

**NECMETTİN ERBAKAN ÜNİVERSİTESİ**

**BİLGİ GÜVENLİĞİ DERSİ DÖNEM PROJE SONUÇ RAPORU**

Fırat Kaya

Environment Variable and Set-UID Lab

Öğretici:

Dr. Öğr. Üyesi Alperen Eroğlu

Bilgisayar Mühendisliği Bölümü

KONYA, [2023]

İçindekiler

[ÖZET 3](#_Toc154324769)

[ABSTRACT 3](#_Toc154324770)

[GİRİŞ 3](#_Toc154324771)

[LİTERATÜR TARAMASI 4](#_Toc154324772)

[Görev 1: Çevre Değişkenlerini Manipüle Etme 4](#_Toc154324773)

[Görev 2: Ana Süreçten Çocuk Sürece Çevre Değişkenleri Aktarma 4](#_Toc154324774)

[Görev 3: execve() Fonksiyonunun Çalışma Mantığı 6](#_Toc154324775)

[Görev 4: system() ve execve() Karşılaştırması 7](#_Toc154324776)

[Görev 5: Set-UID Programlarında Çevre Değişkenleri 8](#_Toc154324777)

[Görev 6: PATH Çevre Değişkeni ve Set-UID Programları 9](#_Toc154324778)

[Görev 7: LD PRELOAD Çevre Değişkeni ve Set-UID Programları 9](#_Toc154324779)

[Görev 8: system() ve execve() Karşılaştırması 10](#_Toc154324780)

[Görev 9: Capability Leaking 11](#_Toc154324781)

[BULGULAR 12](#_Toc154324782)

[Çevre Değişkenleri ve Set-UID Programları: 12](#_Toc154324783)

[Yetenek Sızıntısı: 12](#_Toc154324784)

[Execve() ve System() Karşılaştırması: 13](#_Toc154324785)

[PATH Çevre Değişkeni ve Set-UID Programları: 13](#_Toc154324786)

[LD PRELOAD Çevre Değişkeni ve Set-UID Programları: 13](#_Toc154324787)

[Set-UID Programlarında Güvenlik İlkeleri: 13](#_Toc154324788)

[SONUÇ 13](#_Toc154324789)

[KAYNAKÇA 13](#_Toc154324790)

# ÖZET

Bu çalışma, çevre değişkenleri ve Set-UID programları arasındaki karmaşık ilişkiyi inceleyerek önemli güvenlik konularını ortaya koymaktadır. Çalışma, çeşitli çevre değişkenlerinin etkilerini inceleme, Set-UID programları ile deneyler yapma ve system() ve execve() gibi sistem çağrılarının davranışlarını analiz etme gibi kapsamlı bir yöntem kullanmaktadır. Bulgular, özellikle Set-UID programları bağlamında çevre değişkenlerinin manipülasyonunun ayrıcalık yükselmesine ve izinsiz erişime nasıl neden olabileceğini vurgulamaktadır. Çalışmanın özgünlüğü, çevre değişkenlerinin Set-UID programları içinde manipüle edilmesinin güvenlik üzerindeki etkilerini derinlemesine analiz etmesinde yatmaktadır. Bu çalışma, Linux sistem güvenliğini artırmak için değerli içgörüler sunmayı amaçlamaktadır.

# ABSTRACT

The main objective of this study is to explore and analyze the impact of manipulating environment variables on the security of Linux systems. The study employs a comprehensive methodology that involves examining various aspects of environment variables, conducting experiments with Set-UID programs, and analyzing the behavior of system calls like system() and execve().The findings of this research highlight the potential security risks associated with environment variables, especially in the context of Set-UID programs. It reveals how the manipulation of these variables can lead to privilege escalation and unauthorized access. The originality of this study lies in its in-depth analysis of the security implications of manipulating environment variables within the context of Set-UID programs, providing valuable insights for enhancing Linux system security.

# GİRİŞ

Environment Variable and Set-UID Lab, çevre değişkenlerinin ve çeşitli sistem çağrılarının Unix işletim sistemlerindeki davranışlarını anlamaya yöneliktir. Çalışmanın temel odak noktası, çevre değişkenlerinin işletim sistemi süreçleri üzerindeki etkilerini ve Set-UID programlarıyla ilişkisini incelemektir.Çevre değişkenleri üzerinden olası güvenlik riskleri de ele alınacaktır.

İlk olarak, çevre değişkenleri ve temel komut satırı işlemleri üzerinde genel bilgiler edinilecektir. Ardından, çevre değişkenlerinin ayarlanması ve kaldırılması ile ilgili temel işlemler incelenecektir. Fork() fonksiyonu kullanılarak ana süreçten alt sürece çevre değişkenleri aktarımı ele alınacak ve execve() fonksiyonunun çevre değişkenleri üzerindeki etkisi gözlemlenecektir.

Çalışmanın devamında, system() fonksiyonu ve çevre değişkenleri arasındaki ilişki, Set-UID programlarının çevre değişkenleriyle etkileşimi ve bu programların güvenlik açısından önemi detaylı bir şekilde ele alınacaktır.

# LİTERATÜR TARAMASI

“The Seven Deadly Sins of Linux Security[1]” - Bu makale, Linuxta bulunan 7 büyük sorunu ele alıyor ve nasıl çözüleceği hakkında öneriler sunuyor ve yapılmaması gerekenlerden bahsediyor. Bu sorunlardan biri de Set-UID programlarının genel güvenlik ilkelerini incelemekte ve ayrıcalıkların nasıl yönetildiği konusunda bilgi sağlamakta ve bu programların açabileceği sorunları güvenlik zafiyetlerinden bahsediyor.

“Transparent Serverless Process Offloading[2]” – Bu makale,Linuxta global değişkenlerin kullanımı ve kullanılan fonksiyonların ne işe yaradığını ve nasıl kullanılması gerektiğinden bahsediyor ancak bunu güvenlik amacı yanı sıra sunucu işlemleri , api işlemleri gibi yerlerde nasıl kullanılacağından bahsediyor burada bizim için önemli olan kısım bunların güvenli şekilde kullanımının gösterilmesidir.

“Buffer Overflow Attack and Defense Techniques[3]” – Bu makale , Linuxta Buffer Overflow Saldırılarına karşı savunmadan bahsediyor ancak kullanılan yöntemlerde ve bahsedilen içerikte Set-UID ve global değişkenlerin önemi göz önünde bulunduruluyor.

1. **YÖNTEM**

## Görev 1: Çevre Değişkenlerini Manipüle Etme

Çevre Değişkenlerini İnceleme:

* printenv ve env komutları kullanılarak sistemdeki tüm çevre değişkenleri görüntülendi.
* Örnek olarak, printenv PWD veya env | grep PWD komutları ile belirli bir çevre değişkeni, örneğin PWD, görüntülendi.

Çevre Değişkenlerini Ayarlama ve Silme:

* export komutu ile çevre değişkenleri ayarlandı.
* unset komutu ile belirli çevre değişkenleri silindi.

## Görev 2: Ana Süreçten Çocuk Sürece Çevre Değişkenleri Aktarma

Çocuk Süreçte Çevre Değişkenlerinin Görüntülenmesi:

* myprintenv.c programı kullanılarak, fork() ve execve() fonksiyonları kullanılarak çocuk süreçte çevre değişkenleri görüntülendi.
* İlk olarak, çocuk süreç çevre değişkenlerini görüntüleyerek başladı.

Çocuk Süreçte ve Ana Süreçte Farklı Çevre Değişkenleri:

* Çocuk süreçte çevre değişkenleri ile ana süreçteki çevre değişkenleri karşılaştırıldı.
* diff komutu kullanılarak farklılıklar belirlendi.

Çocuk Süreçte Çevre Değişkenlerinin Güncellenmesi:

* Çocuk süreçte, çevre değişkenleri güncellenerek farklı bir çevre oluşturuldu.
* Güncellenmiş çevre değişkenleri görüntülendi.
* #include <unistd.h>
* #include <stdio.h>
* #include <stdlib.h>
* extern char \*\*environ;
* void printenv()
* {
* int i = 0;
* while (environ[i] != NULL) {
* printf("%s\n", environ[i]);
* i++;
* }
* }
* void main()
* {
* pid\_t childPid;
* switch(childPid = fork()) {
* case 0:  /\* child process \*/
* printenv();
* exit(0);
* default:  /\* parent process \*/
* // printenv();
* exit(0);
* }
* }



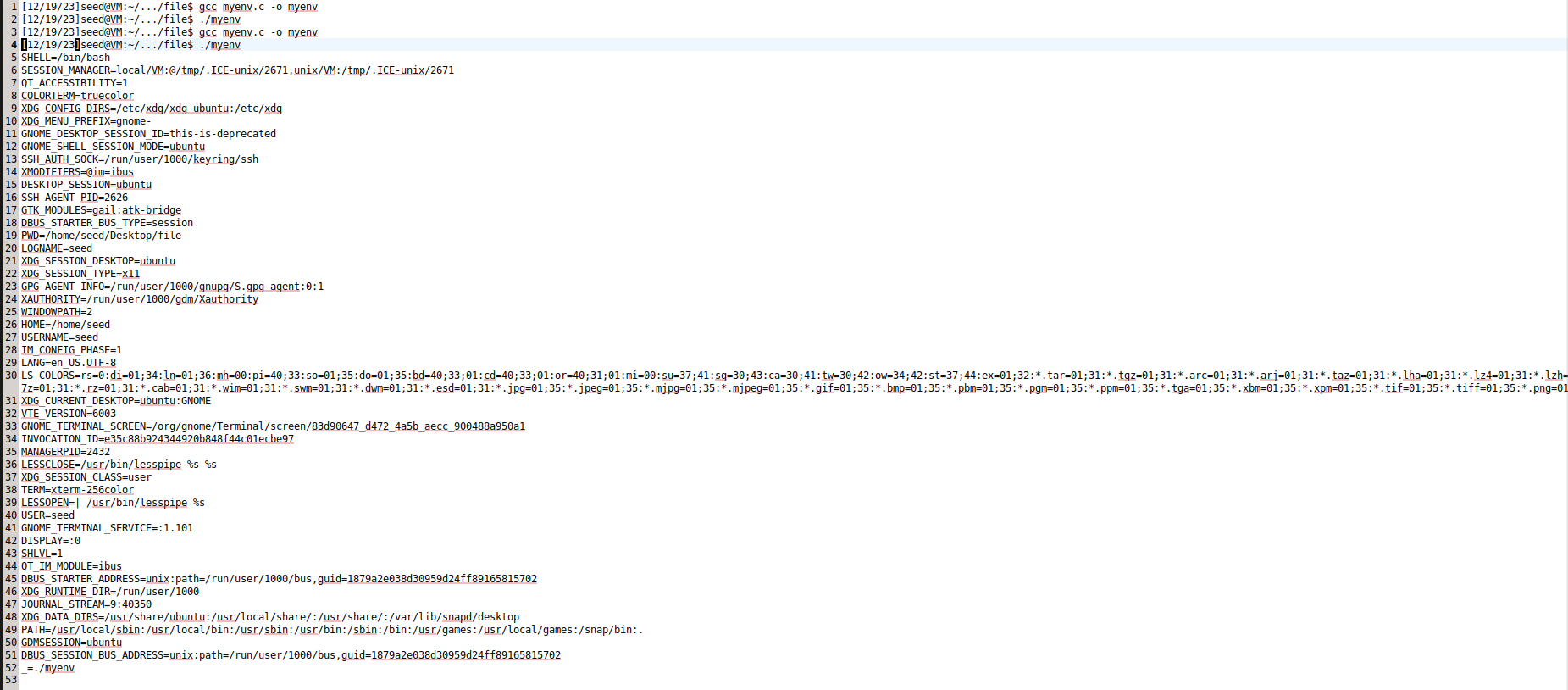
## Görev 3: execve() Fonksiyonunun Çalışma Mantığı

execve() Fonksiyonunun Kullanımı:

* myenv.c programı kullanılarak, execve() fonksiyonunun yeni bir programı nasıl yürüttüğü incelendi.
* execve("/usr/bin/env", argv, environ) kullanılarak /usr/bin/env programı çalıştırıldı.

execve() Fonksiyonu ile LD\_PRELOAD Kullanımı:

* execve("/usr/bin/env", argv, environ) kullanılarak /usr/bin/env programının LD\_PRELOAD ile nasıl etkilenebileceği gözlendi.
* Yeni programın çevre değişkenleri kontrol edildi.
* #include <unistd.h>
* extern char \*\*environ;
* int main()
* {
* char \*argv[2];
* argv[0] = "/usr/bin/env";
* argv[1] = NULL;
* execve("/usr/bin/env", argv, NULL);
* return 0 ;
* }



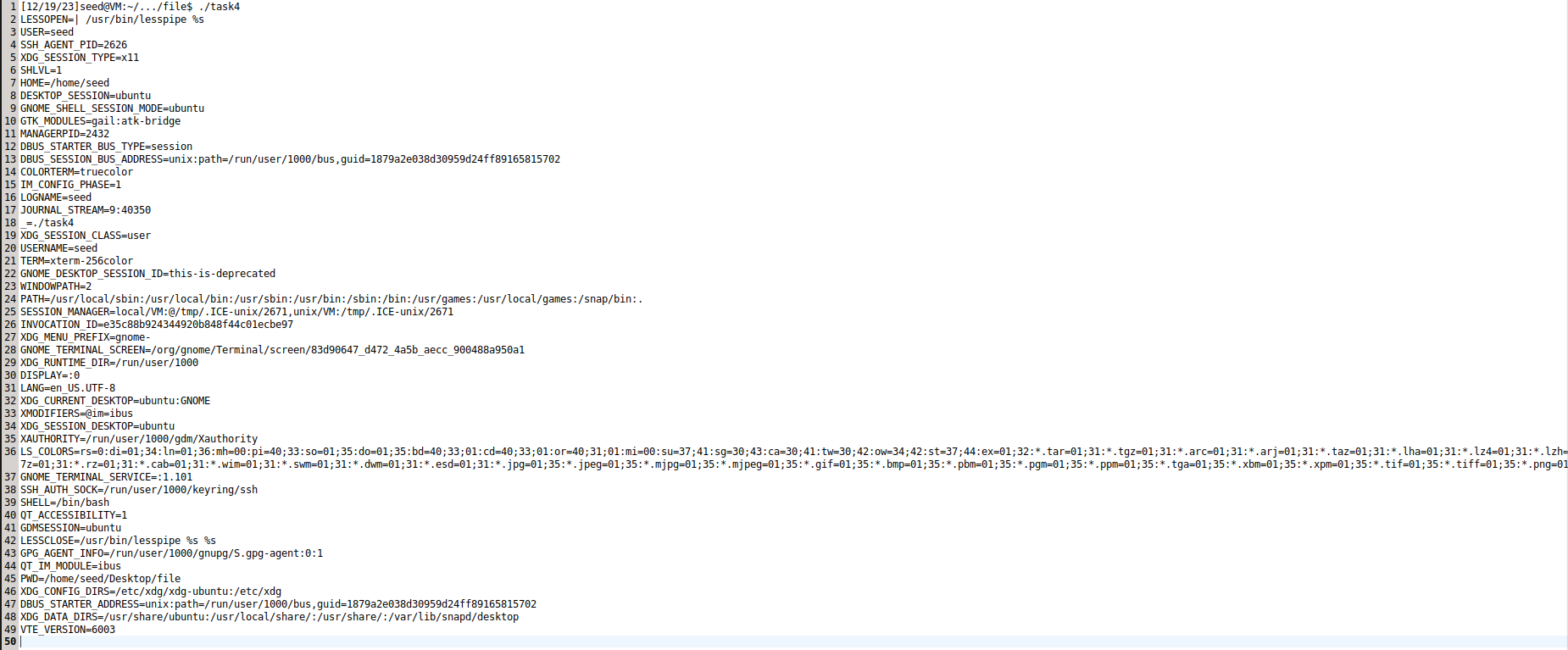
## Görev 4: system() ve execve() Karşılaştırması

system() Fonksiyonu Kullanımı:

* system() fonksiyonunun çevre değişkenlerini nasıl etkilediği incelendi.
* /usr/bin/env programı system("/usr/bin/env") kullanılarak çalıştırıldı.

execve() ve system() Karşılaştırması:

* catall.c adlı Set-UID programı oluşturuldu ve system() ve execve() ile /bin/cat komutu çalıştırıldı.
* Her iki senaryonun sonuçları karşılaştırıldı.
* #include <stdio.h>
* #include <stdlib.h>
* int main()
* {
* system("/usr/bin/env");
* return 0 ;
* }



## Görev 5: Set-UID Programlarında Çevre Değişkenleri

Set-UID Programının Oluşturulması:

* Çevre değişkenlerinin Set-UID programları üzerindeki etkisini anlamak için bir Set-UID programı oluşturuldu.
* Program, çevre değişkenlerini görüntüleyen basit bir C programıdır.

Set-UID Programının Çalıştırılması:

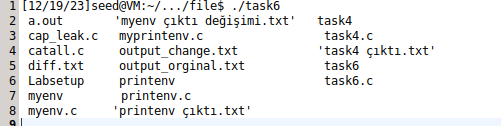
* Set-UID programı, çeşitli çevre değişkenleri ayarlandıktan sonra çalıştırıldı.
* Ayarlanan çevre değişkenlerinin program içindeki etkisi gözlendi.
* #include <stdio.h>
* #include <stdlib.h>
* extern char \*\*environ;
* int main()
* {
* int i = 0;
* while (environ[i] != NULL) {
* printf("%s\n", environ[i]);
* i++;
* }
* }



## Görev 6: PATH Çevre Değişkeni ve Set-UID Programları

PATH Değişkeninin Kullanımı:

* PATH çevre değişkeninin Set-UID programlarında nasıl bir güvenlik riski oluşturabileceği incelendi.
* Basit bir Set-UID programı oluşturularak PATH değişkeni üzerinde denemeler yapıldı.
* $ export PATH=/home/seed:$PATH
* int main()
* {
* system("ls");
* return 0;
* }



## Görev 7: LD PRELOAD Çevre Değişkeni ve Set-UID Programları

Dinamik Yükleme ve LD PRELOAD:

* LD\_PRELOAD çevre değişkeninin dinamik yüklemeyi nasıl etkilediğini görmek için bir C programı oluşturuldu.
* Program, sleep() fonksiyonunu değiştirerek özelleştirilmiş bir davranış sergiledi.

LD PRELOAD'un Set-UID Programlarına Etkisi:

* Oluşturulan Set-UID programı, LD\_PRELOAD çevre değişkeni ile çalıştırıldı.
* LD\_PRELOAD'un Set-UID programının davranışını nasıl değiştirdiği gözlendi.

#include <stdio.h>

void sleep (int s)

{

/\* If this is invoked by a privileged program,

you can do damages here! \*/

printf("I am not sleeping!\n");

}

--------------------------------

/\* myprog.c \*/

#include <unistd.h>

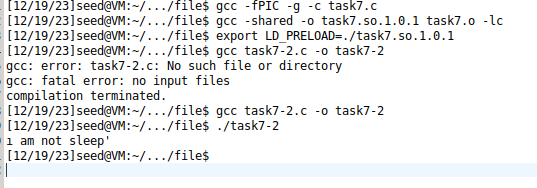
int main()

{

sleep(1);

return 0;

}



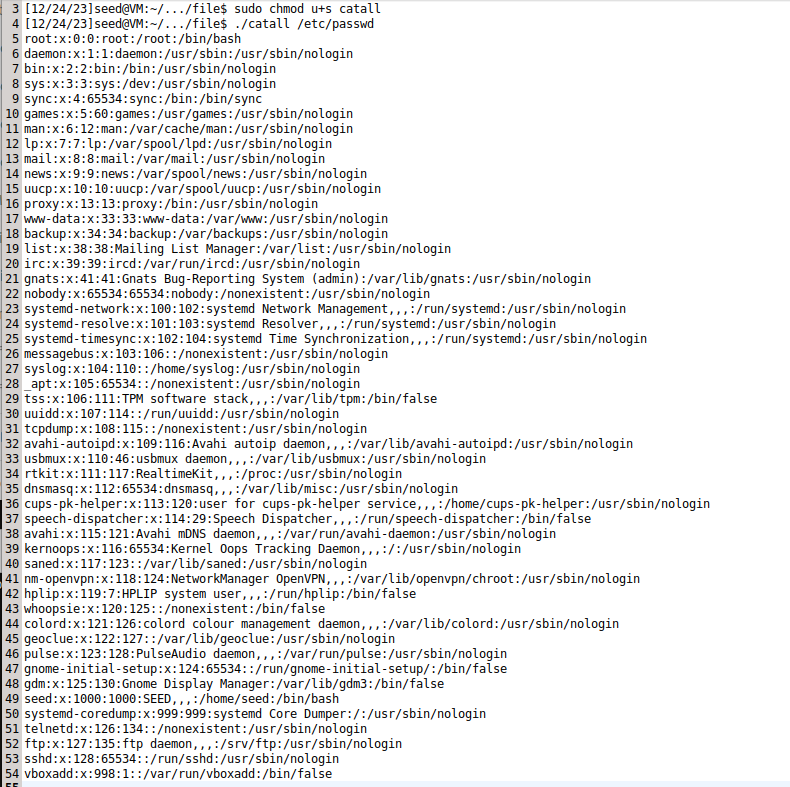
## Görev 8: system() ve execve() Karşılaştırması

system() Fonksiyonu Kullanımı:

* system() fonksiyonunun çevre değişkenlerini nasıl etkilediğini görmek için bir C programı oluşturuldu.
* Program, /bin/cat komutunu kullanarak dosya içeriğini ekrana yazdı.

execve() Fonksiyonu Kullanımı:

* Oluşturulan Set-UID programı, hem system() hem de execve() fonksiyonlarını kullanarak /bin/cat komutunu çağırdı.
* Her iki yöntemin davranışı karşılaştırıldı.



## Görev 9: Capability Leaking

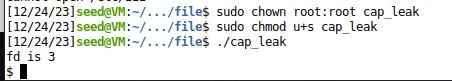
Yetenek Sızıntısının İncelenmesi:

* Bir C programı oluşturularak, Set-UID programlarında yetenek sızıntısının nasıl gerçekleşebileceği gösterildi.
* Program, /etc/zzz dosyasına yazma yeteneğine sahip olmasına rağmen kötü amaçlı kullanım potansiyeli taşıdı.

Yetenek Sızıntısının Exploit Edilmesi:

* Oluşturulan Set-UID programı, normal bir kullanıcı olarak çalıştırıldığında /etc/zzz dosyasına yazma yeteneği sızıntı yaratmaktadır.
* Bu sızıntı, Set-UID programının kullanımı sırasında kötü amaçlı bir eylemi gerçekleştirmek için potansiyel bir saldırı vektörü oluşturabilir.
* void main()
* {
* int fd;
* char \*v[2];
* /\* Assume that /etc/zzz is an important system file,
* \* and it is owned by root with permission 0644.
* \* Before running this program, you should create
* \* the file /etc/zzz first. \*/
* fd = open("/etc/zzz", O\_RDWR | O\_APPEND);
* if (fd == -1) {
* printf("Cannot open /etc/zzz\n");
* exit(0);
* }
* // Print out the file descriptor value
* printf("fd is %d\n", fd);
* // Permanently disable the privilege by making the
* // effective uid the same as the real uid
* setuid(getuid());
* // Execute /bin/sh
* v[0] = "/bin/sh"; v[1] = 0;
* execve(v[0], v, 0);
* }





# BULGULAR

## Çevre Değişkenleri ve Set-UID Programları:

* Çevre değişkenleri, Set-UID programlarının güvenliği üzerinde etkilidir.
* Özellikle PATH, LD\_PRELOAD gibi değişkenler, güvenlik riski oluşturabilir.
* Çevre değişkenlerinin programın güvenliğini etkilediğini gösteren deneyler yapıldı.

## Yetenek Sızıntısı:

* Set-UID programları, ayrıcalıklarını düşürdükten sonra bile yetenek sızıntısı konusunda dikkatli olmalıdır.
* Temizlenmeyen yetenekler, düşürülmüş ayrıcalıklara rağmen programın özel yeteneklere erişim sağlamasına neden olabilir.

## Execve() ve System() Karşılaştırması:

* Set-UID programlarında harici programları çağırmak için execve() ve system() kullanımı karşılaştırıldı.
* System() kullanımının çevre değişkenleri ve shell etkileşimi açısından riskli olduğu gözlendi.
* Execve() kullanımının daha güvenli olduğu ve programın doğrudan çağrıldığı tespit edildi.

## PATH Çevre Değişkeni ve Set-UID Programları:

* PATH çevre değişkeninin Set-UID programlarında güvenlik riski oluşturabileceği gösterildi.
* Kötü amaçlı kullanıcıların PATH değişkenini manipüle ederek istenmeyen kodları çalıştırabileceği belirlendi.

## LD PRELOAD Çevre Değişkeni ve Set-UID Programları:

* LD PRELOAD çevre değişkeninin Set-UID programlarının davranışını değiştirmek için kullanılabileceği incelendi.
* Bu değişkenin dikkatlice kontrol edilmesi gerektiği ve kötü amaçlı kullanıcıların programın akışını değiştirebileceği gözlemlendi.

## Set-UID Programlarında Güvenlik İlkeleri:

* Set-UID programlarının güvenliği için çevre değişkenlerinin temizlenmesi ve güvenlik risklerinin değerlendirilmesi gerektiği vurgulandı.
* Temel güvenlik ilkelerinin benimsenmesi, güvenli bir Set-UID programı geliştirmek için önemli olarak belirlendi.

# SONUÇ

Bu laboratuvar çalışması, çevre değişkenleri ve Set-UID programları arasındaki ilişkiyi detaylı bir şekilde inceleyerek önemli güvenlik konularını ortaya koymuştur. Elde edilen bulgular ve yapılan deneyimlere bakarak çevre değişkenleri ve Set-UID programlarının güvenli tasarımı konularında bilinç oluşturmayı amaçlamaktadır. Çevre değişkenlerinin etkili bir şekilde kontrol edilmesi ve güvenlik prensiplerine uyulması, sistemlerin güvenliğini artırabilir ve potansiyel güvenlik açıklarını en aza indirebilir.

# KAYNAKÇA

[1] <https://dl.acm.org/doi/abs/10.1145/1255421.1255423>

[2] <https://www.research-collection.ethz.ch/handle/20.500.11850/619289>

[3] <https://koreascience.kr/article/JAKO202108038346114.pdf>

<https://books.google.com.tr/books?hl=tr&lr=&id=KsNiDwAAQBAJ&oi=fnd&pg=PP1&dq=Environment+Variable+and+SetUID+linux&ots=M_R-uVf-8y&sig=9fooRGiQ9RZ1qiwbmMtpqyMDEYw&redir_esc=y#v=onepage&q&f=false>

<https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=11cf9adc6a18b83da03d87821c95daef360d94b3>