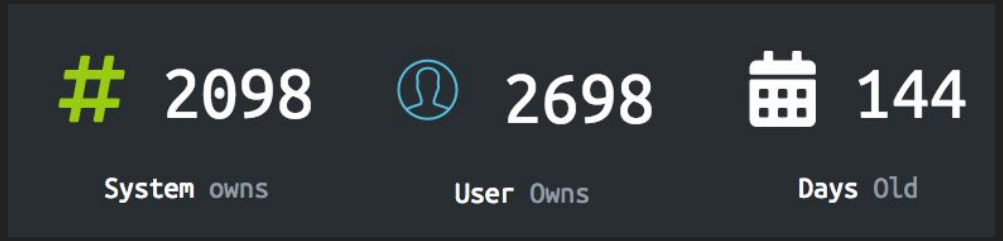
GAH!

CVE 2018-14665!

Feedback


1251
52


Enumeration
Real-Life
CVE
Custom
Exploitation
CTF-Like


2098
System owns
2698
User Owns
144
Days Old


Juha Remes
@jremes84
Follow
#Ypuffy root pwned at @hackthebox_eu. Fun box, kudos to @jtestart for a nice and educational hacking experience! 👍
#HackTheBox #CTF
GIF
10:31 AM - 19 Sep 2018

Some things I learned

- Samba has changed **lots** since version 3.0.22
- Unit testing is critical when building a successful challenge
- A complete run-through from beginning to end on a copy of the “to-be-submitted” box is important too!
- Check your spelling when setting-up YP
- OpenBSD’s reputation should not lead to complacency re: “minimal install”

Observations from write-ups

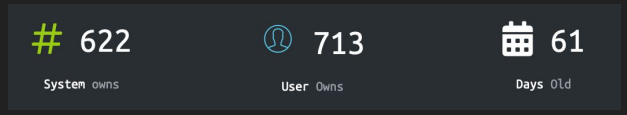
- Confirmed: Many did not know about advanced sshd features
- `/etc/passwd` is not the only account database
  - Difference between ‘account database’ and ‘authentication’
- Public key crypto a mysterious thing for many folks?
- Lots of great ways out there to present information on the web

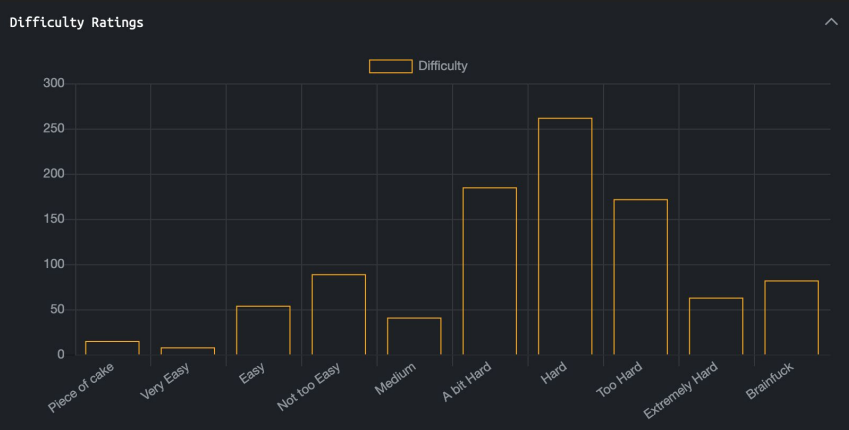
Write-ups

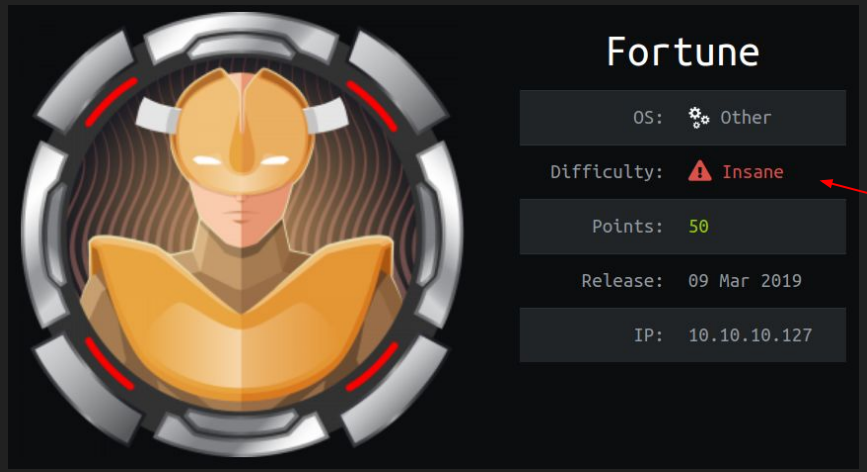
- https://hackso.me/ypuffy-htb-walkthrough/
- https://snowscan.io/htb-writeup-ypuffy/
- https://0xrick.github.io/hack-the-box/ypuffy/
- https://anubissec.github.io/Ypuffy-HackTheBox-WriteUp/
- https://epi052.gitlab.io/notes-to-self/blog/2018-09-15-hack-the-box-ypuffy/
- https://github.com/DoMINAToR98/HTB_Box_Writeups/blob/master/Ypuffy.md
- https://medium.com/@noobintheshell/htb-ypuffy-writeup-b7a666b460d5
- https://0x23b.github.io/posts/hackthebox/2019-02-09-htb_ypuffy_writeup/
- https://0xdf.gitlab.io/2019/02/09/htb-ypuffy.html
- Français: http://devloop.users.sourceforge.net/index.php?article183/solution-du-ctf-ypuffy-de-hackthebox

Next challenge

- More public key crypto
- More OpenBSD features
- More PostgreSQL and friends







Oops!

Thank You | Merci