1.1.1 Boş İstisna Bloğu (Detection of Error Condition without Action) (CWE-390)

Açıklık Önem Derecesi: Düşük

Açıklığın Etkisi: Dayanıklılık eksikliği

Açıklığın Barındıran Dosyalar/Satırlar:

Proje Dosyası/Dosya Adı	Satır Numarası

Açıklığın Açıklaması:

Bir istisna bloğu (örn; catch veya finally bloğu) kullanıldığında eğer blok boşsa bu durum uygulamanın güvenilir şekilde çalışmasını engeller.

Örneğin Java dilinde bu açıklığa sahip ve açıklığın kapatıldığı kod bloğu örnekleri verilmiştir.

JAVA:

```java
// KÖTÜ KOD BLOĞU

public class Main {

    public static void main(String[] args) {

        int a = 1;
        int b = 0;
        int c = 0;

        try {
            c = a / b;
        }
        catch(ArithmeticException ae) {

        }

    }
}
```

JAVA:

```java
// İYİ KOD BLOĞU

public class Main {

    public static void main(String[] args) {

        int a = 1;
        int b = 0;
        int c = 0;

        try {
            c = a / b;
        }
        catch(ArithmeticException ae) {
            log.error("Divided by zero detected, setting to -1.");
            c = -1;
        }

    }
}
```

Bu örneklerde kötü kod bloğunda catch bloğu hiçbir satır içermemektedir. Dolayısıyla doğrudan olmasa da dolaylı yoldan bir güvenlik açıklığı oluşturmaktadır. İyi kod bloğunda ise catch bloğu problemi çözen kod satırlarına sahiptir. Dolayısıyla dolaylı güvenlik açıklığı oluşturmamaktadır.

Uygulamalarda catch veya finally blokları boş olduğundan “Boş İstisna Bloğu (CWE-1069)” açıklığı olarak işaretlenirler.

Kurum uygulamasında “Boş İstisna Bloğu” açıklığı tespit edilmiştir:

::::::::BULGU:::::::

Açıklığın Önlemi:

Catch ve Finally blokları boş bırakılmamalıdır ve bu bloklara uygulamanın çalışırlığını ve güvenliğini koruyucu ve devam ettirici kod satırları girilmelidir.

Referanslar:

1. https://cwe.mitre.org/data/definitions/390.html
2.