3. Security assessment

In this section, we describe:

- Attacker models
- Transit security properties
- Software and hardware assessment

The main motivations for somebody to actually break our security model are as follows:

- Financial gain: Cloning tickets and selling them
- Personal gain: get free rides for friends / family
- Research recognition: academic publication on large conference

The following section gives an overview of the security implications of the Transit solution we proposed in this document. To give you a comprehensive assessment,we define various attacker models (e.g. an attacker with root privileges) and security properties relevant to transit use cases (e.g. it's impossible to copy or clone a transit card), and we assess how the 2 security solutions we propose (software-based and hardware-based) guard against all attacker models and security properties.

Attacker types

The types of attackers we had in mind when designing Google Pay Transit are as follows:

Attacker type	Description	
Root attacker	Assumed to have physical access to the Android phone running Google Pay and Google Account credentials. Additionally, such attacker is willing and able to root the device to obtain root OS access. This is the most powerful type of attacker we face.	
Malicious app attacker	The attacker tricks the victim into installing a malicious app and all permissions granted by the user at installation time. We assume the victim's phone isn't rooted.	
Physical access attacker	Physical access attackers commit fraud using the following methods: • They get access to a lost or stolen physical phone, but not its the authentication credentials • They trick riders into tapping their phone against a compromised NFC reader	

Transit security properties

The following is a set of meaningful security properties for transit use cases:

Property	Description	
Tamper-proof	An unauthorized party cannot successfully modify the content of the Transit Card (more specifically the Card payload). Only the TSH and transit readers can modify the transit card payload.	
Clone-proof	A Transit card or ticket cannot be copied and used on multiple devices (i.e.“device binding”), and the card key material cannot be taken off the device and used in a different device.	
Rollback-proof	The Transit card content - specifically the card payload - cannot be restored to a previous state. Example: An attacker should not be able to use a stored-value card with $10 to pay for a $4 trip and then rollback the new stored value from $6 back to the original $10).	

Transit security assessment

The following is our assessment of how the 2 solutions we’ve implemented into Google Pay Transit work for each attacker model: Software solution and hardware solution.

Software solution

Google Pay Transit’s software guards against the following 4 primary types of fraud attacks:

- **Malicious app attacks** cannot break any of the security properties because the Android Framework Security (specifically Application Sandboxing) protects Google Pay’s storage from unauthorized access by another application (the assumption here is that the user / victim has not rooted their device).

- **Attacks with physical access to the device:** Attackers don’t have memory access (the device is locked) and the Card Key Material is never transferred over NFC (3-pass authentication requires to prove knowledge of key material, not to transfer it over).

- **Attackers attempting to perform relay attacks** (i.e. placing a reader near the victim’s device and relaying information back and forth to a remote reader): It’s important to note that relay attacks are already a threat to transit systems today and, as such, countermeasures for this kind of thread (e.g. reader time out approaches) are typically

already in place. While it is true that not all readers might have built-in capabilities to deal with relay attacks, one could make the argument that the mobile approach described above is not adding a new attack vector by emulating transit cards on a mobile device (as we intend to do). If anything, relay attacks are significantly more difficult to achieve on mobile devices than physical cards as mobile devices require (1) the victim's device to be active (i.e. screen on) and (2) the attacker's fake terminal to be in close proximity to the victim's device (NFC ~= 4-10cms -- RFID up to 50ms).

- **Root attacks:** These are complex attacks to prevent and stop. The following is how the well our software solution does for each security property:

Transit security properties

Tamper-proof: Prevented

This property is typically achieved by signing Transit Card's content, more specifically Transit Payload content. In practice, the Transit Hub signs the card content with a private key and at tap time the Transit Reader can verify the signature with the corresponding public key. The same holds the other way around, the Transit Reader resigns the Transit Card content using a private key.

Clone-proof: Possible, but unlikely

Cloning a card requires an attacker to successfully perform the following:

- Get valid card material: Card material is necessary to authenticate with a transit terminal and convey transit information (eg, store value, expiration date).
- Emulate the transit protocol: A mechanism to communicate with a transit reader and convey transit card info. In other words, a mechanism to emulate the actual transit protocol.

Rollback-proof: Possible

This property cannot be achieved with any solution that does not provide (a) secure storage and (b) secure execution of business logic. Given that the Android framework currently does not provide a way to run our transit SDK on a secure environment, a root attacker would be able to overwrite Transit bundles to their previous state effectively breaking the rollback-proof property. Later in this document, we will discuss mechanisms to identify rollbacks and stop the Transit Cards from working in such cases (see fraud detection and fraud circumvention discussion below).

Get valid card material

To get valid card material, an attacker needs to do one of the following:

- **Get a valid transit card on their Android device** and then extract key material from it

  How it works: For the attacker to extract card material from a mobile device, they must do the following:
  - Bypass Google's attestation infrastructure checks, which we require for card material provision and refreshment
  - Bypass TSH or PTO risk checks, which are required for card material provision and refreshment
  - Bypass Android Sandboxing security, which are required in order to actually extract the card material from Google Pay's sandbox environment

It's also worth noting the steps above need to be reproducible on an ongoing basis (eg, daily) to persistently take advantage of the exploitation (as transit keys are short-lived).

- **Get card material from the TSH or Google** backends or during their propagation step from backends to a mobile device.

  How it works: Getting key material this way is extremely difficult because it requires breaking the security of Google backend, which is built on top of state-of-the-art security technology. Additionally,Google runs a security audit with each partner, so TSH infrastructure will be verified to meet the highest security standards in terms of data storage, access, retentions, etc.

  Note: The security measures to prevent this type of attack are the same as Google Pay's open loop payments solution.

Emulate the transit protocol

To use cloned card material on multiple devices (in case the attacker is in possession of card material / transit bundles), the attacker needs a mechanism to emulate the corresponding transit protocol to communicate with transit readers. This requires a significant amount of work. For example, it typically takes our transit partners several months to produce a Transit SDK able to emulate their protocol). The fact that transit protocol spec and transit reader logic are not publicly known significantly increases the required level of effort build an app emulating a given transit protocol.

Note: Using a cloned card in Google Pay won't work unless the recipient device is also cloned (in order to write into Google Pay's persistent and resilient storage. Additionally, any backend calls (e.g. Refreshing key material) will automatically fail because the backend logic in Google servers will identify card material linked to the wrong account & device.

Scaling this type of attack is very hard. In terms of app distribution, these apps violate the IP copyright of the transit protocol and removed from any Android distribution platform,

Learn more about stopping fraud below.

Software solution security summary

Given the complexity of extracting card material on a regular basis,building up a transit protocol emulator, and having the ability to widely deploy an application to actually use the extracted card material, we consider the attack to be possible but very unlikely.

The following table summarizes attacker capabilities for the various flavours of attackers and security properties:

Attacker type	Clone-proof	Tamper-proof	Rollback-proof
Root attacker	**Possible** Effort: Hard	**Prevented** by signature and verification of card material	**Possible** Effort: Low
Malicious app	**Prevented**		

attacker	by Android Framework Security
Physical access attacker	**Prevented** The attacker doesn't have memory access and keys never get transferred over NFC

Hardware solution

Most of the observations made for our software solution remain true when leveraging hardware-backed security except when a **rooted attacker** with and Clone-proof property.

Clone-proof - Prevented

To clone and use a transit card, an attacker would still need to obtain valid card material and emulate the Transit Protocol. The main difference in this case is that extracting card material (more precisely card key material) becomes significantly more difficult in this case. Given that key material is never exposed in plain memory (keys are imported securely and used only within hardware / secure-zone), extracting keys requires breaking Android Keystore hardware-backed solution. Therefore, we consider that unlikely.

Security analysis for hardware solution

Attacker type	Clone-proof	Tamper-proof	Rollback-proof
Root attacker	**Prevented** By hardware-backed TEE	**Prevented** By signature and verification of card material	**Possible** Effort: Low
Malicious app attacker	**Prevented** By Android framework security		
Physical access attacker	**Prevented** The attacker doesn't have memory access and keys never get transferred over NFC		

4. Fraud analysis

In this section we discuss:

- Fraud overview
- Fraud attack types
- Required fraud tactics

Fraud overview

Public transit fraudis common and relatively easy to achieve. PTOs have varying concerns around fraud, but commercial savings and user convenience usually are more important to them than fraud prevention.

Transit fraud categories

Public transit fraud can be seen as the combination of 2 fraud techniques:

- **Fare evasion**: (i.e. Riders using transportation services without paying for them, such as jumping turnstiles, sharing a monthly pass among multiple people, etc. Also called “fare dodging” or “toll fraud”.
- **Fare falsification:** Making a fake ticket (i.e.)

Ways to prevent fraud

To fight fraud, transit agencies can use many strategies like the following:

- Ticket turnstiles
- Sophisticated technology (e.g. Using smartcards instead of paper tickets or moving away from smartcard technology known to be faulty like Mifare Classic Hack)
- Card ghost copies coupled with card blacklist mechanisms
- Panic bars
- Closed-circuit television monitoring
- Penalty fares
- Civil and criminal penalties
- Ticket inspectors

Learn more about fraud countermeasures.

Fraud rate and tolerance

Despite the availability of many ways to stop it, the rate of fraud in transit is as much as 10-15%higher than in other sectors such as credit card payment.For example, UK transit operators report £210 million yearly losses, and fraud costs German operators Eur 350.

Transit agencies often cite user convenience and high costs to skip adding additional fraud countermeasures

Fraud liability

Fraud liability usually is the responsibility of transit agencies, and, sometimes system integrators (SIs),too. Google will assist transit agencies to get transit cards / tickets to riders to convey them to transit readers via NFC technology, but under no circumstances will Google be liable for transit fraud.

Fraud attack types

HCE solutions to let riders buy and use transit tickets or cards are vulnerable to new types of frauds. Since fraudsters could only try and falsify fares with Google Pay Transit, our design of Google Pay Transit concentrated on the following types of transit fraud attacks:

- Tampering and rollbacks
- Card cloning

Card tampering and rollbacks

What is it: An attacker can tamper with the content of a transit card (e.g. Adding $10 to the card's balance) or roll back cards to previous states to get fraudulent transit rides.

Keep in mind:

- As discussed before (see security assessment Section), these types of attacks require riders to root their phone, making a potential vulnerability very difficult to scale. Many riders don't know how to root their phone, and even if they do, there are well known disadvantages to doing it.
- It's unlikely fraudsters will risk causing a vulnerability on phone since you can prevent, detect and stop fraud (as we will discuss below) and the block fraudsters from using fake cards.
- Fraudsters can be identified by their digital credentials and potentially prosecuted (see example) or held financially responsible by charging the credit card used for the initial card purchase.
- We expect few cases of tampering and rolling back for the following reasons:
    - The amount of effort needed to tamper / rollback tickets (e.g. rooting devices) is significant
- Tampering and rollbacks give attackers limited benefit because PTOs can invalidate fraudulent tickets after a few hours.
- Attackers risk potential legal and financial prosecution thanks to the digital credentials attackers need to commit fraud Fare evasion is far easier.

Card cloning

What is it: An attacker can clone a transit card and use it or let family and friends get free transit rides, or can sell it.

Keep in mind:

- To clone a card, attackers create an application that emulates the transit protocol to communicate with the transit system reader and convey the cloned credentials. This is a significant technological challenge.
- Though clone card fraudsters do not have to root their phones like they do for tampering and rollbacks, they must install the fraudulent application mentioned above. Google provides a mechanism to remove fraudulent apps from Google Play Store, and transit-technology IP owners also can remove these types of apps from non-Google properties.
- Just as for tampering and rollback, PTO's can quickly blacklist a cloned card and its copies. We can identify the attacker's account and payment credentials, and take legal and financial actions them.

- Here's what an attacker has to do to create a stable fraud schema based on card cloning:
    - To keep refreshing key materials, the attacker must continuously clone card material, which is extremely difficult. They must build an Android mobile app that emulates the transit protocol distribute the apponline. The app would have to be downloaded by users on "official channels" (i.e. Not through the Google Play Store)
    - Set up a schema to sell cloned cards.

  PTOs can help prevent cloned-card fraud by requiring rider identity and verification (ID&V) if your system allows for that at the point-of-purchase.

Required fraud prevention measures

To launch a Google Pay Transit integration, we require partners to use the following fraud prevention tactics:

- **Use risk scores to prevent fraud**
  When a rider tries to use a ticket, we provide risk signals to PTOs to help keep t transit bundles from being sent to phones, including:
    - Device signals such as stable hardware identifier, IP address, geo-location info, etc
    - Account signals, such as the age of the Google account and the wallet account, the number of payment instruments in it, etc. A combined device risk and account risk rating between from 1 (riskies) to 5(least risky)

  The transit server hub will then make the decision (optionally in collaboration with PTOs) of whether to create a transit bundle and deliver it to the device or refuse the operation.

  We also sent device and account risk signals to PTOs when riders try to purchase or add value to a a ticket / card[1].

- **Use transit reader logs to detect fraud**
  To spot tampering, rollbacks, or clone attacks, the TSH or PTO must continuously analyze both transit reader logs and the risk signals we send.

  Usually, transit readers logs include the card's unique identifier (UID), station location, and timestamp of each transaction. Simple geospatial and time analysis would allow to detect fraud. For example, logs will show the same card being used many times within a short period of time, or the same card being used in distant stations in a short period of time. We expect either the TSH or PTO keep a copy of each card with the expected value on each card. Then, you can use transit logs to spot card tampering by matching the card's expected value with the actual one presented at transaction time.

- **Blacklist cards and tickets and remove apps to stop fraud you detect**
  When you detect fraud, stop attacks quickly by doing the following:

[1] We're considering sharing basic risk info when we refresh transit bundles (e.g. getting new key material).

    - **Blacklist cards:** After you detect or identify a fraudulent card, you should blacklist it by propagating the UID (or ticket ID if only one ticket should be blacklisted) to transit readers. The TSH can also contact Google Pay Transit to delete or suspend the card.
    - **Remove apps:** Ifan Android app is embedding a cloned transit bundle and emulating a transit protocol for free rides, PTOs can contact us to start the app removal process from Google Play Store (if applicable).

- **Retaliate against fraudsters**
  When you detect fraud, work to identify the credit card used to purchase the initial ticket and use legal means to prosecute attackers and get financial compensation. **Note:** Google Pay Transit doesn't take part in these processes outside providing basic information to PTOs.

Appendix

How short-lived keys work

Transit Hubs can generate short-lived keys. Google Pay will keep refreshing those keys as they approach their expiration time (based on the *requestRefreshTimestamp* value in the Transit bundle).

At transaction time, transit readers will validate the expiration time of transit diversified keys at the beginning of the transaction and stop it in case the key is expired.

The details of how the expiration time is communicated to transit readers vary depending on the actual transit technology (eg Mifare, Calypso) and are out of scope for this document

Transit card source-of-truth

Typical transit deployments are offline, in that the actual authorization takes place without real-time access to a backend holding transit balance (eg, value or tickets). Transit readers

often modify the actual value of the card during the transaction (eg, reducing value on the card, setting a single-use-ticket as used). Similarly, the mobile device might not have online access at transaction time.

Therefore, the card information hosted by the mobile device (more specifically, as part of the Transit bundle Payload) represents the source of truth of transit card balance.

The situation changes when looking at the life-cycle tickets within a transit card. Given that all tickets are issued via the Transit Hub (while the mobile device is online), the hub has knowledge of the number of tickets on each transit card at all times.

Glossary

Term	Acronym (if applicable)	Definition
Active / activation		The process of associating a ticket saved to a Google Account to a specific phone. After a ticket is activated, its information is transferred onto that specific phone only and the ticket can then be used on that phone. A ticket can only be activated on one device at a time.
Google Pay		The Google Pay mobile app for Android devices..
Android device		Any NFC-compatible Android device running Lollipop or above.
Application identifier	AID	An unique code used to address an application on a mobile device.
Carnet of tickets		A collection of identical tickets, usually single-ride tickets, where many of these tickets are purchased as a single ticket (e.g. 10 single rides). When a rider uses a carnet of tickets, the amount of available trips decreases.
Device Encryption Key		This Google-generated key is scoped to both a specific user (based on their Google account) and device. Google Pay retrieves it at device boot

		provided the device hasn't been compromised (per SafetyNet APIs) and it's only kept in transient memory (RAM).
Europay, Mastercard, and Visa	EMV	The technical standard for smart payment cards
Form of payment	FoP	Debit or credit card, cash, or card on file
Google server		Google backend server implementing several transit specific APIs
Host card emulation	HCE	Virtual representation of transit card-using software
Identification and verification	ID&V	The process to verify a rider's identity that's required to buy or use some tickets (e.g. senior, student)
ISO transport protocol specification	ISO	International standard applicable to equipment supporting the transport service
Keystore		Android Keystore system providing APIs to import keys and encrypt / decrypt information. Depending on the Android device running Keystore, keys might be hardware-backed (either by TEEs or SEs).
Keyhole markup language	KML	KML is a file format used to display geographic data in an Earth browser such as Google Earth Learn more about KMLs.
Near field communication	NFC	Contactless communication protocol
Pay with Google	PwG	Google's payment platform that lets riders make a purchase online or in a mobile app using card information securely stored in their personal Google Accounts
Public transit operator	PTO	Government or private authority / agency responsible for shared passenger transit services
Save a ticket		The process to relate a ticket to a rider's Google Account, typically right after a ticket purchase
Save to Google Pay		An application programming interface (API) that enables transit agencies to let their riders save tickets to Google Pay and use these tickets to pay with a tap of their phones on a terminal
Season pass		Also known as a "pass," this is a time based ticket that permits unlimited rides during a given period. These are typically day passes, week passes, monthly passes or yearly passes
System integrator	SI	Third-party responsible for designing and deploying a ticketing solution for PTOs

Terms of Service	ToS	The requirements to use a service.
Ticket vending machine	TVM	Machine that dispenses transit cards / tickets
Transit app		A transit mobile app (i.e. a transit agency app or a transit ticket retail app) and a mechanism to allow transit apps (as well as retail apps or web services) to "push" transit cards and products to Google Pay
Transit app backend		The backend used by the transit agency app / site dealing with ticket purchasing that typically integrates with a payment processor to process payments
Transit bundle		Transit cards are encoded as JSON objects and transferred across services (Transit Hub -> Google Server -> Google Pay).
Transit back office		Provides standard transit capabilities such as fraud analysis, physical card issuance, payment reconciliation, etc. and is typically maintained by SIs
Transit cards		At a high level, transit cards can be seen as the combination of 2 elements: ● **Card payload** - All metadata encoded in transit cards (eg, cardID, ticket info, transaction info, etc) and necessary for transit terminals to process transit transactions. The payload is processed by the Transit SDK and might be obfuscated to Google Pay. ● **Card key material** - The combination of one or more Card DKs, such keys may be short-lived in some cases. Transit card material (payload, keys, and more) are generated by the Transit Server Hub and encoded as Transit bundles. more details about Transit bundles can be found in the Transit Applet Spec.
Transit keys		Various cryptographic keys are in play to secure transit deployments: ● **Transit master keys**: Symmetric keys guarded under maximum security. Typically these keys are only available in HSM and SAM modules and are never deployed to users / cards. Instead, use-only diversified keys are typically used for transit cards to authenticate with readers.

		• **Card-diversified key**: Card-specific key derived from the Transit Master key and card-specific information (eg, UID, expiration time, etc). • **Short-lived diversified keys**: These are card diversified keys encoding an expiration time (e.g. 24 hours). The Transit reader will validate such expiration time during a transit transaction and reject the transaction if validation fails. Google Pay app is in charge to refresh keys prior its expiration time.
Transit protocol		A transit protocol refers to the technology used to (1) encode transit information in a smartcard and (2) communicate information across transit cards and readers (including authentication). Examples of transit protocols include Mifare, DESFire, FeliCa, Calypso, and Cipurse.
Transit sensitive data	TSD	Data flowing across Google's properties (e.g. Google servers and Google Pay). An example of TSD is card key material. The specific definition of TSD can vary among PTOs.
Transit server hub	TSH	Similar to a token service provider (TSP) in the payments world, the TSH is a cloud service providing backend APIs to create Transit Cards and functions as a "hub" to perform the following: • Assess as a risk engine • Manage life-cycle of tickets • Route ticket data to Google Pay for 1st-party, in-app purchases • Store mirror copies of mobile device for risk purposes and customer support
Transit SDK		This is a java library implementing the underlying transit protocol. The SDK is typically provided by the party running the TSH and needs to be compatible with transit readers and the corresponding Transit bundle.
Transit terminal		NFC-compliant transit terminal implementing a Transit Protocol.
Transit ticket		A ticket can be used for media-enabling transactions with a transit terminal, allowing a rider to ride a public transit system. Ticket types include "single," "return," "carnets," "zone-based," "value-on-card," and "pass."

Trusted execution environment or secure element	TEE / SE	Google Pay may use the a TEE or SE to store either DK or wrapping keys using keystore APIs
Value-on-card or Pay-as-you-go	VOC / PAYG	A card that has a stored balance like a digital purse. Each time the card is used, the balance is decreased by the price of the fare (e.g. Clipper card with balance)
User experience design	UX	Represents user's interaction, experience
User interface	UI	A mobile system that riders use
Unique identifier	UID	Refers to a smartcard unique card identifier or serial number.
Zone-based ticket		A ticket that lets riders travel within specific zones that are defined as a geographical region. These zones usually form rings around a city's center. As you move away from the city's center, the zone of each ring incrementally increases (e.g. Caltrain-used zones for the train tickets)