CEN429 GÃ¼venli Programlama Hafta-6

Java iÃ§in RASP Teknikleri

Yazar: Dr. Ã–Ğr. Ãœyesi UÄŸur CORUH

İçindekiler

1 **CEN429 GÃ¼venli Programlama** 1
1.1 Hafta-6 . . . 1
1.1.1 Outline . . . 1
1.2 **Hafta-6: RASP (Runtime Application Self-Protection) Java TarafÄ±** . . . 1

Şekil Listesi

Tablo Listesi

1 CEN429 GÃ¼venli Programlama

1.1 Hafta-6

1.1.0.1 Java iÃ§in RASP Teknikleri Ä°ndir

- PDF[1]
- DOC[2]
- SLIDE[3]
- PPTX[4]

1.1.1 Outline

- RASP (Ã‡alÄ±ÅŸma ZamanÄ± Uygulama KorumasÄ±) Nedir?
- Java Ä°Ã§in RASP Teknikleri
- EmÃ¼latÃ¶r, Root ve Debug Modu Tespiti
- GÃ¼venlik KÃ¼tÃ¼phaneleri ve SSL Pinning

1.2 Hafta-6: RASP (Runtime Application Self-Protection) Java TarafÄ±

Java uygulamalarÄ±nda RASP (Runtime Application Self-Protection), uygulamalarÄ±n Ã§alÄ±ÅŸma zamanÄ±nda gÃ¼venliklerini saÄŸlamak iÃ§in kullanÄ±lan tekniklerden oluÅŸur. Bu hafta, Java tabanlÄ± uygulamalar iÃ§in RASP stratejilerini inceleyeceÄŸiz. Uygulamalar, Ã¶zellikle mobil uygulamalar, Ã§alÄ±ÅŸma zamanÄ±nda Ã§eÅŸitli tehditlere karÅŸÄ± kendilerini koruyabilmelidir. AÅŸaÄŸÄ±daki baÅŸlÄ±klar, Java tarafÄ±nda RASP iÃ§in kullanÄ±lan teknikleri kapsamaktadÄ±r.

1.2.0.1 1. EmÃ¼latÃ¶r Tespiti (Emulator Detection) Teorik AÃ§Ä±klama:

EmÃ¼latÃ¶rler, saldÄ±rganlarÄ±n uygulamayÄ± analiz etmeleri ve zayÄ±f noktalarÄ± keÅŸfetmeleri iÃ§in kullanabilecekleri araÃ§lardÄ±r. EmÃ¼latÃ¶r tespiti, uygulamanÄ±n bir emÃ¼latÃ¶r ortamÄ±nda Ã§alÄ±ÅŸÄ±p

[1] pandoc_cen429-week-6.pdf
[2] pandoc_cen429-week-6.docx
[3] cen429-week-6.pdf
[4] cen429-week-6.pptx

Ã§alÄ±ÅŸmadÄ±ÄŸÄ±nÄ± anlamasÄ±na olanak tanÄ±r. Qemu gibi popÃ¼ler emÃ¼latÃ¶rler iÃ§in Ã¶zel tespit mekanizmalarÄ± uygulanabilir.

Kaynak ve Uygulama:

- Qemu ARM EmÃ¼latÃ¶r Tespiti iÃ§in kullanÄ±lan bir Ã¶rnek: Anti Emulator for Qemu ARM[5]
- EmÃ¼latÃ¶r ortamÄ±nÄ± algÄ±lama ve Ã§alÄ±ÅŸma sÃ¼recinde uygulamanÄ±n iÅŸlevini deÄŸiÅŸtirme.

1.2.0.2 2. Hata AyÄ±klama Modu Tespiti (Debug Mode Detection) Teorik AÃ§Ä±klama:

Bir uygulamanÄ±n hata ayÄ±klama (debug) modunda Ã§alÄ±ÅŸmasÄ±, kÃ¶tÃ¼ niyetli kiÅŸilerin uygulamayÄ± analiz etmeleri iÃ§in bir fÄ±rsat saÄŸlar. UygulamanÄ±n hata ayÄ±klama modunda olup olmadÄ±ÄŸÄ±nÄ± tespit etmek, bu modda Ã§alÄ±ÅŸmasÄ±nÄ± engelleyerek gÃ¼venliÄŸi artÄ±rÄ±r.

Uygulama Ã–rnekleri:

1. UygulamanÄ±n Ã§alÄ±ÅŸma zamanÄ±nda hata ayÄ±klama modunda olup olmadÄ±ÄŸÄ±nÄ± kontrol eden kod parÃ§acÄ±klarÄ± eklemek.
2. Hata ayÄ±klama modunda olduÄŸunda uygulamanÄ±n Ã§alÄ±ÅŸmasÄ±nÄ± sonlandÄ±rmak veya farklÄ± bir iÅŸlev sergilemesini saÄŸlamak.

1.2.0.3 3. Debugger BaÄŸlantÄ±sÄ± Tespiti (Debugger Attach Detection) Teorik AÃ§Ä±klama:

Hata ayÄ±klayÄ±cÄ±larÄ±n (debugger) uygulamaya baÄŸlanmasÄ±, uygulamanÄ±n izlenmesine ve analiz edilmesine yol aÃ§ar. Debugger tespiti, uygulamanÄ±n Ã§alÄ±ÅŸma sÄ±rasÄ±nda bir hata ayÄ±klayÄ±cÄ±ya baÄŸlanÄ±p baÄŸlanmadÄ±ÄŸÄ±nÄ± kontrol eder ve buna gÃ¶re hareket eder.

Uygulama Ã–rnekleri:

1. Debugger tespit edildiÄŸinde uygulamanÄ±n kapanmasÄ±nÄ± veya iÅŸlev deÄŸiÅŸtirmesini saÄŸlama.
2. Hata ayÄ±klayÄ±cÄ±ya baÄŸlantÄ±yÄ± algÄ±layan gÃ¼venlik mekanizmalarÄ± eklemek.

1.2.0.4 4. RootBeer Implementasyonu (RootBeer Implementation) Teorik AÃ§Ä±klama:

RootBeer, Android cihazlarÄ±nÄ±n root olup olmadÄ±ÄŸÄ±nÄ± kontrol eden bir kÃ¼tÃ¼phanedir. Root edilmiÅŸ cihazlar, uygulamanÄ±n gÃ¼venliÄŸini tehlikeye atabilir. RootBeer kullanarak, root edilmiÅŸ cihazlarÄ±n tespiti yapÄ±labilir.

Uygulama Ã–rnekleri:

1. RootBeer kullanarak cihazÄ±n root olup olmadÄ±ÄŸÄ±nÄ± tespit etme.
2. Root edilmiÅŸ cihazlarda uygulamanÄ±n Ã§alÄ±ÅŸmasÄ±nÄ± engelleme veya kÄ±sÄ±tlÄ± iÅŸlev saÄŸlama.

1.2.0.5 5. AndroidSecurityManager ile Root Tespiti (AndroidSecurityManager Rooted Device Check) Teorik AÃ§Ä±klama:

AndroidSecurityManager, Android cihazlarÄ±nÄ±n gÃ¼venlik durumu hakkÄ±nda bilgi saÄŸlayan bir gÃ¼venlik yÃ¶neticisidir. Root edilmiÅŸ cihazlarÄ± tespit ederek uygulamanÄ±n bu cihazlarda Ã§alÄ±ÅŸmamasÄ±nÄ± saÄŸlar.

Uygulama Ã–rnekleri:

1. AndroidSecurityManager kullanarak root kontrolÃ¼ gerÃ§ekleÅŸtirme.
2. Root edilmiÅŸ cihazlarda belirli Ã¶zellikleri devre dÄ±ÅŸÄ± bÄ±rakma.

1.2.0.6 6. SafetyNet Implementasyonu (SafetyNet Implementation) Teorik AÃ§Ä±klama:

Google SafetyNet, cihazÄ±n gÃ¼venlik durumunu deÄŸerlendirmek iÃ§in kullanÄ±lan bir API'dir. Uygulamalar, SafetyNet ile cihazÄ±n gÃ¼venlik bÃ¼tÃ¼nlÃ¼ÄŸÃ¼nÃ¼ kontrol edebilir ve gÃ¼venlik ihlalleri tespit edildiÄŸinde tepki verebilir.

Uygulama Ã–rnekleri:

[5] https://github.com/strazzere/anti-emulator/blob/master/AntiEmulator/jni/anti.c

1. SafetyNet API'yi kullanarak cihazÄ±n gÃ¼venlik bÃ¼tÃ¼nlÃ¼ÄŸÃ¼nÃ¼ kontrol etmek.
2. GÃ¼venlik ihlalleri tespit edildiÄŸinde uygulamanÄ±n davranÄ±ÅŸÄ±nÄ± deÄŸiÅŸtirmek veya sonlandÄ±rmak.

1.2.0.7 7. KullanÄ±lan Native KÃ¼tÃ¼phane Checksum KontrolÃ¼ (Used Native Library Checksum Control) Teorik AÃ§Ä±klama: UygulamanÄ±n kullandÄ±ÄŸÄ± native kÃ¼tÃ¼phanelerin checksum deÄŸerlerini kontrol etmek, bu kÃ¼tÃ¼phanelerin deÄŸiÅŸtirilip deÄŸiÅŸtirilmediÄŸini anlamamÄ±zÄ± saÄŸlar. Bu, uygulamanÄ±n gÃ¼venliÄŸini korumanÄ±n Ã¶nemli bir yoludur.

Uygulama Ã–rnekleri:

1. Ã‡alÄ±ÅŸma zamanÄ±nda kullanÄ±lan kÃ¼tÃ¼phanelerin checksum deÄŸerlerini kontrol etme.
2. KÃ¼tÃ¼phane Ã¼zerinde bir deÄŸiÅŸiklik tespit edilirse uygulamanÄ±n Ã§alÄ±ÅŸmasÄ±nÄ± sonlandÄ±rma veya iÅŸlev deÄŸiÅŸtirme.

1.2.0.8 8. Tamper Cihaz Tespiti (Tamper Device Detection) Teorik AÃ§Ä±klama: CihazÄ±n veya uygulamanÄ±n manipÃ¼le edilip edilmediÄŸini kontrol etmek, uygulamayÄ± gÃ¼venlik ihlallerine karÅŸÄ± korur. Tamper tespiti ile cihaz veya uygulama Ã¼zerinde yapÄ±lmÄ±ÅŸ herhangi bir deÄŸiÅŸikliÄŸi algÄ±layabilirsiniz.

Uygulama Ã–rnekleri:

1. Cihaz veya uygulamanÄ±n tamper edilmiÅŸ olup olmadÄ±ÄŸÄ±nÄ± tespit etme.
2. Tamper tespit edildiÄŸinde uygulamanÄ±n Ã§alÄ±ÅŸmasÄ±nÄ± durdurma veya kÄ±sÄ±tlama.

1.2.0.9 9. SSL Pinning ve WebView SSL Pinning (SSL Pinning and Webview SSL Pinning) Teorik AÃ§Ä±klama: SSL Pinning, uygulamanÄ±n belirli bir sunucuya gÃ¼venli ÅŸekilde baÄŸlanmasÄ±nÄ± saÄŸlamak iÃ§in kullanÄ±lÄ±r. WebView Ã¼zerinde SSL pinning uygulamak, kullanÄ±cÄ±larÄ±n sahte sunucularla baÄŸlantÄ± kurmasÄ±nÄ± engeller.

Uygulama Ã–rnekleri:

1. WebView'da SSL pinning uygulayarak sunucunun kimliÄŸini doÄŸrulamak.
2. YanlÄ±ÅŸ sunucularla baÄŸlantÄ± kurulduÄŸunda baÄŸlantÄ±yÄ± kesmek.

1.2.0.10 10. Sunucu SertifikasÄ± KontrolÃ¼ (Server Certificate Check) Teorik AÃ§Ä±klama: UygulamanÄ±n bir sunucuya baÄŸlanÄ±rken sunucu sertifikasÄ±nÄ±n doÄŸruluÄŸunu kontrol etmesi, sahte sunucularla baÄŸlantÄ± kurmayÄ± engeller. Bu, man-in-the-middle saldÄ±rÄ±larÄ±na karÅŸÄ± Ã¶nemli bir koruma saÄŸlar.

Uygulama Ã–rnekleri:

1. Sunucu sertifikasÄ±nÄ±n doÄŸruluÄŸunu Ã§alÄ±ÅŸma sÄ±rasÄ±nda kontrol etme.
2. YanlÄ±ÅŸ sertifika tespit edildiÄŸinde baÄŸlantÄ±yÄ± kesme.

1.2.0.11 11. Cihaz ve SÃ¼rÃ¼m BaÄŸlama (DeviceBinding & VersionBinding) Teorik AÃ§Ä±klama: Cihaz baÄŸlama, uygulamanÄ±n belirli bir cihaz Ã¼zerinde Ã§alÄ±ÅŸmasÄ±nÄ± saÄŸlar ve baÅŸka bir cihazda Ã§alÄ±ÅŸmasÄ±nÄ± engeller. SÃ¼rÃ¼m baÄŸlama ise uygulamanÄ±n belirli bir sÃ¼rÃ¼mde Ã§alÄ±ÅŸtÄ±ÄŸÄ±ndan emin olur.

Uygulama Ã–rnekleri:

1. UygulamanÄ±n sadece belirli bir cihazda Ã§alÄ±ÅŸmasÄ±nÄ± saÄŸlayan cihaz baÄŸlama iÅŸlemlerini gerÃ§ekleÅŸtirme.
2. UygulamanÄ±n yalnÄ±zca belirli sÃ¼rÃ¼mlerde Ã§alÄ±ÅŸmasÄ±nÄ± kontrol eden sÃ¼rÃ¼m baÄŸlama iÅŸlemleri.

1.2.0.12 12. TÃ¼ketici DoÄŸrulamasÄ± (Consumer Verification) Teorik AÃ§Ä±klama: UygulamanÄ±n gerÃ§ek kullanÄ±cÄ± tarafÄ±ndan kullanÄ±ldÄ±ÄŸÄ±nÄ± doÄŸrulamak, sahte kullanÄ±cÄ±larÄ± ve otomatik iÅŸlemleri engellemeye yardÄ±mcÄ± olur. Bu doÄŸrulama iÅŸlemi, tÃ¼keticinin kimliÄŸini doÄŸrular.

Uygulama Ã–rnekleri:

1. TÃ¼ketici doÄŸrulamasÄ± iÃ§in gÃ¼venlik testleri ve algoritmalar kullanmak.
2. DoÄŸrulanmamÄ±ÅŸ kullanÄ±cÄ±lar iÃ§in eriÅŸim kÄ±sÄ±tlamalarÄ± koymak.

$6. Hafta - Sonu$