17 Ein Beispiel aus der realen Welt: Google Wallet

Google Wallet (seit 2011): Kontaktlose Bezahlen am Point of Sale

- Kreditkarten werden im Sicherheitselement des Smartphone abgelegt
- Kommunikation über NFC-Schnittstelle (Near Field Communication)

Sicherheitsanforderungen an Bezahlsysteme: Festlegung durch EMVCo (Europay International, MasterCard und VISA)

- Card Verification: Gültigkeit der Karte
- Cardholder Verification: Bindung zwischen Karteninhaber und Karte
- Limit Verification: Prüfung des Verfügungsrahmens

Vertraulichkeit wird nicht gefordert

Mögliche Umsetzungen der Anforderungen:

Card Verification:

- Card Verification Code: symmetrsiches Challenge Response (MAC) (symmetrischer Schlüssel zwischen Karte und Kartenherausgeber)
  - Challence c wird vom POS an Karte geschickt
  - Karte erzeugt MAC aus c und kartenspezifischen Daten d
  - MAC wird zusammen mit c und d an Herausgeber geschickt
- Static Data Authentication: asymmetrische Signatur (kartenspezifische Daten sind vom Herausgeber signiert)
  - Karte schickt Zertifikat (des Herausgebers) an POS
  - Karte schickt kartenspezifische Daten und Signatur an POS
  - POS prüft (mit Zertifikat des Herausgebers)
- Dynamic Data Authentication: asymmetr. Challenge Response

Cardholder Verification (CVM):

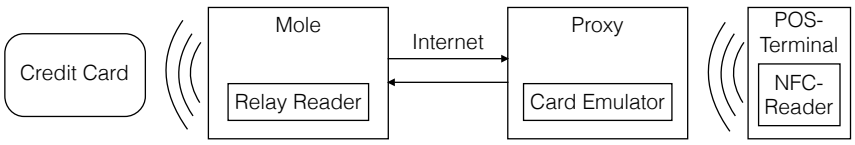
- Online-CVM: Eingabe PIN am POS, Prüfung beim Herausgeber
  kartensp. Daten (aus CV) werden mit PIN an Herausgeber geschickt
- Offline-CVM: Eingabe PIN am POS, Prüfung durch Karte
  - Karte hat Schlüsselpaar für asymm. Verschlüsselung
  - POS erzeugt challenge c, verschl. c und PIN und
  - sendet beides an die Karte
- On Device CVM: Device (z.B. Smartphone) prüft Cardholder-Verif.
  - Keine genaue Spezification durch EMVCo
  - Mögliche Umsetzungen: PIN, Fingerabdruck

Limit Verification: Für unsere Betrachtung unwichtig

Relay-Angriff auf kontaktlose Karten:

Attacke auf Kreditkarten mit kontaktloser NFC-Schnittstelle
werden seit einigen Jahren ausgestellt

- Angreifer nutzt am POS einen Kartenemulator (Proxy)
- Kommunikation mit POS wird an NFC-Reader (Mole) weitergeleitet



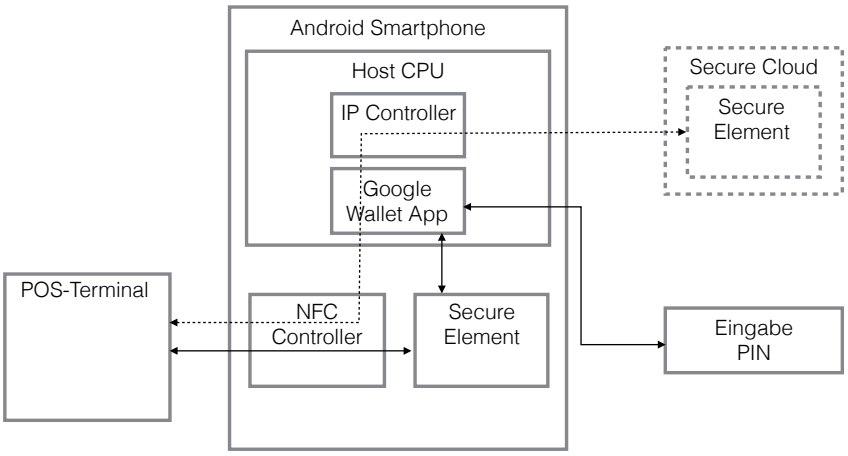
Probleme (aus Sicht des Angreifers):

- PIN für Karte muss vorab ermittelt werden
  nicht bei kleinen Beträgen
- Mole muss in der Nähe der Karte sein (Kommunikation über 10 cm)
  Lösung hier: Nutze Smartphone (mit NFC) des Inhabers

Übertragung auf Google Wallet

Angriff praktisch durchgeführt von M. Roland (2013)
Kosten: 200 Euro (für Smartphone) + 200 Euro (für Notebook)

- Card Verification: Card Verification Code
  - Nutzung eines Sicherheitschips (SE) für Schlüssel und Protokolle
  - Werden bei der Initialisierung sicher übertragen (Over-the-Air-Protocol der Global Plattform)
  - Kommunikation zwischen POS und SE über NFC und Host-CPU
- Cardholder Verification: On Device CVM
  - Nutzung einer 4-stelligen nummerischen PIN
  - PIN wird im Device (nicht im SE) verifiziert
  - Senden des Ergebnisses der Verifikation an SE



Exkurs: Sichere Speicherung von Passwörtern

- PINs und Passwörter sollten nicht im Klartext gespeichert werden
- Daher: Speicherung des Hashwertes (z.B. Sha256, MD5)
  Eingabe PIN → Hashwert PIN → vgl. Hashwerte

- Hashfunktionen nicht umkehrbar, aber Brute Force
  Für Passwort $\in \{A, \ldots, Z, a, \ldots, z, 0, \ldots, 9\}^8$: 62^8 Versuche
- Effizientere Methode: Rainbow-Tables (Time-Memory-Tradeoff)
- Gegenmaßnahme Salt: Speicherung von Salt, Hash(PIN||Salt)
  Eingabe PIN $\rightarrow$ Hashwert PIN||Salt $\rightarrow$ vgl. Hashwerte
- Rainbow-Table müsste für jedes Salt angelegt werden
- Sicherer Pepper (geheimer Salt): sichere Speicherung des Salt

Angreifer kann sich Bot-Netz aus gekaperten Android-Smartphones aufbauen

Was ist noch zu tun:

- Smartphone rooten (erlaubt Zugriff als root)
  Auch ohne Mitwirkung des Nutzers durchführbar
- PIN ermittlen:
  - Google speichert Salt, Hash(PIN||Salt) zusammen ab
    PIN 4-stellig, nummerisch: 10.000 Versuche nötig
  - Key-Logger: Ermitteln der PIN während Eingabe durch Nutzer
  - Google Wallet informiert SE über PIN-Verifikation
    Fälschung des Status
- Installation des Mole auf dem Smartphone
  (Komm. zwischen POS und SE über NFC-Schnittstelle und Host)

Lösungsvorschläge:

- Sicherheit der PIN 1:
  - Speicherung der PIN im SE und nicht im Smartphone
  - PIN wird im SE überprüft (inkl. Fehlbedienungszähler)

  Aber: Key-Logger immer noch möglich
- Sicherheit der PIN 2: Eingabe der PIN am POS (Off- oder Online)
  Keylogging nicht mehr möglich

- Alternative: Fingerabdruckscanner mit direkter Verbundung zum SE
  Umgesetzt in Apple Pay
- Sicherheit Zugriff auf das SE: Nicht über Host-CPU,
  sondern direkte Kommunikation zwischen NFC-Schnittst. und SE