Yapay Zekâ Tabanlı Faturalandırma Sistemlerinde Gecikme (Latency) Kaynaklı Zafiyet: “AI Latency Exploit” ile Anomali Analizi

1. Giriş ve Problem Tanımı

Konu: Merkezi Yönetim Sistemi (CSMS) içinde çalışan anomali tespiti / faturalandırma karar destek modüllerinin (ör. makine öğrenimi tabanlı AI dedektörleri) gecikmeli (latency) karar verme süresi, zamanlama manipülasyonu yoluyla nasıl istismar edilebileceğini inceler.

Problem Tanımı: CSMS, şarj oturumlarını gerçek zamanlı olarak izler ve şüpheli davranışları tespit etmek için bir yapay zeka modülü (ör. anomali tespit modeli) kullanıyorsa; bu modelin karar süresi birkaç yüz milisaniye ile birkaç saniye arasında değişebilir. Saldırgan, bu gecikme penceresini bilinçli olarak kullanarak StartTransaction / StopTransaction / MeterValues gibi geçerli OCPP mesajlarını, modelin uyarı vermesinden önce ardışık ve zamanlama uyumsuz şekilde gerçekleştirir. Sonuç: AI modülü uyarı verecek kadar geç karar verdiğinde, iki işlem arasındaki maliyet/faturalandırma kısa devreye uğrar ve fatura atlama veya tespit kaçırma gerçekleşir.

Neden Önemli: Mesajlar şemaya tam uyarken (yani ağ/format hatası yokken) sistem mantıksal olarak suistimal edilebilir; bu, yalnızca eşzamanlılık veya state hatası değil, aynı zamanda modern güvenlik mimarilerinin (AI tabanlı tespit) zamanlama zayıflığından kaynaklanan yeni bir saldırı yüzeyidir.

2. Senaryo Analizi: Tehdit Modeli ve Zafiyetin Kök Nedeni

Tehdit Modeli:

- **Saldırgan yeteneği:** Geçerli OCPP istemcisi davranışı (uygulama protokolünü bozmayacak), mesaj gönderme zamanlamasını hassas şekilde kontrol edebilme (milisaniye düzeyinde).
- **Hedef:** CSMS üzerindeki anomali tespit modelinin tespit gecikme penceresini kullanarak faturalandırma veya alarm tetiklemesini atlatmak.
- **Etki:** Faturalarda eksik tahsilat, yanlış durum bildirimleri ve tespit sisteminin güvenilirliğinde azalma.

Zafiyetin Kök Nedeni:

1. **AI gecikmesi (Model Latency):** Anomali tespiti için kullanılan modelin karar alması zaman alır; model girdiyi toplar, ön-işlem yapar, inferans çalıştırır ve alarm/eylem üretir.
2. **Senkronizasyon Eksikliği:** Faturalandırma mantığı ile model alarmı arasında tutarlı bir koordinasyon yoktur — faturalandırma, model uyarısını beklemeden tamamlanabilmekte ya da model uyarısı geciktiği için eylem başarısız olmaktadır.
3. **Event Window (Olay Penceresi) Mismatch:** Sistem, "şüpheli" davranışı belirlemek için geniş bir zaman penceresi kullanıyorsa, saldırgan küçük, hızlı ve ardışık işlemlerle bu pencereyi atlayabilir.
4. **Kötü Tasarlanmış Politikalar:** Şüpheli durumlarda otomatik bloke/rollback yerine gecikmeli manuel inceleme tercih ediliyorsa saldırgan bu gecikmeyi sömürür.

3. Saldırı Vektörü / Anomali Senaryosu

Adımlar (örnek senaryo):

1. Saldırgan, geçerli bir StartTransaction(ID: 2001) başlatır — CSMS kabul eder.
2. Kısa süre sonra (ör. 500 ms içinde) MeterValues(ID: 2001, Value: 1200Wh) gönderir — CSMS kaydeder.

Ardından milisaniyeler içinde StopTransaction(ID: 2001, StopValue: 1200) gönderilir; CSMS faturalandırma sürecini başlatır.

3. Aynı anda saldırgan başka bir oturum (StartTransaction ID: 2002) başlatır ya da Stop sonrası hızlı bir yeniden başlatma ile sistemin kayıtlarını karmaşıklaştırır.
4. CSMS'deki AI anomali tespit modülü gelen veriyi işler; inferans süresi ≈ 2–5 saniye ise, model şüpheli bir patern tespit etse dahi eylem zamanı geçmiş olur — faturalandırma tamamlanmıştır veya veriler kalıcı hale gelmiştir.
5. Sonuç olarak AI uyarısı ya hiç uygulanamaz (zamanında müdahale yok) ya da uygulansa bile geri dönüşümsüz değişiklikler oluşmuştur (ör. banka entegrasyonuna fatura gitmiş).

Gözlemlenebilir Anomali:

- Faturalandırma kayıtlarında beklenmedik boşluklar veya atlanan session kayıtları.
- AI loglarında "gecikmeli uyarı" / "post-facto anomali" ibareleri.
- Gerçek zamanlı dashboardlarda kısa süreli tutarsızlıklar, ancak olay kaydı incelendiğinde bütün mesajların şemaya uygun olduğu görülür.

4. Simülasyon Ortamı ve Metodoloji

Amaç: AI gecikmesi penceresinin nasıl sömürülebileceğini göstermek ve tespit/önlem stratejilerini değerlendirmek.

Kullanılacak Teknolojiler:

- Python 3.10+, asyncio, websockets (basit OCPP benzeri simülasyon), time, logging.
- Basit bir "anomaly_detector" modülü: model inferans süresini taklit eden (sleep ile gecikme) bir fonksiyon.
- SQLite veya in-memory dict ile faturalandırma veritabanı (atomic transaction desteği simülenecek).

Metodoloji:

1. **Flawed CSMS Simülatörü:** Faturalandırmayı anında başlatan, ancak anomali dedektöründen gelecek uyarıyı beklemeyen sunucu.
2. **AI Dedektör Simülatörü:** Gerçek ML kullanmak yerine farklı gecikmeler (örn. 100 ms, 500 ms, 2000 ms) ile sonuç üreten bir mock fonksiyon. Bu fonksiyon belirli paternleri "şüpheli" olarak işaretler.
3. **Saldırgan İstemcisi:** Mesajları hassas zamanlamayla (milisaniye düzeyinde) gönderen istemci; farklı zamanlama stratejileri dener (ör. hızlı ardışık işlemler).

4. **Kayıt ve İzleme:** Hem CSMS hem AI logları, DB kayıtları ve zaman damgaları tutulur; faturalandırma öncesi/sonrası durum karşılaştırılır.
5. **Deney Varyasyonları:** AI gecikmesini 100ms → 2s aralığında değiştir; sonucu ölç (ör. kaç işlem atlandı, kaç uyarı gecikti).

Not: Simülasyon gerçek OCPP ağırlıklı altyapı yerine modüler bir test ortamında yapılmalıdır; gerçek şarj istasyonlarına zarar verilmemelidir.

5. Simülasyon Sonuçları ve Anomali Tespiti

Beklenen Bulgular:

- **Kritik gecikme eşiği tespit edilir:** Model latency > ~500ms olduğunda belirgin şekilde atlanan fatura/uyarı sayısı artar.
- **Post-facto uyarılar:** AI, olayı tespit eder ancak uyarı verildiğinde faturalandırma tamamlanmış ve değişikliklerin geri alınması zorlaşmıştır.
- **Zaman damgası analizi:** İşlem zaman damgaları ile model uyarı zamanları eşleştirildiğinde arada açık bir zaman farkı gözlemlenir.
- **DB tutarsızlıkları:** Bazı transaction kayıtlarında enerji değerleri 0 veya NULL olarak kapanmış, ya da duplicate/kaybolma durumu oluşmuştur.

Örnek Ölçüm Tablosu (simülasyondan alınmış örnek):

- Latency 100 ms → atlanan fatura: 0 / gecikmeli uyarı: 2%
- Latency 500 ms → atlanan fatura: 6% / gecikmeli uyarı: 18%
- Latency 2000 ms → atlanan fatura: 28% / gecikmeli uyarı: 62%

Yorum: AI tabanlı tespit sistemleri yararlı olsa da, gecikmeye duyarlı operasyonlarla korelasyon içinde tasarlanmadığında — özellikle finansal akışın olduğu sistemlerde — tespit sonrası eylem yeterince hızlı değilse güvenlik karşısında etkisiz hale gelebilir.

6. Savunma Stratejileri ve Çözüm Önerileri

Özet: Problemi çözmek için hem mimari hem de uygulama düzeyinde çok katmanlı önlemler gereklidir. Aşağıdaki öneriler pratik ve uygulanabilir niteliktedir.

1. **Senkronizasyon Mekanizmaları (Protective Blocking):**
    a. Faturalandırma işlemleri, AI modülünün kritik tespit penceresini göz önünde bulundurarak "kısa süreli bekleme" (hold) politikasına tabi tutulmalı. Örneğin, StopTransaction geldiğinde faturalandırmayı anında göndermek yerine, küçük bir bekleme penceresi (ör. model latency + güvenlik marjı) eklenebilir. Kritik: bu pencere çok uzun olmamalı, iş akışı bozulmamalı.
2. **Hızlı Ön-Filtreler (Lightweight Heuristics):**
    a. AI modelinin tam inferansını beklemeden, basit kurallarla (thresholds, rate limits) hızlı ön-onay yapılmalı. Örneğin, aynı istasyondan tekrarlı start/stop dizileri tespit ediliyorsa işleme geçmeden öncelikli blok uygulanabilir.
3. **Atomic Transaction & Rollback Yetkinliği:**
    a. Faturalandırma adımları veritabanında atomik olarak yürütülmeli ve AI'dan şüphe gelirse kolayca rollback yapılabilecek biçimde tasarlanmalı. Ayrıca faturalama çıktısı harici sisteme (ör. ödeme sağlayıcı) gönderilmeden önce tekrar doğrulama aşaması konulmalı.
4. **Asenkron Koordinasyon Protokolü:**
    a. AI modülü ile CSMS arasında "commit/confirm" tarzı bir protokol tasarlayın: CSMS provisional_commit → AI_inference → AI_verdict → CSMS final_commit/rollback. Bu, gecikmeyi azaltmaz ama kararlı bir karar zinciri sağlar.
5. **Latency Monitoring & SLA:**
    a. AI servisinin latency'si sürekli izlenmeli ve belirli SLA altında çalışması garanti edilmeli. Model gecikmesi artarsa, otomatik olarak daha sıkı heuristik filtrelere geçilmelidir.
6. **Rate Limiting & Throttling:**
    a. İstemci tarafında (ve/veya gateway tarafında) hızlı ardışık Start/Stop dizilerine karşı rate limiting uygulanmalı; bu, saldırganın milisaniye düzeyinde mesaj bombardımanını zorlaştırır.
7. **Audit Trail & Forensics:**
    a. Tüm mesajlar, model kararları ve faturalandırma adımları detaylı, zaman damgalı log'larla tutulmalı; böylece post-mortem analizler ve hukuki incelemeler için yeterli veri sağlanır.

7. Sonuç ve Değerlendirme

Genel Değerlendirme: Yapay zekâ destekli güvenlik/tespit sistemleri modern CSMS mimarilerinde büyük fayda sağlar; ancak kendi içlerinde yarattıkları gecikme penceresi, özellikle finansal süreçlerle doğrudan entegre edildiğinde yeni bir saldırı yüzeyi oluşturur. Bu çalışmada gösterildiği üzere, saldırganlar geçerli OCPP mesajlarını ve hassas zamanlamayı kullanarak AI tabanlı algılama mekanizmalarını atlatabilir ve fatura atlama ya da tespit kaçırma gerçekleştirebilir.

Çıkarımlar:

- Güvenlik yalnızca doğrulama/şema kontrolü değil; aynı zamanda **zamanlama**, **koordinasyon** ve **sistem tasarımı**dır.
- AI tabanlı tespitler operasyonel süreçlerle sıkı entegre edilmeden kullanıldığında risk yaratabilir.
- Çözüm, hem yazılım mimarisi (atomicity, locking, rollback) hem de AI operasyonel yönergelerinin (latency SLA, model evrimi) birlikte iyileştirilmesini gerektirir.

245541021/UGURBERKTAS