INF523: Assurance in Cyberspace Applied to Information Security

Student Case Studies (Part B)

Prof. Clifford Neuman

Lecture 13B
19 November 2020

Extra Session Thursday 19 November 1:00 PM – 4:20 PM PST

Mobile Devices
- Mobile OS - Chinmaya Pandit and Harshit Kothari
- Android - Mohammed Ababtain

Payment
- Apple Pay - Jairo Hernandez
- Apple Pay and Google Pay - MaryLiza Walker
- Apple Pay - Shanice Williams
- Apple Pay - Yang Xue

Assurance in Payment Systems - Uddipt Sharma
- (this may be at end of Friday Session)

Friday November 20

Operating Systems

- Linux Applications - Aditya Goindi
- Linux - Tejas Pandey
- Chrome OS - Malavika Prabhakar

Infrastructure and Vehicle Control Systems

- US Voting Infrastructure - Anthony Cassar
- Autonomous Vehicles - Chris Samayoa
- Autonomous Vehicles - Amarbir Singh
- Connected and Automated Vehicles - Abhishek Tatti
- Tesla - Dwayne Robinson
- Avionics - Pratyush Prakhar

INF523: Assurance in Cyberspace Applied to Information Security

Student Case Studies (Part B)

Prof. Clifford Neuman

Lecture 13B
19 November 2020

Extra Session Thursday 19 November 1:00 PM – 4:20 PM PST

Mobile Devices
- Mobile OS - Chinmaya Pandit and Harshit Kothari
- Android - Mohammed Ababtain

Payment
- Apple Pay - Jairo Hernandez
- Apple Pay and Google Pay - MaryLiza Walker
- Apple Pay - Shanice Williams
- Apple Pay - Yang Xue

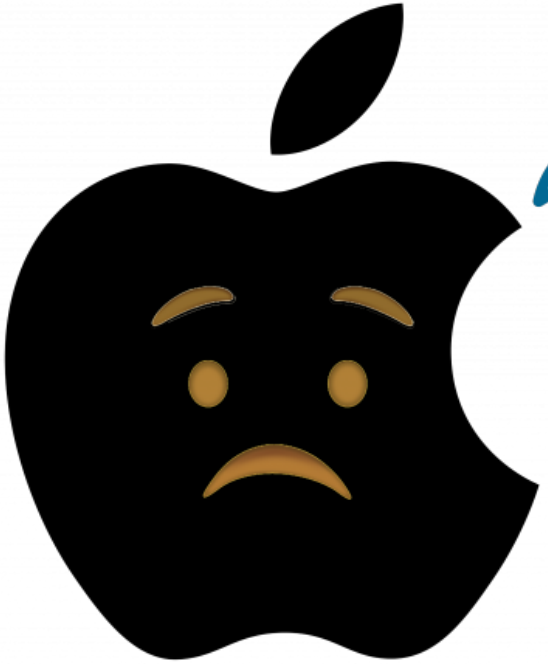
Assurance in Payment Systems - Uddipt Sharma
- (this may be at end of Friday Session)

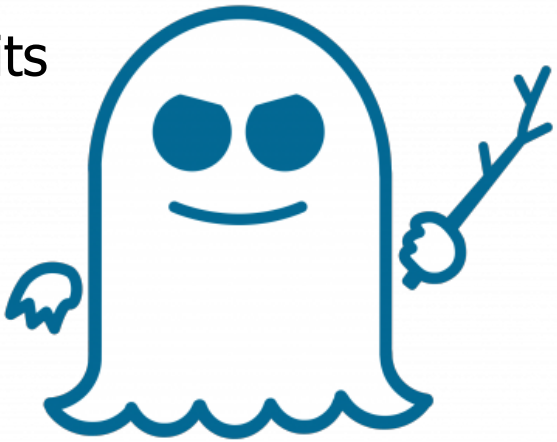
iOS Assurance

Chinmaya Pandit

Threats to the system

Malware and Exploits





iOS Security

iOS is a major leap forward in security for mobile devices

Many layers of defense built into current devices to make attacks by malware and exploitation by attackers difficult.

Key security features like device encryption are not configurable

System security is designed so that both software and hardware are secure across all core components of every iOS device.

Security Architecture

Provides security right from the hardware level (encryption)

System security is designed such that both hardware and software are secure across all core components of iOS.

Architecture is central to security in iOS.

The tight integration of hardware-software ensures that each component of the system is trusted

Each component of the system validates the system as a whole

Every process is analyzed to ensure that the hardware and software are performing optimally and using the resources properly

Minimization

Reduced Attack Surface

If Apple has a vulnerability in some code, and either the attacker can't reach it or Apple doesn't ship the code at all in iOS, an attacker cannot base an exploit on this vulnerability. Therefore, a key practice is minimizing the amount of code an attacker can access, especially remotely

Stripped Down iOS

Beyond just reducing the potential code an attacker might exploit, Apple also stripped down the number of useful applications an attacker might want to use during and after exploitation.

Example - The most obvious example is that there is no shell (/bin/sh) on an iOS device.

Minimization

Privilege Separation

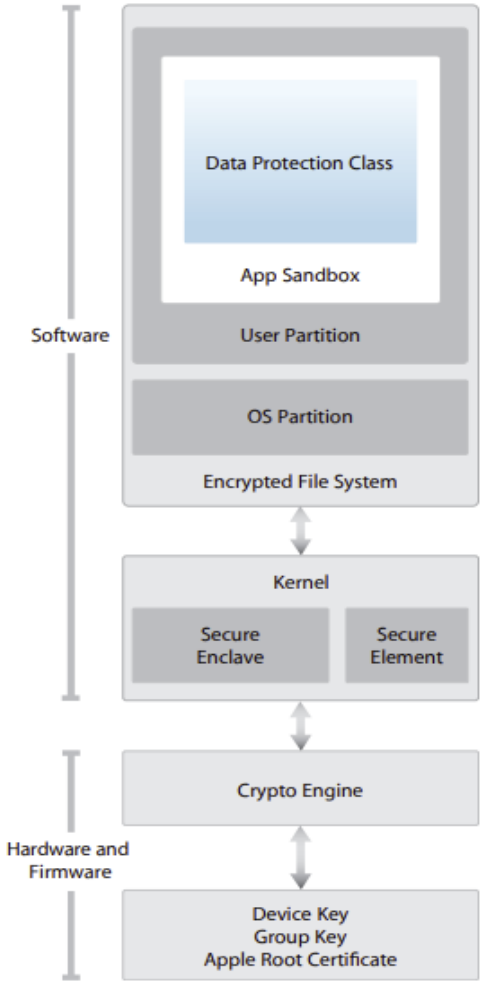
iOS separates processes using users, groups, and other traditional UNIX file permission mechanisms.

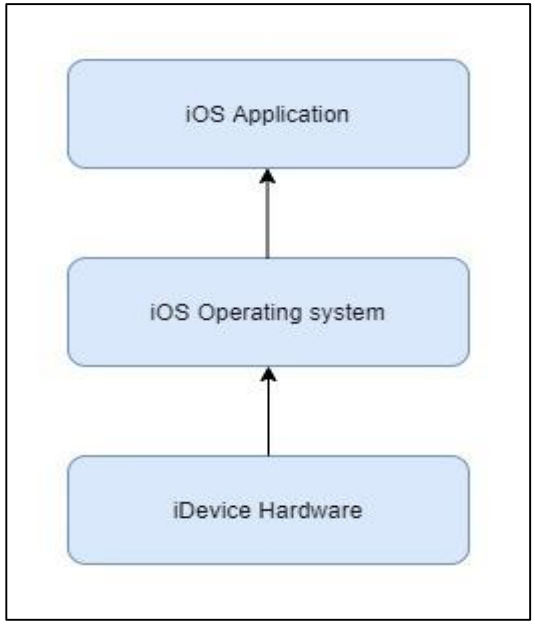
mobile

root

_wireless

_mdnsresponder


Data Protection Class
App Sandbox
User Partition
OS Partition
Encrypted File System
Software
Kernel
Secure Enclave
Secure Element
Crypto Engine
Hardware and Firmware
Device Key
Group Key
Apple Root Certificate


iOS Application
iOS Operating system
iDevice Hardware

Security Architecture

At the software layer:

- The application sandbox
- Partitions that house user data and operating system data, separately,
- Data protection classes that determine when data from apps is accessible
- A file system that stores data and files that contribute to the device's functionality.

At the OS layer:

- The **kernel**, or core component of the phone's operating system,
- The **secure enclave**, which handles device keys and other security features such as Touch ID
- The **secure element**, which helps ensure the security of things like payment apps (Apple Pay)

At the hardware level:

- A **crypto engine**, responsible for encrypting and decrypting important files
- **Keys** for security of the device and root certificates that show trust between iOS and apps.

Secure Enclave

It is a coprocessor. Fabricated in Apple A7 and later A-series processors

Includes a hardware-based key manager, which is isolated from the main processor to provide an extra layer of security.

Each Secure Enclave is provisioned during fabrication with its own UID (Unique ID) that is not accessible to other parts of the system and is not known to Apple.

No-one can have access to this key, and it is burnt into the chip.

There are other coprocessors like the image sensor and the motion sensor coprocessor. Apple uses a mechanism called System Coprocessor Integrity Protection to prevent the modification of the coprocessor firmware

Tasks of Secure Enclave

Its main focus is to provide a trusted boot environment, and where it checks from registers that the boot process has not been interfered with.

On iPhone,, the Secure Enclave manages the authentication process and enables a payment transaction to proceed.

Enclave stores encryption keys used to lock down that biometric data.

Responsible for processing fingerprint data from the Touch ID sensor. Enables Touch ID and Face ID.

Enabling access or purchases on behalf of the user.

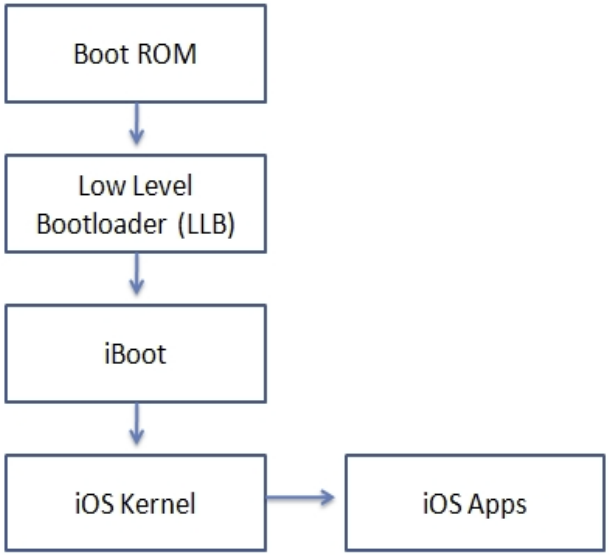
Makes it very difficult for hackers to decrypt sensitive information without physical access to your device.

Third party programs can create and store keys in the enclave to lock down data

Secure boot chain

Uses secure boot chain to provide security in the booting process.

Ensures that the low-level software is not compromised and iOS is running on validated iDevice.



How does Apple protect your data?

Apple uses **Data Protection** to protect data stored in flash storage on the device.

Data Protection allows the device to respond to common events, such as incoming phone calls, but also enables a high level of encryption for user data.

Key system apps, such as Messages, Mail, Calendar, Contacts, Photos and Health data values use Data Protection by default, and third-party apps receive this protection automatically.

Data Protection is implemented by constructing and managing a hierarchy of **keys**

Controlled on a per-file basis by assigning each file to a **class**

Apple Security Assurance

Apple pursues a comprehensive approach with security certifications to provide customers with the appropriate assurance for all Apple platforms.

With comprehensive development and management of the whole platform from silicon through to the operating system, services and apps.

H/S certifications and validations - certification building blocks - corecrypto, secure enclave processor, secure element

Cryptographic Module Validations - FIPS 140-2 / 3 (ISO / IEC 19790)

Product Certifications - Common Criteria ISO/IEC 15408

Service Certifications - ISO/IEC 27001 and 27018 to enable Apple customers to address their regulatory and contractual obligations. These certifications provide our customers with an independent attestation regarding Apple Information Security and Privacy practices for in-scope systems.

Apple’s ISO Certifications cover the following security domains

ISO 27001 - ISMS

- Information security policies
- Organization of information security
- Asset management
- Human resources security
- Physical and environmental security
- Communications and operations management
- Access control
- Information systems acquisition, development, and maintenance
- Information security incident management
- Business continuity management
- Compliance

ISO 27018 - Protection of personally identifiable information (PII) in public cloud environments.

- Consent and choice
- Purpose legitimacy and specification
- Collection limitation
- Data minimization
- Use, retention, and disclosure limitation
- Accuracy and quality
- Openness, transparency, and notice
- Individual participation and access
- Accountability
- Information security
- Privacy compliance

Security controls

Encryption and Data Protection

Hardware security features
File Data Protection
Passcodes
Data Protection classes
Keychain Data Protection
Access to Safari saved passwords
Keybags FIPS 140-2

App Security

App code signing
Runtime process security
Extensions
App Groups
Data Protection in apps

Network Security

SSL, TLS
VPN
Wi-Fi
Bluetooth
Single Sign-on
AirDrop security

Device Controls

Passcode protection
iOS pairing model
Configuration enforcement Mobile device management (MDM)
Device Enrollment Program
Apple Configurator
Device Restrictions
Supervised-only restrictions
Remote wipe
Find My iPhone and Activation Lock

iOS Assurance

Harshit Kothari

iOS Security Architecture

- Code Signing
- Data Execution Prevention
- Address Space Layout Randomization

Code Signing

- One of the most important security mechanisms in iOS is code signing. All binaries and libraries must be signed by a trusted authority (such as Apple) before the kernel will allow them to be executed.
- Furthermore, only pages in memory that come from signed sources will be executed.
- This means apps cannot change their behavior dynamically or upgrade themselves.
- Together, these actions prevent users from downloading and executing random files from the Internet.

Code Signing

- All apps must come from the Apple App Store (unless the device is configured to accept other sources).
- Apple has the ultimate approval and inspects applications before they can be hosted at the App Store.
- In this way, Apple plays the role of an antivirus for iOS devices. It inspects each app and determines if it is okay to run on iOS devices.
- This protection makes it very hard to get infected with malware. In fact, only a few instances of malware have ever been found for iOS.

Code Signing

- The other impact of code signing is that it complicates exploitation.
- Once an exploit is executing code in memory, it might want to download, install, and execute additional malicious applications.
- This will be denied because anything it tries to install will not be signed.
- Therefore, exploits will be restricted to the process they originally exploit, unless it goes on to attack other features of the device.
- This code signing protection is, of course, the reason people jailbreak their phones.
- Once jailbroken, unsigned applications can be executed on the device.
- Jailbreaking also turns off other security features.

Data Execution Prevention

- Normally, data execution prevention (DEP) is a mechanism whereas a processor can distinguish which portions of memory are executable code and which portions are data; DEP will not allow the execution of data, only code.
- This is important because when an exploit is trying to run a payload, it would like to inject the payload into the process and execute it.
- DEP makes this impossible because the payload is recognized as data and not code.
- The way attackers normally try to bypass DEP is to use return-oriented programming (ROP).
- ROP is a procedure in which the attacker reuses existing valid code snippets, typically in a way never intended by the process, to carry out the desired actions.

Data Execution Prevention

- The code-signing mechanism in iOS acts like DEP but is even stronger. Typical attacks against DEP-enabled systems use ROP briefly to create a section of memory that is writable and executable (and hence where DEP is not enforced). Then they can write their payload there and execute it.
- However, code signing requires that no page may be executed unless it originates from code signed by a trusted authority.
- Therefore, when performing ROP in iOS, it is not possible to turn off DEP like an attacker normally would.

Data Execution Prevention

- Combined with the fact that the exploit cannot execute applications that they may have written to disk, this means that exploits must only perform ROP.
- They may not execute any other kinds of payloads such as shellcode or other binaries.
- Writing large payloads in ROP is very time-consuming and complex.
- This makes exploitation of iOS more difficult than just about any other platform.

Address Space Layout Randomization

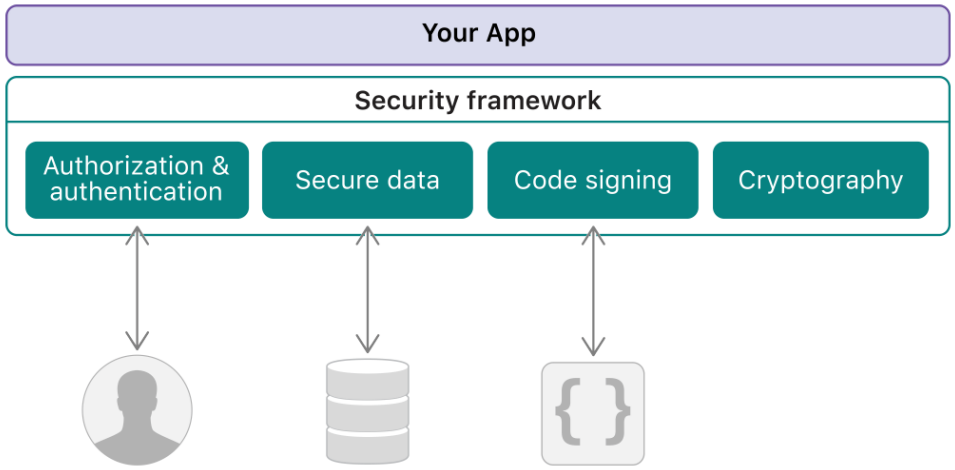
- As discussed in the previous section, the way attackers try to bypass DEP is to reuse existing code snippets (ROP). However, to do this, they need to know where the code segments they want to reuse are located.
- Address space layout randomization (ASLR) makes this difficult by randomizing the location of objects in memory.
- In iOS, the location of the binary, libraries, dynamic linker, stack, and heap memory addresses are all randomized.
- When systems have both DEP and ASLR, there is no generic way to write an exploit for it.
- In practice, this usually means an attacker needs two vulnerabilities: one to obtain code execution and one to leak a memory address in order to perform ROP.

iOS Applications

Apps are sent to Apple for review. If approved, they are signed by Apple's private key and pushed out to App store for download.

Apps need to be signed by Apple, or they will not run on the device due to mandatory code signing requirement in iOS.

Additionally app store apps use sandbox at a low privilege level to reduce the damage they can cause.



Assurance Techniques - Sandboxing

- The final piece of the iOS defense is sandboxing. Sandboxing allows even finer-grained control over the actions that processes can perform than the UNIX permission system mentioned earlier.
- For example, both the SMS application and the web browser run as user mobile, but perform very different actions.
- The SMS application probably doesn't need access to your web browser cookies and the web browser doesn't need access to your text messages.
- Third-party apps from the App Store shouldn't have access to either of these things.
- Sandboxing solves this problem by allowing Apple to specify exactly what permissions are necessary for apps.

Sandboxing

- Sandboxing has two effects. First, it limits the damage malware can do to the device.
- If you imagine a piece of malware being able to get through the App Store review process and being downloaded and executed on a device, the app will still be limited by the sandbox rules.
- It may be able to steal all your photos and your address book, but it won't be able to send text messages or make phone calls, which might directly cost you money.

Sandboxing

- Sandboxing also makes exploitation harder.
- If an attacker finds a vulnerability in the reduced attack surface, manages to get code executing despite the ASLR and DEP, and writes a productive payload entirely in ROP, the payload will still be confined to what is accessible within the sandbox.
- Together, all of these protections make malware and exploitation difficult, although not impossible.

Secure software authorization

Apple regularly releases software updates to address emerging security concerns and also provide new features; these updates are provided for all supported devices simultaneously.

Users receive iOS update notifications on the device and through iTunes, and updates are delivered wirelessly, encouraging rapid adoption of the latest security fixes.

The startup process helps ensure that only Apple-signed code can be installed on a device. To prevent devices from being downgraded to older versions that lack the latest security updates, iOS uses a process called System Software Authorization.

Random number generator

Cryptographic pseudo-random number generators (CPRNGs) are an important building block for secure software.

Apple provides a trusted software CPRNG running in the iOS

It’s responsible for aggregating raw entropy from the system and providing secure random numbers to consumers in both the kernel and user space.

Hardware security

Secure software requires a foundation of security built into hardware.

All modern iPhones with a dedicated AES hardware engine to power line-speed encryption as files are written or read. This ensures that Data Protection and FileVault protect users' files without exposing long-lived encryption keys to the CPU or operating system.

On iOS devices, security begins in immutable code called the Boot ROM, which is laid down during chip fabrication and known as the hardware root of trust.

The security features of Apple devices are made possible by the combination of silicon design, hardware, software and services available only from Apple.

Privacy controls

Location Services

Access to personal data

Privacy policy

Jailbreaking as an assurance issue?

- Jailbreaking is the process by which Apple users can remove software restrictions imposed on iOS and Apple products.
- Jailbreaking allows root access to iOS and lets users install applications, extensions, and other software applications that are not authorized by Apple's App Store.
- Rooting refers to the same process on Android smartphones.

Is Jailbreaking even legal?

- The legality of jailbreaking a device falls under the Digital Millennium Copyright Act (DMCA). Section 1201's of DMCA has divided technological measures into two categories:
- Measures that prevent unauthorized access to a copyrighted work
- Measures that prevent the "copying" of a copyrighted work.
- According to the act, "Making or selling devices or services that are used to circumvent either category of a technological measure is prohibited in certain circumstances."
- However, jailbreaking a device is exempt from these criteria.

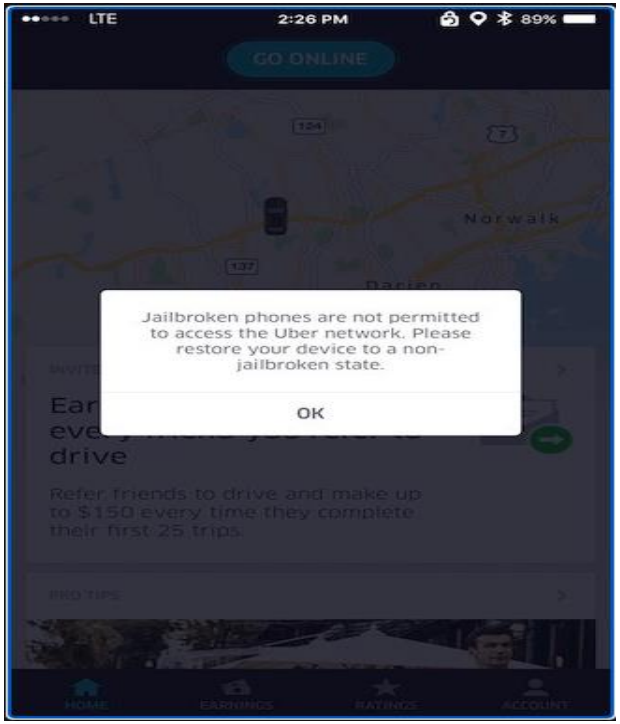
Is Jailbreaking even legal?

- According to the U.S. Copyright Office's Federal Register Notice Vol. 75, No. 143:
- Turning to the second fair use factor, it is customary for operating systems – functional works – to enable third party programs to interoperate with them. It does not and should not infringe any of the exclusive rights of the copyright owner to run an application program on a computer over the objections of the owner of the copyright in the computer's operating system. Thus, if Apple sought to restrict the computer programs that could be run on its computers, there would be no basis for copyright law to assist Apple in protecting its restrictive business model."

Warning: Jailbreaking an iPhone Voids Your Warranty

Apple strongly cautions against installing any software that hacks iOS. It is also important to note that unauthorized modification of iOS is a violation of the iOS end-user software license agreement and because of this, Apple may deny service for an iPhone, iPad, or iPod touch that has installed any unauthorized software.”

Apple’s Untrusted Domain



References

Links

https://www.apple.com/ph/privacy/docs/iOS_Security_Guide.pdf

https://support.apple.com/en-gb/guide/security/secc1ffa3a47/web

https://study.com/academy/lesson/ios-security-architecture-layers-features.html

Books

iOS Hacker's Handbook

Learning iOS Penetration Testing

Android Assurance

Mohammed Ababtain

About Android

- Owned By Google
- Open source based on modified Linux kernel
- Released on September 2008
- 87% market share of the global market in 2019
- Around 50% market share of US market
- 2.5 billion active users
- Runs on ARM and x86 architectures

Android Assurance Mechanisms

- Secure Element
- Trusted Execution Environment
- TCB
- Verified boot
- Encryption
- Sandboxing
- Application Signing
- MAC
- Rooting of Devices

Secure Element

- Separate microchip, with its own CPU, storage, RAM, etc.
- Designed specifically for security-relevant purposes. (Payments)
- Provides resistant to wide variety of attacks, both logical and physical, such as side-channel attacks.
- More strictly as a discrete separate hardware than Trusted Execution Environment(TEE).

Trusted Execution Environment:

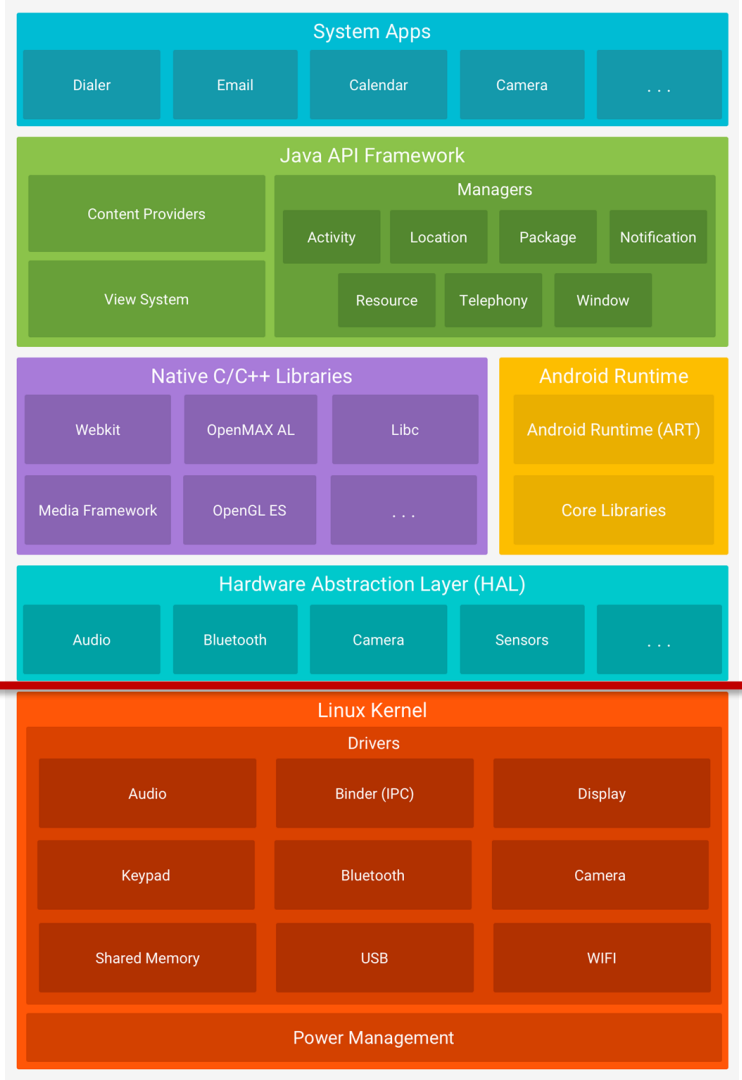
- Isolated environment called a Trusted Execution Environment (TEE) . This enables further separation from any untrusted code.
- TEE is responsible for some of the most security-critical operations on the device, including:
    - Lock screen passcode verification
    - Fingerprint template matching
    - Protection and management of KeyStore keys

Lock Screen Biometrics Requirement

- A false acceptance rate not higher than 0.002% and a false rejection rate of less than 10%.
- After 5 rejected attempts, leave at least 30 seconds between subsequent attempts.
- All identifiable fingerprint data must be encrypted and cryptographically authenticated such that they cannot be acquired, read, or altered outside of the TEE.

TCB

Trust Boundary



Verified boot

- Uses Device-Mapper’s verity (DM verity) from Linux kernel
- Root hash is the root node of the tree. Its value is based on all the hashes of all the lower levels.
- The leaf nodes are the hash of the physical data blocks. Intermediate nodes are hashes of their below nodes.
- Verifying the root hash is sufficient to show the blocks are genuine
- The system will boot only on success

Encryption

- Full disk encryption using random 128-bit AES key
- The master key is encrypted using another 128-bit key called KEK
- KEK is usually derived from a password supplied by the user
- It protect the data while at rest but not when the volume is mounted

Sandboxing

- Each application is assigned a unique UID and a private directory to read/write data
- This will prevent applications from accessing others' private directories
- Applications can grant others to access their private directory. (DAC)
- Sharing Access is not recommended but available.

Application Signing

- Every Application has to be signed by the developer
- A digital certificate, that identify the developer, must be attached to the application
- Applications cannot updated if signed with different signature
- Application signing could be from third-party or self signing
- Android does not use CA for certificate

Mandatory Access Control

- Android uses DAC as a default access control.
- MAC was added later by using SELinux
- Isolate system services and user applications in different domains and each domain has a specific access policy
- Has two modes: permissive and enforced

Rooting of Devices

- By default, only the kernel and small number of core applications runs with root.
- Android does not prevent a user or application with root permission from modifying the OS or kernel
- Root access can be gained by unlocking the bootloader to instal different OS.
- The bootloader unlock mechanism requires the erase of any existing user data.

Does it work?

- Android had the most vulnerabilities of any operating system in 2019, 2017, and 2016.
- Most devices don’t receive these security updates on time

Recommendation

- Encrypt by default
- Enforce MAC
- The use of CA
- Minimization
- Patch time

Resources

- https://source.android.com/security/reports/Google_Android_Enterprise_Security_Whitepaper_2018.pdf
- https://www.statista.com/statistics/272307/market-share-forecast-for-smartphone-operating-systems/#:~:text=Smartphones%20running%20the%20Android%20operating,percent%20share%20of%20the%20market
- https://www.e-consystems.com/blog/system-on-module-SOM/android-hal-and-device-driver-architecture/
- https://source.android.com/security
- https://www.fastcompany.com/90473597/android-had-the-most-vulnerabilities-of-any-os-in-2019-says-report

THANK YOU!

Apple and Google Pay

Presented to DSCI 523, Fall 2020

Presented by: MaryLiza Walker, Yang Xue, Jairo Hernandez, Shanice Williams

Agenda

- Brief history of credit cards
- Introduction of Apple Pay
- Adding payment options to Apple Pay
- Paying with Apple Pay
- Comparing Apple Pay to Credit Cards
- Introduction of Google Pay
- Google Pay Payment Process
- Comparing Apple Pay to Google Pay
- Assurances of Apple Pay
- Possible Subversions
- Suggestions for improvements

Where do we come from?

A quick history of Credit Cards:

- 1960’s Magnet Stripes were introduced.
- Early 2000’s RFID technology was introduced.
- 2010’s Digital Wallets technology was introduced.
- Mid 2010’s EMV chip cards were introduced.
- Future - Biometrics reading machines available at check out.

What is Apple Pay?

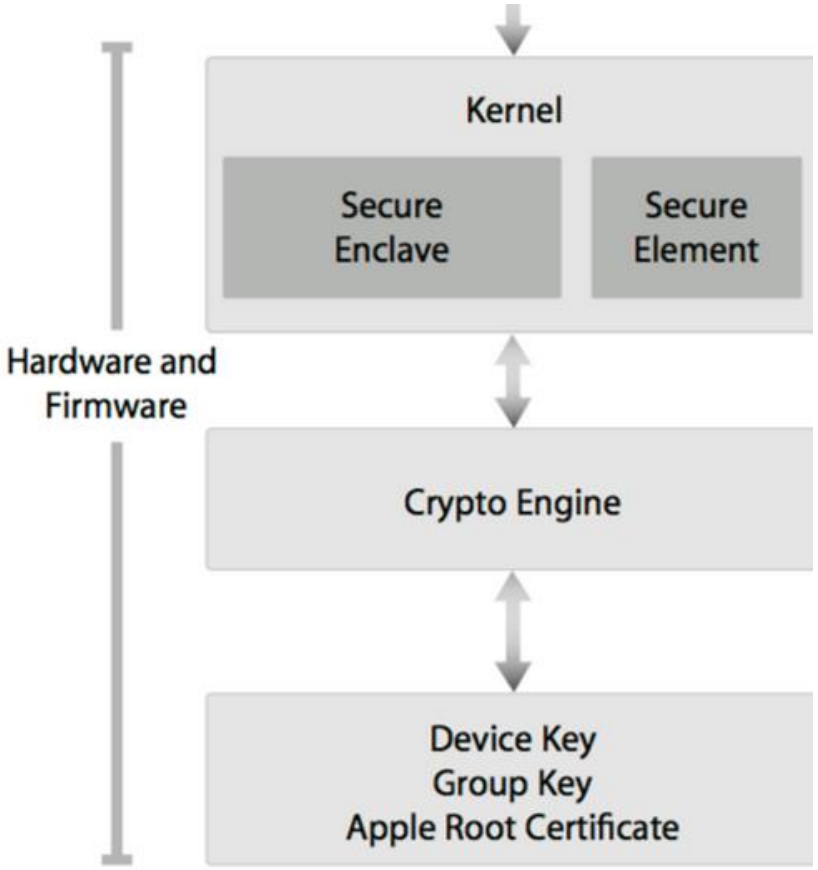
- First launched in October 2014
- It allows for users to pay in an simple and secure way, using any apple device
- You can use it in stores, online, apps or peer to peer
- It was designed to protect users personal information.
- Apple states that they do not collect users payment or transactions information

Components Involved in Apple Pay

NFC controller	Manages the communication between secure element and POS
Wallet application	App used to add and manage linked cards
Apple Pay servers	Manages DAN distributions that are stored in Secure Element
Secure Element	Manages all financial transactions
Secure Enclave	Manages authentication

Secure Element

- A chip running Java Card platform
- Located in iPhone Security Kernel
- Designed to host payment networks
- NFC controllers communicate with secure element through dedicated hardware
- Stores and secures Device Account Numbers



Secure Element (SE)

A Secure Element (SE) is a **tamper-resistant** platform (typically a one chip secure microcontroller) capable of securely **hosting applications and their confidential and cryptographic data** in accordance with the **rules and security requirements** set forth by a set of well-identified trusted authorities.

GlobalPlatform is a non-profit industry association driven by over 100-member companies.GlobalPlatform is an organization that has been established by leading companies from the **payments and communications industries**. One element of its work is the standardization and interoperability of application management within an SE..

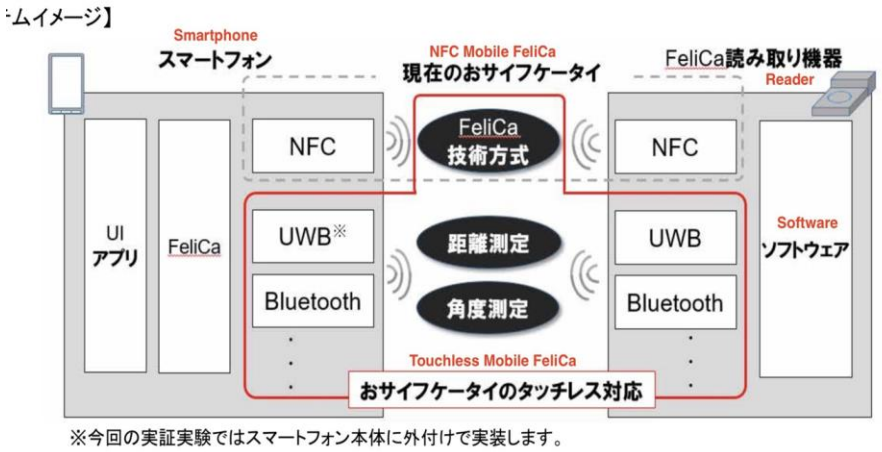
GlobalPlatform Card Specification v2.2 [12] is a proven foundation on which many GlobalPlatform SE technical documents are developed. It defines **card components, command sets, transaction sequences and interfaces**.

How Apple Pay uses the Secure Element and NFC controller

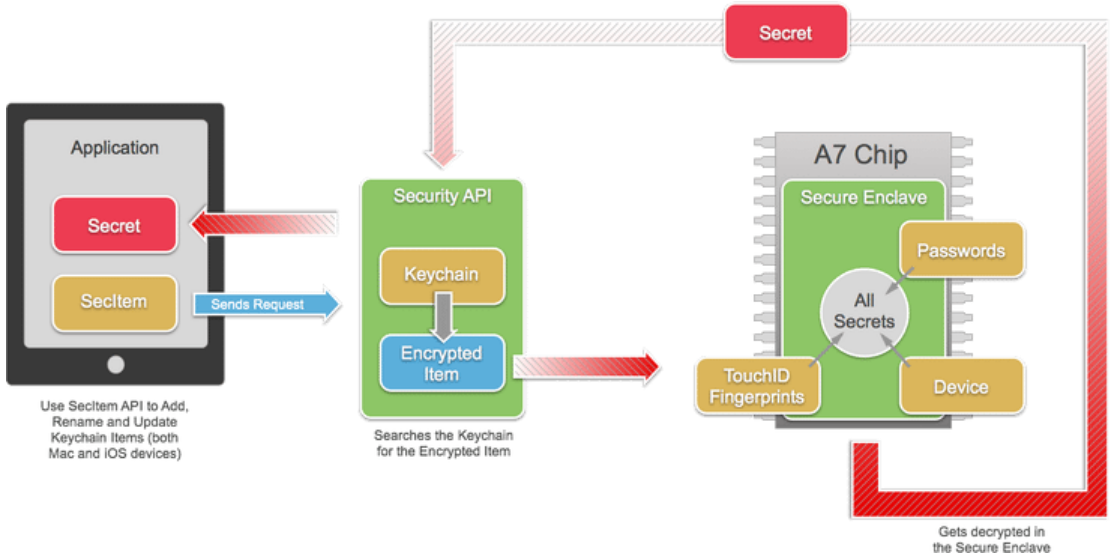
Secure Element:
worked hand in glove with the **NFC controller**
hosts a specially designed applet to manage Apple Pay **certified** by payment networks or card issuers.

NFC controller:
gateway to the Secure Element
contactless payment transactions are conducted in close proximity with the device.

Contactless Responses:
encrypted by the payment applets in SE
exclusively **routed** by the controller to the NFC field
never **exposed** to the application processor



Using Find My iPhone, or restoring their device mode, IOS will instruct the Secure Element to mark all cards as deleted



Secure Enclave

- Located in iPhone Security Kernel
- Responsible for authenticating users to their device
- Manages all devices encryptions keys
- Communicates with Secure Element via a serial interface

Adding Card to Apple Pay

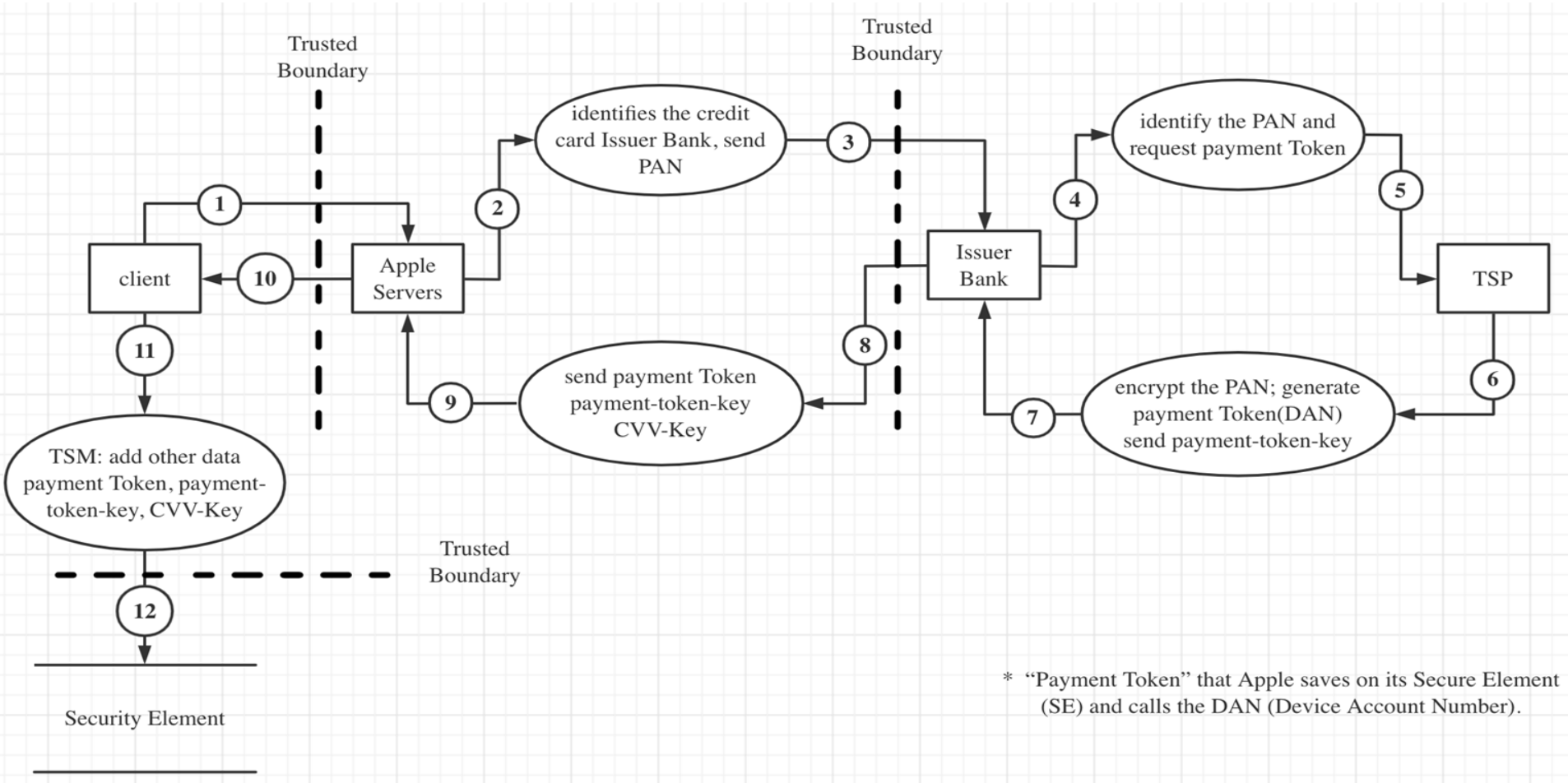
- You can add a card via iTunes/Apple Store account, from the card issuer's bank app or manually inserting the card info.
- When adding card, it goes through 3 phases
  - Required Fields
  - Check Card
  - Link and Provision

Apple pay process

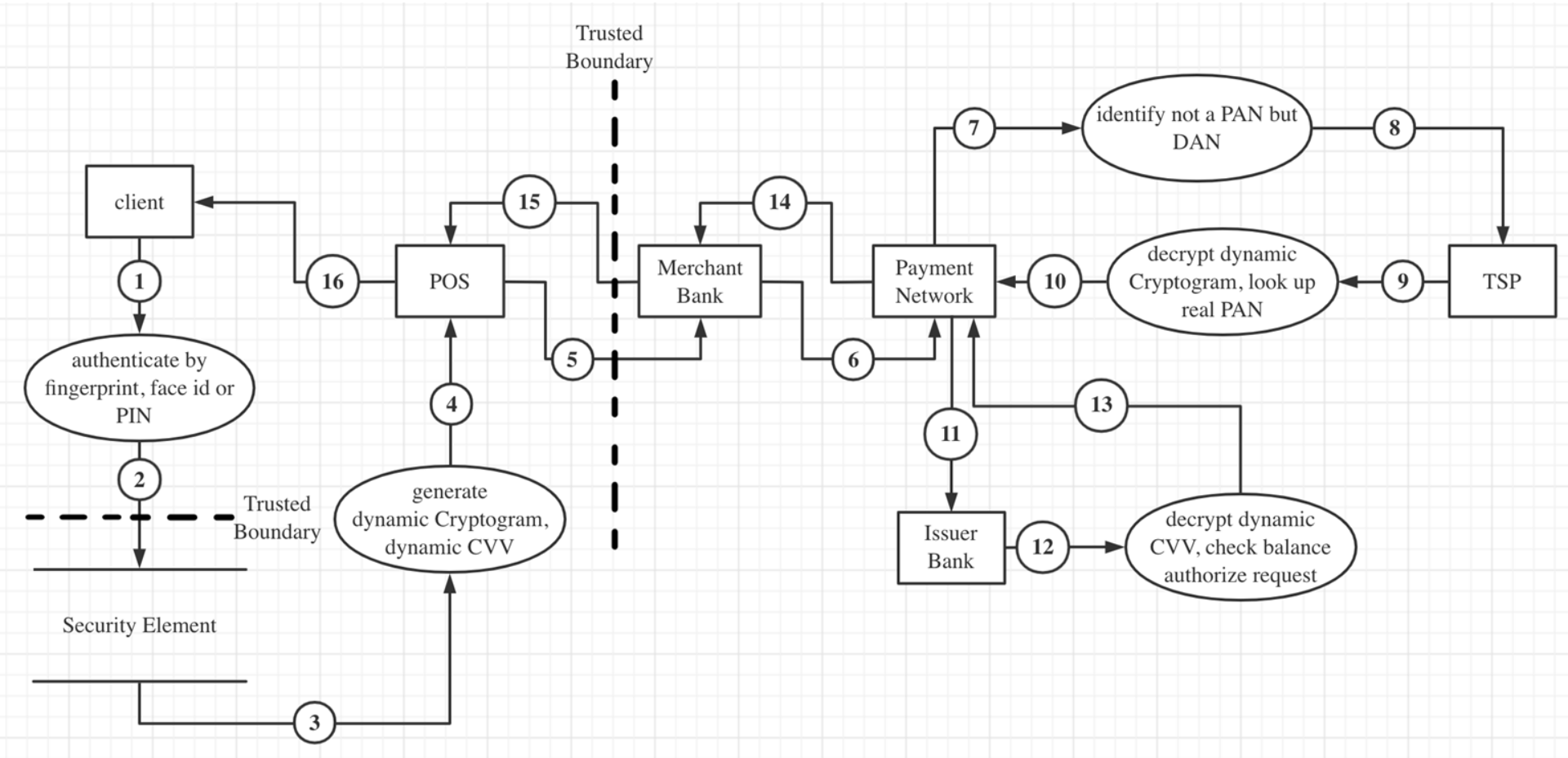
Phase 1: Adding a Card to Apple Pay

Phase 2: Initiating a Transaction Using Apple Pay

Phase 1 DFD for Add Payment Card to Apple Pay



Phase 2 When you Pay using Apple Pay to operate contactless payment



Encrypt Hash Signature

- Credit Card Auth API request example
- CardAuthReply message example [8]

```
</WalletPaymentInformation>
<OrderId>1</OrderId>
<BillingFirstName>John</BillingFirstName>
<BillingLastName>Smith</BillingLastName>
<BillingPhoneNo>6101234567</BillingPhoneNo>
<BillingAddress>
   <Line1>123 Main St</Line1>
   <Line2>Building 123</Line2>
   <Line3>4th Floor</Line3>
   <Line4>Apt 12</Line4>
   <City>Philadelphia</City>
   <MainDivision>PA</MainDivision>
   <CountryCode>US</CountryCode>
   <PostalCode>19019</PostalCode>
</BillingAddress>
<CustomerEmail>customer@sample.com</CustomerEmail>
<CustomerIPAddress>208.247.73.130</CustomerIPAddress>
<ShipToFirstName>John</ShipToFirstName>
<ShipToLastName>Smith</ShipToLastName>
<ShipToPhoneNo>6101234567</ShipToPhoneNo>
<ShippingAddress>
```

```
<?xml version="1.0" encoding="UTF-8"?>
   <CreditCardAuthReply xmlns="http://api.gsicommerce.com/schema/checkout/1.0">
      <PaymentContext>
         <OrderId>1</OrderId>
         <!-- You will receive a token in the response, which is a scrambled version of the Credit Card number.
            This token gets passed to the Order Service, not the original credit card number -->
         <PaymentAccountUniqueId isToken="true">4054131ImpMJ6965</PaymentAccountUniqueId>
      </PaymentContext>
      <AuthorizationResponseCode>APPROVED</AuthorizationResponseCode>
      <BankAuthorizationCode>614653</BankAuthorizationCode>
      <CVV2ResponseCode>0</CVV2ResponseCode>
      <AVSResponseCode>E</AVSResponseCode>
      <PhoneResponseCode></PhoneResponseCode> <!-- AmEX only -->
      <NameResponseCode></NameResponseCode> <!-- AmEX only -->
      <EmailResponseCode></EmailResponseCode> <!-- AmEX only -->
      <AmountAuthorized currencyCode="USD">14.99</AmountAuthorized>
      <TenderType>VC</TenderType>
   </CreditCardAuthReply>
```

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<CreditCardAuthRequest xmlns="http://api.gsicommerce.com/schema/checkout/1.0" requestId="1234567">
   <WalletPaymentInformation>
     <ApplePayTransactionId>84a7e17570c940f6a268d9121a4c090f1d2e22bab4da0f2054c7ba53035dc79a</ApplePayTransactionId>

<EphemeralPublicKey>MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAEAtpj8+ZI9S3gQ7QHCM/dLYTjdSgKdXlDDI/ezdwmHc4HosSYFdYYE/v8t2CcF
VRO57W3VxeAg==</EphemeralPublicKey>
     <PublicKeyHash>i3F+FlIbyhjfYqMqm3M/dpeWRO9tnD+U9BJng3tkkig=</PublicKeyHash>
     <Version>EC_v1</Version>

<Data>GcUsL3ZgQVf9Raf7fBY+0AXsoO/5REeQWE6mROAJM4QvxJgOO6mxW1CuM1P7Ox9hQo1Qt1dg/VIS5fszNq7YhB0oQNgoEwAh7bTXEUBtmx0lzvN
v+7CuakmRG6c6hO4xQlayCNGV6diFK1Ng6zNphSW53b1Di6vhqJcDnGs2tvu6wHyGfqIH6AeUmCNsiBvkHsaiiHbiWcL6BPpWuoW7y5bZK3mMxgygYgzZ
pPDcL8ouuVU/oBMZn9ox/Ql3r9E8f3g7+uTdKZ+TEoyKIH52VQUb1/YXU3SZ2Fr2J/oidUPz9+fTvGZkFPQvSX4G+jzYWlCQmrx6qZ0cfTY/ZcGmITSSb
qmL8wqmOzfr202R3sXeGBjS1BqAm5DNzbgWHGrOoc/I=</Data>

<Signature>MIAGCSqGSIb3DQEHAqCAMIACAQExDzANBglghkgBZQMEAgEFADCABgkqhkiG9w0BBwEAAKCAMIID4jCCA4igAwIBAgIIJEPyqAad9XcwCg
wejEuMCwGA1UEAwwlQXBwbGUgQXBwbGljYXRpb24gSW50ZWdyYXRpb24gQ0EgLSBHMzEmMCQGA1UECwwdQXBwbGUgQ2VydGlmaWNhdGlvbiBBdXRob3Jp
AoMCkFwcGxlIEluYy4xCzAJBgNVBAYTAlVTMB4XDTE0MDkyNTIyMDYxMVoXDTE5MDkyNDIyMDYxMVowXzElMCMGA1UEAwwcZWNjLXNtcC1icm9rZXItc2
PRDEUMBIGA1UECwwLaU9TIFN5c3RlbXMxEzARBgNVBAoMCkFwcGxlIEluYy4xCzAJBgNVBAYTAlVTMFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAEwhV3
JChIY3HsL1vLCg9hGCV2Ur0pUEbg0IO2BHzQH6DMx8cVMP36zIg1rrV1O/0komJPnwPE6OCAhEwggINMEUGCCsGAQUFBwEBBDkwNzAlBggrBgEFBQcwAY
```

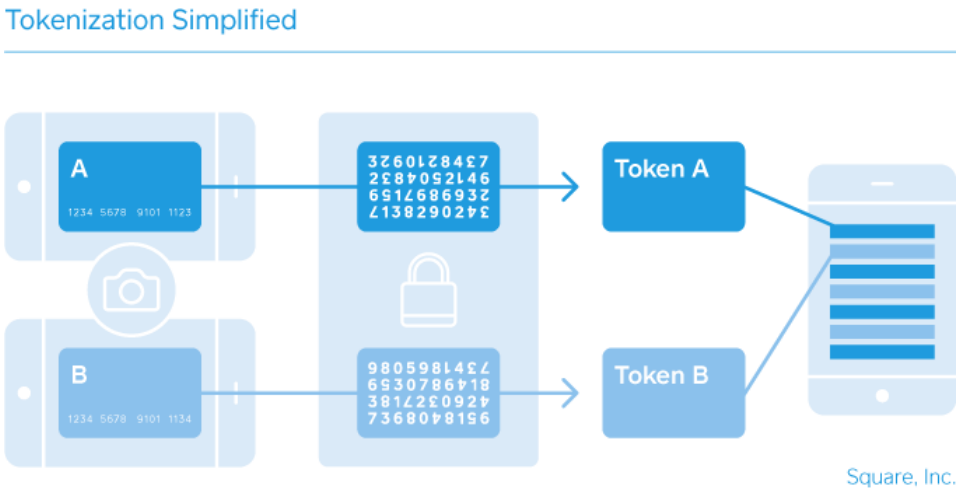
Tokenization

1, The Payment token or Device Account Number (DAN) is a permanent unique number and does not change. The DAN acts as a proxy for the real credit card number (PAN) and the personal details.

2, DAN is unique to that particular iPhone device. The same card added to a different device will have a different DAN.

3, Any transaction records for purchases made using Apple Pay will not show the last 4 digits of your credit card. Rather the transaction records will show the last 4 digits of the DAN.

4, Apple Pay does not store the real card numbers on the device or Apple servers or inside the Secure Element, and payment token data never stored in their cloud servers iCloud [6,7].



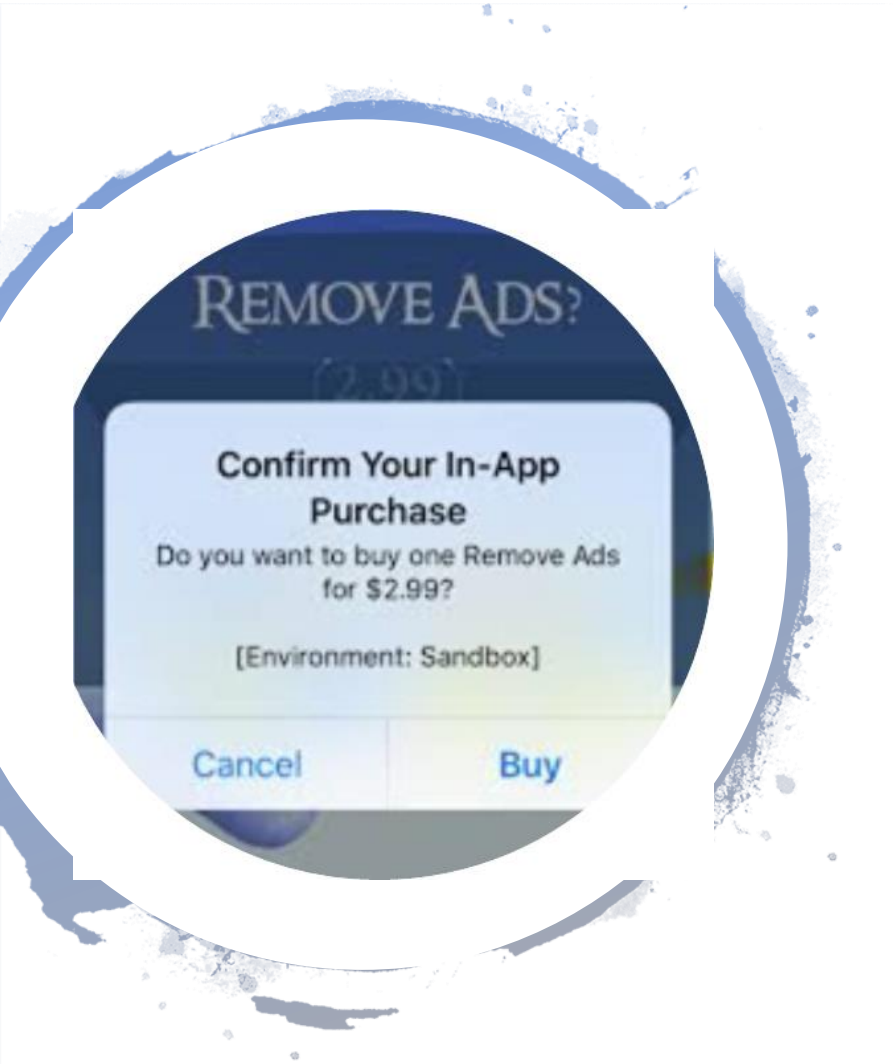
5, the actual PAN and customer information remains on the private credit card networks only, and is never transmitted to or from the POS. Thus the transactions are extremely secure.

Ways to use Apple Pay

- Contactless payment
- In-App Payments
- Paying on website or via Handoff
- Peer to Peer Payment transactions (Apple Cash)

Contactless Payments

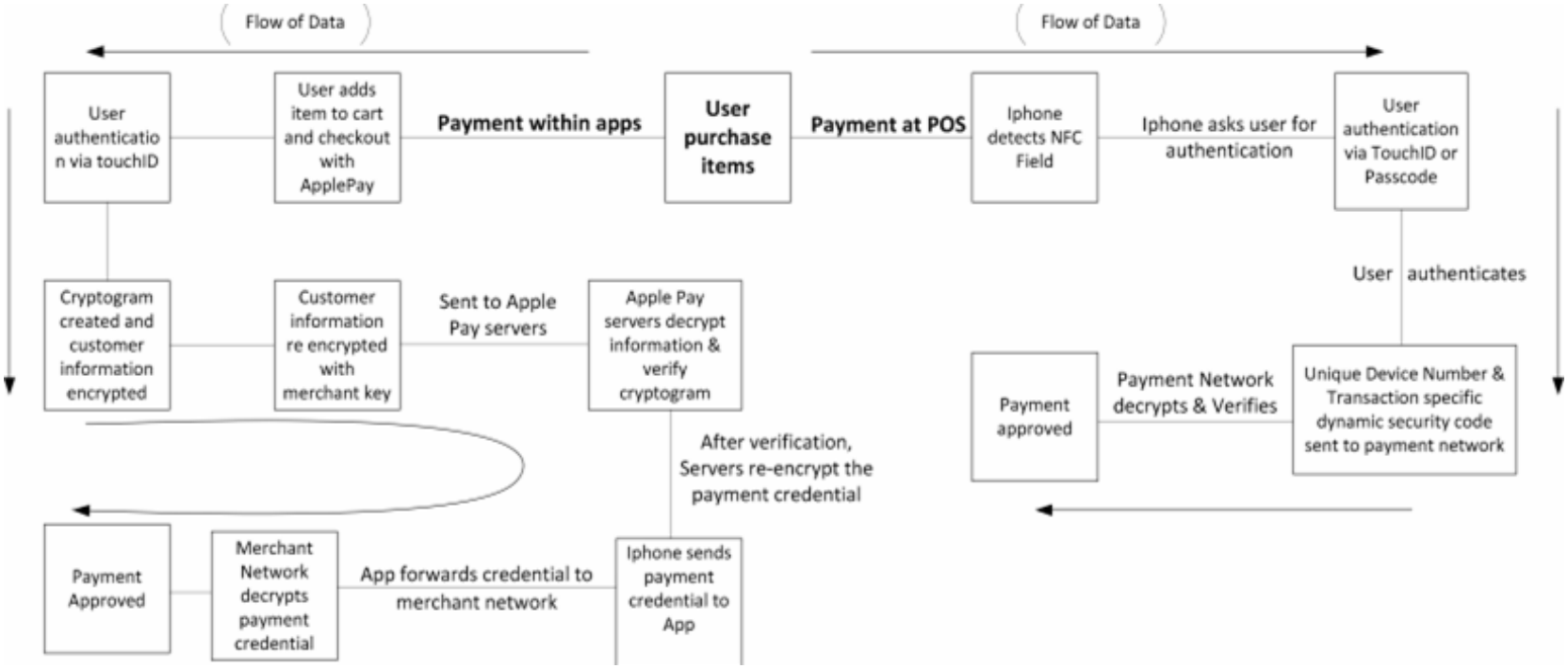
- Uses Near Field Communications technology
- Users authenticate using Face ID, Touch ID, or passcode
- Uses Device Account Number in place of Credit/Debit cards
- Device will never send merchants your real Credit/Debit card number



In-App Payments

- Developers will be provided a private key by Apple upon signing up.
- User authorizes payment to app by authenticating to their device
- A call is made to Apple servers to create cryptographic nonce.
- Secure Element generates payment credentials and sends to apple servers
- Apple Servers will verify payment and nonce is accurate
- Server will re-encrypt with a developer specific key

Payment via Apps / POS



Paying on the web or via Handoff

- Works only on devices with iOS10 or higher; macOS Sierra or later
- Websites MUST use HTTPS
- All participating merchants are required to register with Apple
- All participating merchants must obtain a secure session by Apple to send payment data
- Uses end to end IDS protocol during transit of data
- Uses same security as in-app purchases

Peer to Peer Transactions

- New Feature in iOS 11.2 and watchOS 4.2 and up
- User must have two-factor authentication enabled on their iCloud accounts
- Apple partnered with Green Dot Bank
- Apple created Apple Payments Inc. a different subsidiary than rest of Apple
  - Created to provide security to your purchases but also used to ensure regulatory/fraud concerns

Peer to Peer Transactions

- When user sends, adds or transfer money using Apple Cash, Apple Servers obtain a cryptographic none
- The nonce and transaction data are sent to Secure Element to create payment signature
- Signature gets verified by Apple Servers before sent to peer
  - Provides authentication and integrity of the payment
- If Apple suspects Cash account to have suspicious activity, user will be prompt to verify identity.

How do credit cards compare to Apple Pay

What do they do similarly?

- Contactless Payment
- Tokenization (EMV and Apple Pay)

What do they do differently?

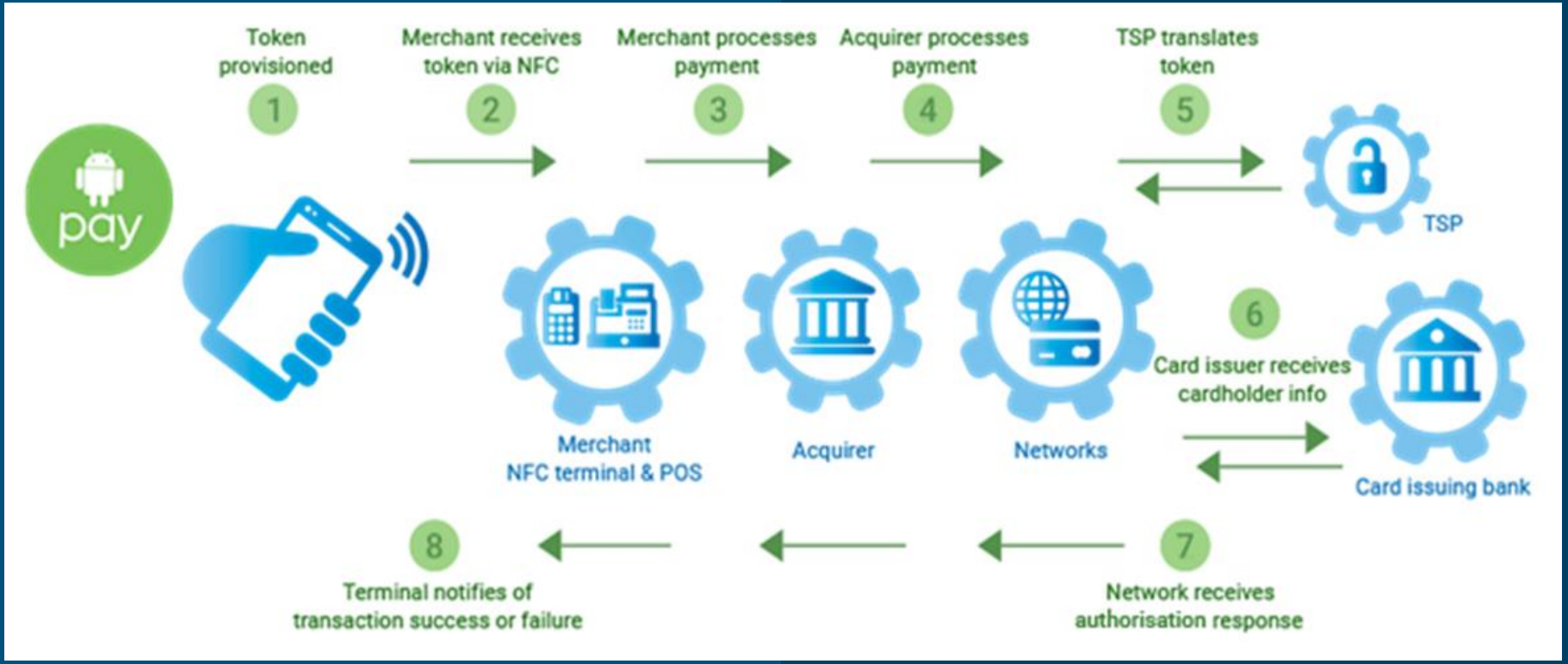
- Same security measures online
- Information storage
- Complete package

Conclusion: The pros for Apple Pay seem to outweigh conventional plastic cards, but both are headed to the same technological end.

Google Pay

- Formerly known as Android Pay (2015)/ Google Wallet (2011)
- Can be used peer-to peer, make payments through POS with NFC and allows for storing passes, and used online purchases
- Uses Near-Field Communication and two-factor authentication
- Uses Software based secure Element solutions (Host Card Emulation)

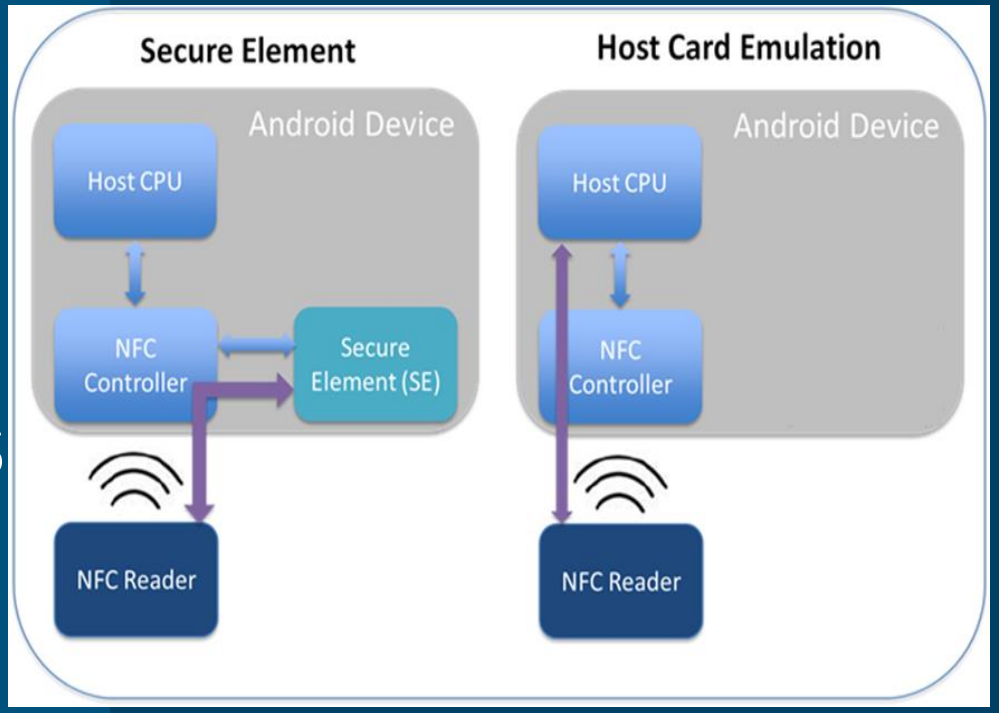
Google Pay In Store Process



Google Pay Secure Element-Host Card Based Emulation

- Cloud based
- Requires Communication with the POS
- Less Secure
- No permanent payment Keys
- BIND_NFC_SERVICE

Google Pay Payment Process



Google Pay Basic Requirements

•To protect the application from hacking it is required to follow basic requirements:

1.Implementation of one-time tokens with a limited lifetime;

2.The identification of the user;

3.Encryption of transmitted and stored data;

4.Ability of temporary activation of mobile payments;

5.Ability of deactivation mobile payments (via web service);

Differences in Apple Pay v Google Pay

- Verification Process
- How the information is stored
- Entities (Issuers, credit card, payment intermediaries, processor and
- merchants)
- Payment
- Locked Screen

Apple Pay Assurance Techniques

- Apple is built entirely upon its own ecosystem
- Uses a minimized system
- Use Secure Enclave which acts as a reference monitor
- Isolates the TCB
- Apple never transmits or stores the original CC number to the merchant, only the Device Account Number – information hiding
- No data is stored on Apple Servers (CC information or transaction information)
- Uses encryption when sending and storing data to protect against data hijack - MitM
- Use of Transaction-specific dynamic security code is random number created to verify each transaction.
- Allows users the ability to remove cards via other options - Protection on lost data

Google Pay Assurances

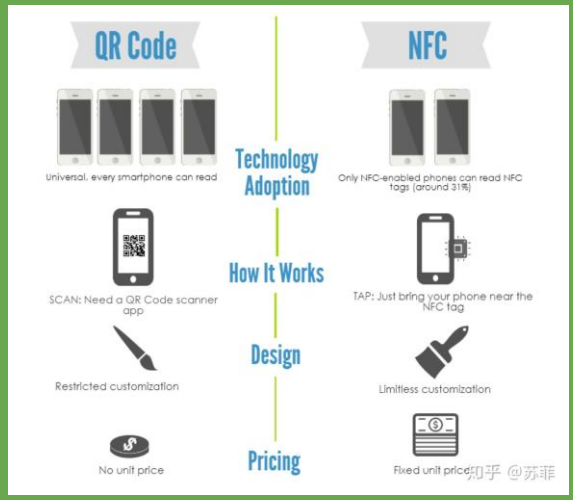
- Data Protection
- Virtual Account Numbers
- Screen Lock and small payment
- Google has its own secure servers that it uses to protect your information
- Will not work 100+ with input of your pin on the terminal
- Another device has to be within 4cm in order for an attack to even be possible

Subversion for Google Pay

- Bugs in Code
- Man in the Middle Attack
- Eavesdropping
- Theft
- Data Corruption and Manipulation

Ways to Improve Security

- Automatic deletion of all credit card information when no locked screen is present
- Unify/Standardize wallet technology to prevent issues with other companies
- Eliminate users from authenticating payment transactions using passcode
- Learn from QR Code payment
    - adding more noise in the payment transaction message to hiding the real confidential information
    - adding more checksum which can find and correct errors
    - double-authenticate



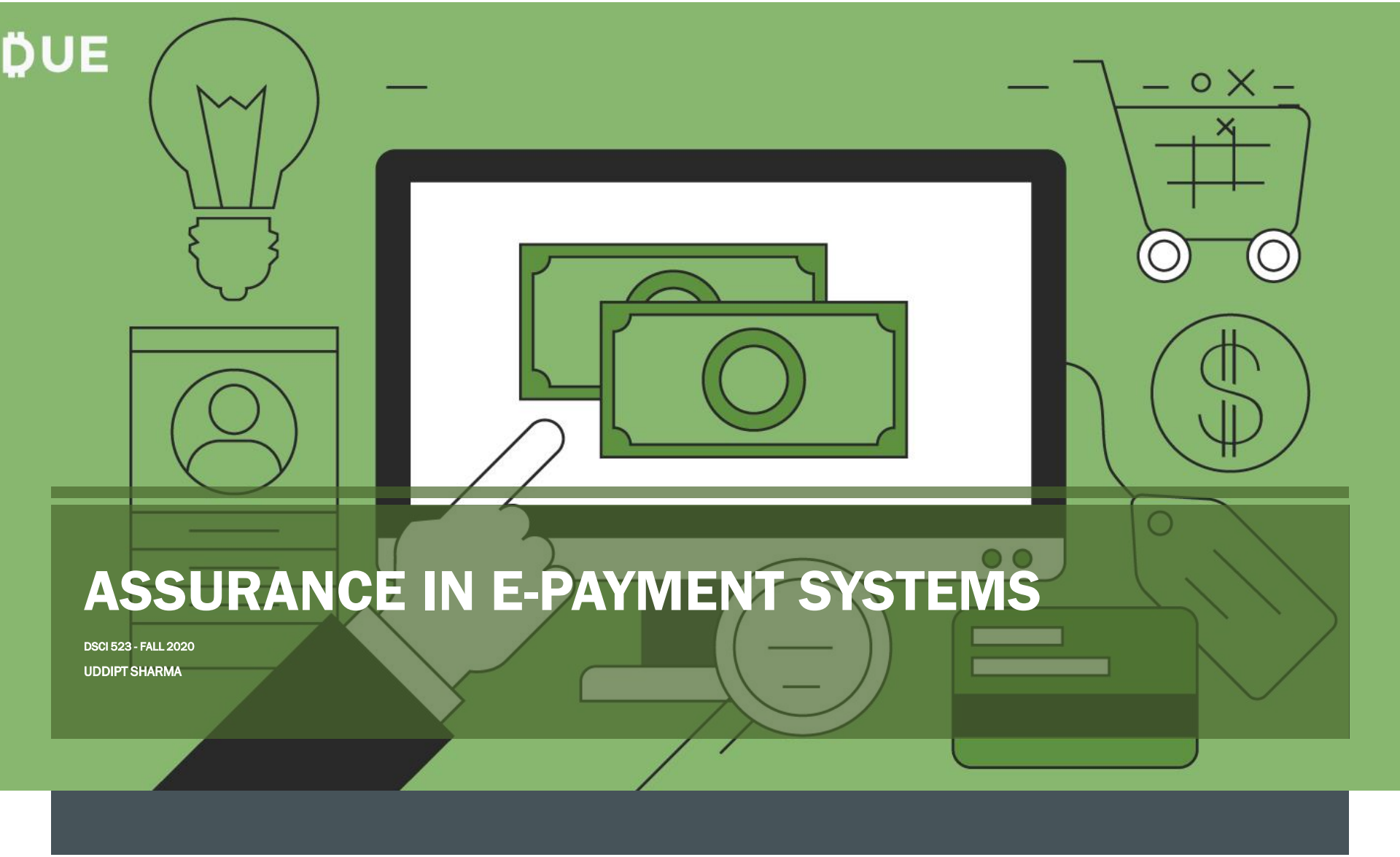
References

1. https://www.forbes.com/sites/thomasbrewster/2019/03/27/millions-are-being-lost-to-apple-pay-fraudwill-apple-card-come-to-the-rescue/#138d1b50622f
2. https://krebsonsecurity.com/2014/10/replay-attacks-spoof-chip-card-charges/
3. https://www.esecurityplanet.com/mobile-security/apple-pay-how-secure-is-it.html
4. https://www.ccc.de/en/updates/2013/ccc-breaks-apple-touchid
5. https://codeburst.io/how-does-apple-pay-actually-work-f52f7d9348b7
6. https://support.apple.com/en-us/HT203027
7. https://squareup.com/us/en/townsquare/what-does-tokenization-actually-mean
8. https://docs.radial.com/ptf/Content/Topics/payments/apple-integration.htm
9. https://atadistance.net/2020/04/01/ios-14-apple-pay-going-the-distance-with-ultra-wide-band-touchless-and-qr/
10. https://globalplatform.org/wp-content/uploads/2018/05/Introduction-to-Secure-Element-15May2018.pdf
11. https://support.apple.com/en-asia/guide/security/seccb53a35f0/1/web/1
12. https://globalplatform.org/wp-content/uploads/2018/06/GPC_Specification-2.2.1.pdf
13. https://www.thestreet.com/personal-finance/credit-cards/history-of-credit-cards
14. https://www.moneycrashers.com/emv-chip-credit-cards-technology-security/
15. https://www.pocket-lint.com/phones/news/apple/130870-what-is-apple-pay-how-does-it-work-and-which-banks-support-it
16. https://www.elitepersonalfinance.com/technologies-that-make-credit-card-secure/
17. https://www.forbes.com/advisor/credit-cards/contactless-credit-cards/
18. https://www.finder.com.au/credit-card-technologies#pb-nfc
19. https://www.digitalspy.com/tech/apple/a658189/apple-pay-can-be-used-anywhere-that-accepts-contactless-cards-not-just-retail-partners/
20. https://support.apple.com/guide/iphone/make-contactless-payments-iphbd4cf42b4/ios

References

21. https://support.apple.com/en-us/HT201469
22. https://www.macworld.com/article/3236545/apple-pay-cash-guide-what-it-is-how-it-works-and-what-it-costs.html
23. https://www.creditcards.com/credit-card-news/history-credit-card-magnetic-stripe-1273/
24. https://squareup.com/us/en/townsquare/why-are-chip-cards-more-secure-than-magnetic-stripe-cards
25. https://slate.com/human-interest/2015/08/credit-cards-passports-and-rfid-fraud-are-special-blocking-wallets-necessary.html
26. https://smartface.io/apple-pay-different-google-wallet-nfc-based-m-wallets-mobile-payment-systems/
27. https://usa.kaspersky.com/blog/secure-element/15411/
28. https://www.marketwatch.com/story/what-is-google-pay-and-is-it-safe-2019-05-24
29. http://cryptowiki.net/index.php?title=NFC_mobile_payments_security
30. https://www.apple.com/ca/business-docs/iOS_Security_Guide.pdf
31. https://www.esecurityplanet.com/mobile-security/apple-pay-how-secure-is-it.html
32. https://ieeexplore-ieee-org.libproxy1.usc.edu/document/7870214
33. https://www.forbes.com/sites/thomasbrewster/2016/03/01/apple-pay-fraud-test/?sh=6d414ffe46c6
34. https://developer.android.com/guide/topics/connectivity/nfc/hce#HCE

Thank you!!! Any Questions?!?!



ASSURANCE IN E-PAYMENT SYSTEMS

DSCI 523 - FALL 2020

UDDIPT SHARMA

WHAT ARE ELECTRONIC PAYMENT SYSTEMS?

- An e-payment system is a way of making transactions or paying for goods and services through an electronic medium, without the use of checks or cash. It's also called an electronic payment system or online payment system
- E-payment methods could be classified into two areas, credit payment systems and cash payment systems
- Credit Payment System- Includes Credit Card, E-wallet and Smart Card
- Cash Payment System- Includes Debit Card, E-check, Prepaid Cards and E-cash

SOUNDS GREAT, SO ARE THERE ANY DRAWBACKS?

- E-commerce **fraud** is growing at 30% per year. If you follow the security rules, there shouldn't be such problems, but when a merchant chooses a payment system which is not highly secure, there is a risk of sensitive data breach which may cause identity theft.
- **The lack of anonymity** — For most, it's not a problem at all, but you need to remember that some of your personal data is stored in the database of the payment system.
- **The need for internet access** — As you may guess, if the internet connection fails, it's impossible to complete a transaction, get to your online account, etc.

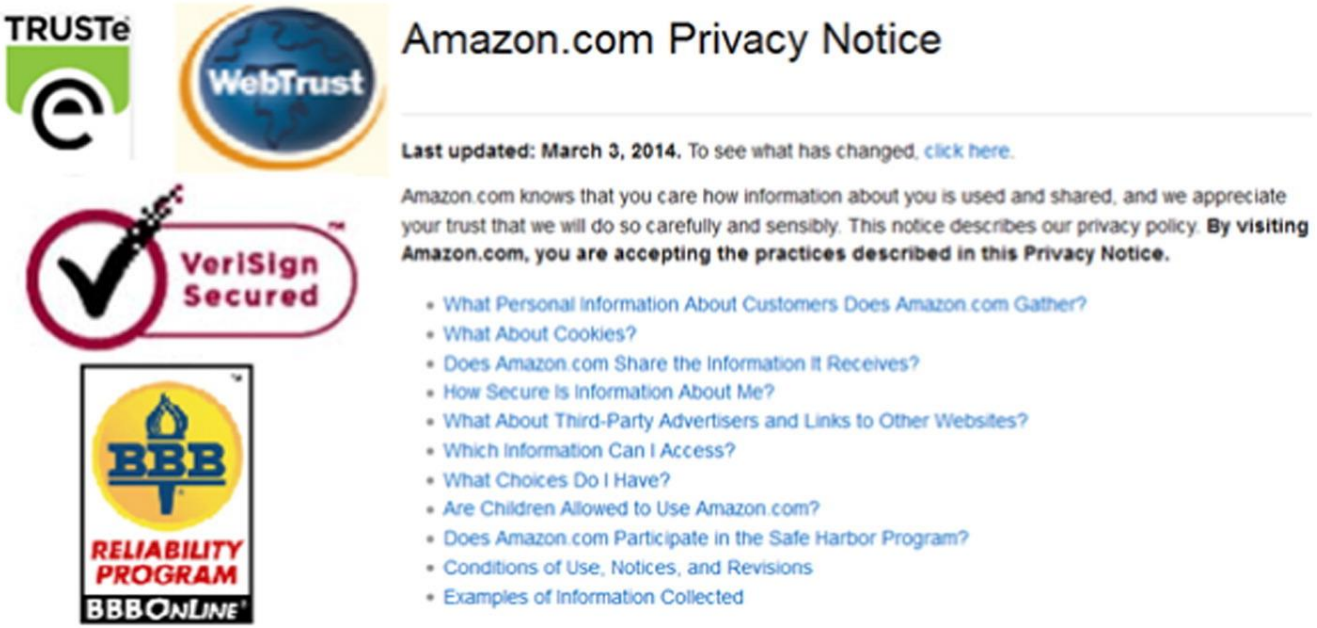
UNDERSTANDING THE PAYMENT GATEWAY

- Payment gateways facilitate communication and transmit transaction information between a payment portal (such as a website, mobile phone or interactive voice response service) and front-end processor of the acquiring bank
- It encrypts payment information, and then proceeds to authorizing payment and securely passing the information between sender and receiver
- When an order is confirmed by both the customer's as well as merchant's web server, a request from the application is sent to the payment gateway for payment processing
- After completion of the processing, gateway sends a response to the application in terms of success or failure
- The key concern of millions of people across the Globe lies around – "Is my transaction safe?", "Is my information secure?

ASSURANCE IN E-PAYMENT SYSTEM

- The uttermost importance to a business or Bank is the security and integrity of their payment processing system
- These system are often developed by third parties and thus needs a formal assurance for its security
- Security and Functional testing plays an important role in safeguarding the interest of both the parties
- Payment gateway testing requires continuous planning and diligence since it involves testing of different aspects such as security, web service connectivity, authorization, and data encryption
- End-to-end testing is to be performed with dedication and accuracy as the application is to be used for sensitive purposes
  - Functional Testing
  - Integration Testing
  - Security Testing
  - Performance Testing

SOME EXAMPLES OF ASSURANCE



REGULATIONS AND STANDARDS

- The Payment Card Industry Data Security Standard (PCI DSS) is a set of requirements intended to ensure that all companies that process, store, or transmit credit card information maintain a secure environment
- It was launched on September 7, 2006, to manage PCI security standards and improve account security throughout the transaction process
- While the PCI SSC has no legal authority to compel compliance, it is a requirement for any business that processes credit or debit card transactions
- PCI certification ensures the security of card data at your business through a set of requirements established by the PCI SSC
- PCI-compliant security provides a valuable asset that informs customers that your business is safe to transact with and provide assurance

OVERVIEW OF PCI SSC DATA SECURITY STANDARDS

- Use and Maintain Firewalls
- Proper Password Protections
- Protect Cardholder Data
- Encrypt Transmitted Data
- Use and Maintain Anti-Virus
- Restrict Data Access
- Unique IDs for Access
- Restrict Physical Access
- Create and Maintain Access Logs
- Scan and Test for Vulnerabilities
- Document Policies

PCI COMPLIANCE CHECKLIST 1/2

- Safeguard cardholder data by implementing and maintaining a firewall.
- Create custom passwords and other unique security measures rather than using the default setting from your vendor-supplied systems.
- Safeguard stored cardholder data.
- Encrypt cardholder data that is transmitted across open, public networks.
- Anti-virus software needs to implemented and actively updated.
- Create and sustain secure systems and applications.

PCI COMPLIANCE CHECKLIST 2/2

- Keep cardholder access limited by need-to-know.
- Users with digital access to cardholder data need unique identifiers.
- Physical access to cardholder data needs to be restricted.
- Network resources and cardholder data access needs to be logged and reported.
- Run frequent security systems and processes tests.
- Address information security throughout your business by creating a policy.

GUIDELINES FOR PROTECTING CARDHOLDER DATA ELEMENTS

		Data Element	Storage Permitted	Protection Required	If Allowed to Store- Must Render Unreadable
ACCOUNT DATA	Cardholder Data	Primary Account Number (PAN/SSN)	YES	Yes	YES
		Cardholder Name	YES	YES	NO
		Service Code	YES	YES	NO
		Expiration Date	YES	YES	NO
	Sensitive Authentication Data	Full Magnetic Stripe Data	NO	n/a	n/a
		CAV2/CVC2/CVV2/CID	NO	n/a	n/a
		PIN / PIN Block	NO	n/a	n/a

REQUIREMENTS OVERVIEW

Requirement	Sub-Requirements
Build and Maintain a Secure Network and Systems	1. Install and maintain a firewall configuration to protect cardholder data 2. Do not use vendor-supplied defaults for system passwords and other security parameters
Protect Cardholder Data	3. Protect stored cardholder data 4. Encrypt transmission of cardholder data across open, public networks
Maintain a Vulnerability Management Program	5. Protect all systems against malware and regularly update anti-virus software or programs 6. Develop and maintain secure systems and applications
Implement Strong Access Control Measures	7. Restrict access to cardholder data by business need to know 8. Identify and authenticate access to system components 9. Restrict physical access to cardholder data
Regularly Monitor and Test Networks	10. Track and monitor all access to network resources and cardholder data 11. Regularly test security systems and processes
Maintain an Information Security Policy	12. Maintain a policy that addresses information security for all personnel

OTHER STANDARDS

- Federal Financial Institutions Examination Council (FFIEC) has developed a Cybersecurity Assessment Tool (CAT) helps institutions identify their risk level and determine the maturity of their cybersecurity programs inducing payment system
- NIST Special Publication 800-63B, Special Publication (SP) 800-63, SP 800-63A, and SP 800-63C provide technical guidelines to agencies for the implementation of digital authentication
- ISO 20022 is an ISO standard for electronic data interchange between financial institutions.
- It describes a metadata repository containing descriptions of messages and business processes, and a maintenance process for the repository content
- It covers financial information transferred between financial institutions that includes payment transactions, securities trading and settlement information, credit and debit card transactions and other financial information

FFIEC

- Federal Financial Institutions Examination Council1 (FFIEC) developed the Cybersecurity Assessment Tool (Assessment), on behalf of its members, to help institutions identify their risks and determine their cybersecurity maturity.
- The content of the Assessment is consistent with the principles of the *FFIEC Information Technology Examination Handbook (IT Handbook)* and the National Institute of Standards and Technology (NIST) Cybersecurity Framework,2 as well as industry accepted cybersecurity practices
- To complete the Assessment, management first assesses the institution's inherent risk profile based on five categories:
  - Technologies and Connection Types
  - Delivery Channels
  - Online/Mobile Products and Technology Services
  - Organizational Characteristics
  - External Threats

FFIEC

- Management then evaluates the institution's Cybersecurity Maturity level for each of five domains:
  - Cyber Risk Management and Oversight
  - Threat Intelligence and Collaboration
  - Cybersecurity Controls
  - External Dependency Management
  - Cyber Incident Management and Resilience

FFIEC

Figure 1: Inherent Risk Profile Layout

Risk Levels

Activity, Service, or Product

Category: Technologies and Connection Types	Risk Levels				
	Least	Minimal	Moderate	Significant	Most
Total number of Internet service provider (ISP) connections (including branch connections)	No connections	Minimal complexity (1–20 connections)	Moderate complexity (21–100 connections)	Significant complexity (101–200 connections)	Substantial complexity (>200 connections)
Unsecured external connections, number of connections not users (e.g., file transfer protocol (FTP), Telnet, rlogin)	None	Few instances of unsecured connections (1–5)	Several instances of unsecured connections (6–10)	Significant instances of unsecured connections (11–25)	Substantial instances of unsecured connections (>25)
Wireless network access	No wireless access	Separate access points for guest wireless and corporate wireless	Guest and corporate wireless network access are logically separated; limited number of users and access points (1–250 users; 1–25 access points)	Wireless corporate network access; significant number of users and access points (251–1,000 users; 26–100 access points)	Wireless corporate network access; all employees have access; substantial number of access points (>1,000 users; >100 access points)

Figure 2: Inherent Risk Summary

	Risk Levels				
	Least	Minimal	Moderate	Significant	Most
Number of Statements Selected in Each Risk Level					
Based on Individual Risk Levels Selected, Assign an Inherent Risk Profile	**Least**	**Minimal**	**Moderate**	**Significant**	**Most**

WHAT IS ISO 20022?

- **ISO 20022 is an emerging global and open standard for payments messaging**
- It creates a common language and model for payments data across the globe. One that provides higher quality data than other standards which means higher quality payments for all. One that can adapt to new needs and new approaches. One that's not controlled by a single interest. One that can be used by anyone in the industry and implemented on any network
- ISO 20022 Financial services – Universal financial industry message scheme
- ISO 20022-1:2013 Part 1: Metamodel
- ISO 20022-2:2013 Part 2: UML profile
- ISO 20022-3:2013 Part 3: Modelling
- ISO 20022-4:2013 Part 4: XML Schema generation
- ISO 20022-5:2013 Part 5: Reverse engineering
- ISO 20022-6:2013 Part 6: Message transport characteristics
- ISO 20022-7:2013 Part 7: Registration
- ISO 20022-8:2013 Part 8: ASN.1 generation

BENEFITS OF 20022

- Richer, better structured and more granular data
- Quality data means quality payments
- Improved analytics, less manual intervention
- Supporting end-to-end automation
- Using modern technology for seamless integration
- Worldwide adoption

THANK YOU