TRR
JOURNAL OF THE TRANSPORTATION RESEARCH BOARD

Research Article

Transportation Research Record
1–9

DOI: 10.1177/0361198120945982
journals.sagepub.com/home/trr
SAGE

Cybersecurity Vulnerabilities in Mobile Fare Payment Applications: A Case Study

Kevin Dennis[1], Maxat Alibayev[1], Sean J. Barbeau[2], and Jay Ligatti[1]

Abstract

Mobile fare payment applications are becoming increasingly common in the public transportation industry as a convenience for customers and as part of an effort to reduce fare management costs and improve operations for agencies. However, there is relatively little literature on vulnerabilities and liabilities in mobile fare payment applications. Furthermore, few public agencies or supporting vendors have policies or established processes in place to receive vulnerability reports or patch vulnerabilities discovered in their technologies. Given the rapidly increasing number of data breaches in general industry IT systems, as well as that mobile fare payment apps are a nexus between customer and agency financial information, the security of these mobile applications deserves further scrutiny. This paper presents a vulnerability discovered in a mobile fare payment application deployed at a transit agency in Florida that, because of the system architecture, may have affected customers in as many as 40 cities across the United States, an estimated 1,554,000 users. Lessons learned from the vulnerability disclosure process followed by the research team as well as recommendations for public agencies seeking to improve the security of these types of applications are also discussed.

Cybersecurity is a significant concern across all industries. Many high-profile attacks and breaches have been reported such as the Equifax data breach (*1*) as well as the ransomware attacks in Atlanta (*2*) and Baltimore (*3*). The ransomware attack in Atlanta was estimated to have cost around $2.6 million, and costs from the Equifax breach have reached over $1.4 billion (*4*).

Transportation technologies, as they continue to rapidly expand from individual nodes to large, interconnected networks of technologies, may be susceptible to many unique vulnerabilities that have yet to be discovered or patched. In addition, these technologies make an alluring target for adversaries seeking to cause wide-reaching and costly damage. Transportation agencies have already been affected by breaches. On February 22, 2018, the Colorado Department of Transportation shut down 2,000 employee computers after the SamSam ransomware virus infected their systems and stole files (*5*).

Mobile fare payment applications are becoming increasingly commonplace in the public transportation industry as a convenience for customers and as part of an effort to reduce fare management costs and improve operations for agencies. However, there is relatively little literature on vulnerabilities and liabilities in mobile fare payment applications. Furthermore, few public agencies or their supporting vendors have policies in place to receive vulnerability reports or patch vulnerabilities discovered in their technologies, including mobile fare payment applications. Given the rapidly increasing number of data breaches in general industry IT systems, as well as that mobile fare payment apps are a nexus between customer and agency financial information which are, therefore, critical to operations, the security of these mobile applications deserves further scrutiny.

This paper presents a vulnerability discovered in a mobile fare payment application deployed at a transit agency in Florida that, because of the system architecture, may have affected customers in as many as 40 cities across the United States. The paper begins with a brief background of mobile fare payment applications, including potential vulnerabilities, and a review of related work. The background is followed by an analysis of the vulnerability discovered in the mobile fare payment

[1]Computer Science and Engineering Department, University of South Florida, Tampa, FL
[2]Center for Urban Transportation Research, University of South Florida, Tampa, FL

Corresponding Author:
Sean J. Barbeau, barbeau@usf.edu

application. The paper concludes with recommendations and suggested policies for public agencies, including a review of existing federal and Florida state security policies.

Mobile Fare Payment Applications and Their Vulnerabilities

This section introduces mobile fare payment applications, describing the technologies involved and the benefits achieved by mobile fare payment.

Mobile fare payment applications are a form of contactless electronic ticketing that enables passengers to purchase a ticket and validate the purchase using their mobile device. Mobile fare payment is typically added as an additional, more convenient fare payment option, rather than replacing existing options entirely (*6*, p. 36). Mobile fare payment may reduce production and cash-handling costs (*7*, p. 6). Passenger Transport reported that First Bus's mobile fare payment system reduced boarding times by up to 75% (*8*).

Typical Implementations

Mobile fare payment is typically implemented using one, or a combination, of the following technologies: visual validation; Quick Response (QR) codes; Near-Field Communications (NFC); or Bluetooth Low Energy (BLE).

In a visual validation scheme, the user is provided with an image on their mobile device that is shown to agency staff to verify their ticket purchase. Visual validation requires no additional equipment on the vehicle or station and instead uses the data connection of the user's device for any necessary communication with a remote server (*6*).

QR codes (*9*) require QR scanners to be available on the vehicle, or at the station, and require a method for communicating with a remote server, such as Wi-Fi or cellular, for verification (*6*). In a QR code system, a QR code is provided to the user on their mobile device when they purchase of their ticket. This ticket is then held under a QR scanner for verification (*7*).

NFC provides short-range radio communication between the user's mobile device and the NFC device (*10*). Unlike visual validation or QR codes, NFC requires the mobile device to support it with specific hardware. NFC is a contactless payment form in which users bring their devices into proximity to the sensors, similar to RFID.

BLE is a low-energy implementation of Bluetooth (*11*). BLE provides a larger communication range than NFC or RFID. This increased range allows mobile fare payment to be conducted in a Be-In/Be-Out (BIBO) style by detecting the device when the user passes through the gates or boards the vehicle (*6*), avoiding the constraint of passing the device by a sensor as in NFC.

Related Work

Mobile fare payments are a relatively new technology, and little has been published on their security concerns. However, related studies on other forms of payment technologies may be relevant. Mobile fare payment shares the privacy concerns present in other forms of electronic ticketing described in Section 3.2.1. Kieseberg et al. (*12*) analyze the attack vectors generally present in QR codes. These include command injection, phishing, and other social engineering attacks. Private user information could also be stolen through Trojan applications that mislead users by masquerading as other applications—one such occurrence was discovered with the Uber app (*13*).

Wang et al. (*14*) describe how a Sybil attack can be performed on crowdsourced applications. In a Sybil attack, the attackers create many fake users that report falsified traffic incidents, causing the application to reroute users and thus create traffic jams. Uber and Lyft drivers have reportedly executed a similar type of attack to artificially trigger surge pricing, increasing the amount of money they make for each ride (*15*).

Case Study: MyJTA Mobile Fare Payment App

In this study, the MyJTA (*16*) Android application for the Jacksonville, Florida area was analyzed for vulnerabilities. The analysis focused on the application programming interface (API) communication between the application and back-end server, and the ticket activation process.

Tools

To perform the analysis, the following tools were used: Fiddler (*17*); Android Studio (*18*); and apktool (*19*).

Fiddler is a web debugging proxy from Telerik. The Fiddler application is used to establish a proxy to which the Android device can connect to capture the API requests made by the application. The application has a variety of features that enhance the analysis process, including HTTPS certificate generation to decrypt HTTPS traffic from the device, request replays (to repeat previous communication), and request editing (to modify the content of the communication).

Apktool is a reverse engineering tool used to analyze and debug closed-source, compiled Android applications that are in the Android Package (APK) format. Apktool disassembles the resources inside an APK file to a

```
{
    "status": 200,
    "data": [
        {
            "tickets_purchase_ptticket_id": "3040847",
            "parkerid": "7915576",
            "ticket_usage_id": "2879942",
            "entrytime": "2018-09-12 00:58:00",
            "exittime": "2018-09-13 02:00:00",
            "os": "android",
            "osversion": "4.4.2",
            "appversion": "6.2.35",
            "manufacturer": "LGE",
            "model": "Nexus 5",
            "app_uuid": null,
            "device_identifier": "00000000-555a-4ce5-0000-000077409 3f6",
            "phonenumber": "239-XXXX",  //Removed by researchers
            "tickettype": "1-Day $4 (Transit \/ Bus)",
            "fareid": "45",
            "zonename": "1-Day $4 (Transit \/ Bus)",
            "parentzonename": "1 Day Pass",
            "farename": "1 Day Pass",
            "backgroundcolor": "FFA100",
            "letter": "1D",
            "textcolor": "FFFFFF",
            "billingtype": "Credit\/Debit Card",
            "billingtypeid": "1",
            "merch_transaction_id": "3739547797",
            "transaction_date": null,
            "purchase_date": "09\/12\/2018 12:45 AM",
            "cardtail": "XXXX",  //Removed by researchers
            "cardtype": "XXXX",  //Removed by researchers
            ... SNIPPED ...
        }
    ], "count": null
}
```


Figure 1. Javascript Object Notation (JSON) output from getParkerHistory.

Table 1. Required Body Parameters for getParkerHistory Application Programming Interface (API) POST Request

Name	Value
Reporttype	5
Parkerid	7915576
Istransit	1
Apikey	be43ad7b9b541ccff67397dbbc325539

Table 2. Required Body Parameters for Parkerstatus2 POST Request

Name	Value
whitelabel_operator_id	400
ppverifiedparker	DIxPQWIGqKt1zuTtWVtkdRazgyRkd5U7/jx/8Liukbn4QY1c + X2IDkt6RIZVmQ0aNPy/zRGBskwJLbasggBjvw==
parkerid	7915576
apikey	be43ad7b9b541ccff67397dbbc325539

human-readable format; it can be used to repackage edited resources to create an installable APK with modified code.

Target System

The MyJTA mobile application provides users with the ability to review their rider history. This rider history provides a summary of each ticket purchase made by the user, including the fare type, transaction date, and total fee. When the rider opens this view, the application makes an API call to https://ppprk.com/apps/v7/mobile/api/index.php/getparkerhistory (getParkerHistory) to retrieve the user data. For the sake of brevity, this paper will refer to this as the getParkerHistory API. The mobile app vendor provides mobile app payment solutions for both public transportation and parking facilities, therefore the name, "parker history." To analyze the application, the research team created an account via the mobile app. This account served as the target for the discovered vulnerability.

Methodology

The authors captured and observed the getParkerHistory API response in Fiddler and discovered that the response from the server contained far more data about the user than what is displayed in the application user interface. Figure 1 shows the truncated output from the getParkerHistory API call response. The information from the API call includes phone number, last four digits of the credit card used to purchase a pass, ticket number, and ticket details (e.g., color, background, and letter). As the name of the API suggests, license plate number, location, and entry/exit time and date can also be found for parking app users. Apparently, multiple mobile applications deployed by the vendor for both mobile fare payments as well as parking payments used the same set of APIs to store and retrieve user data.

The authors discovered that the getParkerHistory API allows a malicious user to access the information of any other user in the system by first obtaining a valid session ID with one user, and then reusing this valid session ID with another user. Table 1 shows the required request parameters for the getParkerHistory API.

The ppverifiedparker parameter, which appears to be used to verify an authenticated user session, was not one of the required parameters. When providing the four required parameters in a request, the response from the server was the same as when all 17 parameters were provided. As a result, the getParkerHistory API returned user information without verifying the user's ppverifiedparker value. In other words, the server did not validate the pairing of the session ID and the user ID, which allowed a malicious user to reuse their session ID to access the information of any other user.

Table 2 shows the required parameters for a different API, parkerstatus2, which retrieves additional information about the user.

```
{
  "status": 403,
  "message": "Unauthorized",
  "authorizations": null,
  "apiversion": null
}
```


Figure 2. Output from parkerstatus2 without the ppverifiedparker value.

Note that the ppverifiedparker value is a required parameter for this API. Removing or editing this value resulted in a "403 Unauthorized" status code when the API call was executed. Figure 2 shows the output of the parkerstatus2 API call without the ppverifiedparker value.

The getParkerHistory API should have provided a similar response when the ppverifiedparker parameter was omitted from the API request. However, as stated earlier, it did not; instead, the getParkerHistory API returned information about any user in the system.

Results

This vulnerability exposed private information from all users of this system, including name, phone number, and the last four digits of the user's credit card number. Additionally, because parking and transit payment apps share the same server, license plate numbers and parking location history could also be retrieved for parking app users. Figure 3 shows the compromised researcher account information on the rider history screen in the application.

The same vendor has many similar versions of the application deployed for many different public sector agencies (i.e., "white-labeled" solutions) that appear to use the same server. As a result, customers in as many as 40 cities across the United States may have had their information exposed. The exact number of users affected is difficult to calculate with public information, as neither Google Play for Android devices nor the Apple App Store for iOS devices disclose the exact number of app installations. Google Play does publish a lower bound of installations that indicates an order of magnitude of the number of times the application has been downloaded (10,000 + , 50,000 + , 100,000 + , etc.). Additionally, some websites such as Sensor Tower (*20*) estimate the number of iOS installations from the Apple App Store. Based on these resources, an estimated 1,554,000 users across the United States may have been affected by this vulnerability.

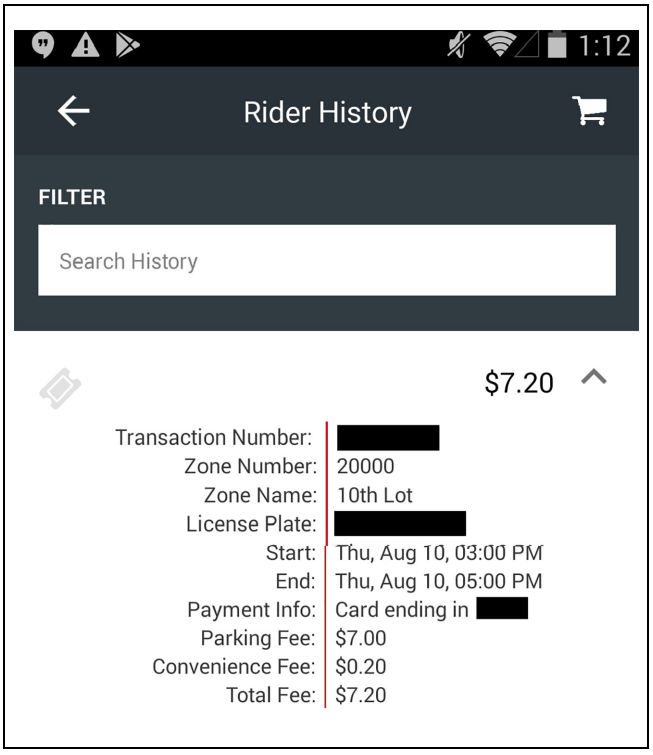

Figure 3. Compromised researcher account displayed in the MyJTA application.

Discussion

On discovering this server-side vulnerability, it was determined that other applications published by the same vendor should be analyzed to determine if they used the vulnerable API call. The PassportParking Mobile Pay application (*21*) was also analyzed. It was found that this application used the same API but a different combination of parameters. This combination of parameters did not result in the same vulnerability when the ppverifiedparker value was omitted from the API request.

However, as discussed above, parking information for a PassportParking account created for the analysis could still be retrieved using the API parameter format from the MyJTA application. In other words, users of the PassportParking app could still have their information exposed by this vulnerability. Figure 4 provides a diagram showing how this vulnerability affected multiple agencies. It is assumed these applications share the same server-side database.

Figure 5 shows how a vulnerability in a white-labeled app can quickly expose many users.

The MyJTA app, which the research team used to discover the vulnerability, has an estimated 15,000 users. This app accounts for less than 1% of all users in the same server database. The main Passport Parking app has an estimated 580,000 users across multiple cities

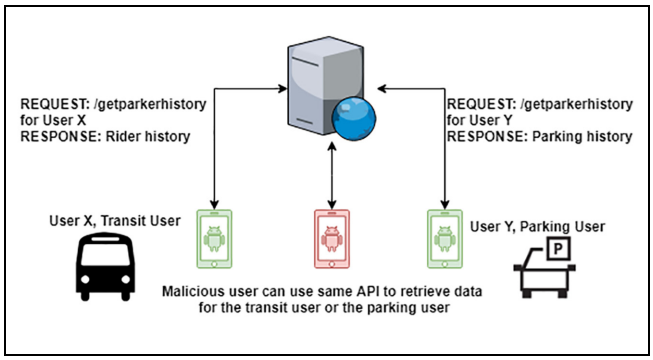

Figure 4. Many apps by the same vendor used the same vulnerable server Application Programming Interface (API).

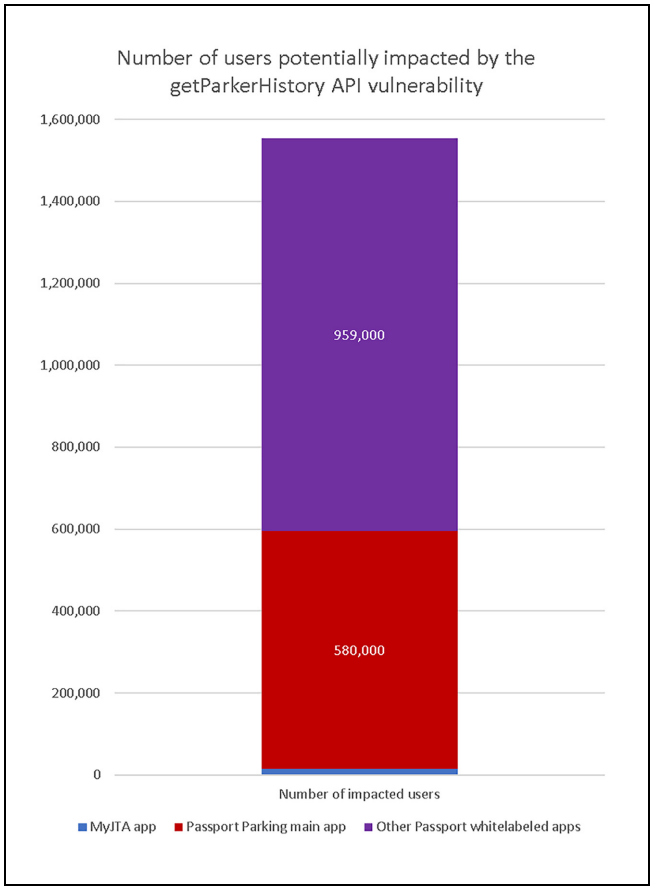

Figure 5. Number of estimated users affected by the discovered vulnerability.
Note: API = Application Programming Interface.

(about 37% of all users). All other white-labeled versions of the app (around 40 apps) account for an estimated 959,000 users (62%). Because of the shared API and database, an estimated 595,000 users were potentially exposed, with up to 1,554,000 users exposed if the other white-labeled applications used the same API and database. The authors have no evidence that the vulnerability was exploited nor that users were affected; these user counts are rough estimates based on publicly available data to gauge the potential severity in the worst-case scenario.

This vendor is not alone in publishing white-label applications. As a result, this paper should not be viewed as singling out a specific agency or vendor but should instead be taken as a warning that this type of vulnerability could potentially happen in any system.

An attacker who can gather information about users by exploiting this vulnerability may be able to combine this information with other sources to steal another user's identity. License plate information and parking locations may provide access to a large amount of additional information, including the user's home and work addresses, which could be combined with the name and credit card information to create a more complete profile. In addition, if an attacker were able to identify the method used by the Passport server to create the QR codes, the attacker may be able to use the information from this API call to recreate tickets from another user's account.

For cyber incidents in the transportation industry, the average cost is $121 for each record involved in the incident (*22*, p. 2); it is currently unknown how this fact relates to the vulnerability discovered by the research team.

Recommended Fix

In disclosing the vulnerability to the agency, the research team provided a recommended fix for the vulnerability. The recommended fix suggested comparing the implementation of the authorization check for the vulnerable API with the API that correctly authenticated the session and user ID pairing. Given that the session and user ID pairing was correctly validated in other API calls made by the application, the team believed that addressing this vulnerability required a very small amount of development effort (e.g., copying code from one section of the server application to another). In addition, the patch was not believed to require a client update as the client application exhibited the correct behavior.

Vulnerability Disclosure Process

The vulnerability was discovered in early October 2018 and was reported to Jacksonville Transportation Authority (JTA) on October 30, 2018. The research team received no further communication from the transit agency or mobile app vendor but confirmed that the vulnerability had been patched by December 8, 2018, via additional testing. The time taken to patch the vulnerability was well within the current industry standard

given the level of effort required to fix the issue as estimated by the research team.

The vulnerability report included the following disclosure statement: "This research project follows a 60-day disclosure deadline. After 60 days elapse or a patch has been made broadly available, the vulnerability will be disclosed to the public." Following vulnerability disclosure policies from various organizations including Google (*23*) and Cisco (*24*), the research team would typically use a 90-day disclosure deadline. However, because of the perceived ease of the vulnerability patch, which only required a server-side change, and the sensitivity of the exposed information, the team shortened the period to 60 days. As mentioned above, the vulnerability was fixed in under 38 days.

The vulnerability report was sent to a staff member from the transit agency participating in a Florida-based cybersecurity working group that was organized by the research team under this same project funded by the Florida Department of Transportation (DOT). Neither the transit agency nor the mobile app vendor published any public vulnerability disclosure policies. If the research team did not have this existing relationship with this staff member, the disclosure would have been sent to the general contact email listed on the transit agency's website. While the vulnerability was fixed, the research team is unaware of the exact process used to communicate and patch the vulnerability. However, this transit agency is not alone. After a review of the websites of transit agencies in Florida, the research team could not find public vulnerability disclosure policies for any agencies. While the vulnerability was patched quickly in this scenario, policies are needed to ensure future incidents are also resolved quickly and with improved communication with relevant parties. The following section describes the need for publicly available vulnerability disclosure processes in detail.

As mentioned earlier, because the vendor's application was "white-labeled," or a shared system with deployments at other transit agencies and cities, up to 40 organizations and their customers may have been affected by this vulnerability across the United States. The research team does not know if the transit agency or the vendor notified any of the other potentially affected organizations or end-users. The Florida Information Protection Act of 2014 (*25*) requires a covered entity to provide notification within 30 days after a breach affecting 500 or more individuals in Florida is discovered. However, this specific Florida law uses a relatively narrow definition of personally identifiable information (PII): it primarily focuses on social security or driver license numbers, full credit card numbers in combination with a PIN, and full plain text usernames and passwords that could be used to access an account directly. If the breach does not meet this definition of PII, it does not need to be reported. Notably, the information obtained by the authors using the discussed vulnerability does not appear to meet this threshold and, as a result, the transit agency and vendor may not be legally required to disclose this vulnerability. The research team did not review the data breach laws for other states, which may have broader definitions of PII or specific requirements in the event of a breach.

Recommendations and Suggested Policies

Many existing general security best practices and policies are effective in the area of transportation security. This section provides recommendations for agencies seeking to improve cybersecurity in their organization. These recommendations are included based on the research team's evaluation of the current state of cybersecurity in public transportation in Florida.

Vulnerability Disclosure Policies

Agencies seeking to deploy mobile fare payment and other mobile applications should discuss how discovered vulnerabilities will be addressed with their vendors, ideally before any contract is signed. Agencies should ask about the timeline for patching discovered vulnerabilities and the vendor's responsibilities if a breach were to occur (e.g., if the agency or passengers will be notified if a vulnerability in the vendor's system is discovered or if the agency will be charged for fixing the vulnerability).

While there does not appear to be a general-purpose federal law that covers data breaches (only "financial institutions" are covered by the Gramm-Leach-Bliley Act [GLBA]), other states have laws similar to the Florida Information Protection Act of 2014 (*26*, *27*). Agencies and their vendors should be familiar with state laws that cover data breaches. By providing expeditious notification of all affected parties in its vulnerability disclosure program, an agency may be more prepared to meet this requirement. Rapid notification of relevant vendors and other agencies may also reduce exploitation of the vulnerability. A federal law could provide a national baseline that would give both the public and private sector a minimum set of thresholds to follow. Ideally, vendors and agencies would alert the public if the vulnerability might have been exploited.

Recommendations for Vulnerability Disclosure Programs. When attempting to report the mobile fare payment vulnerability described in this paper, the research team was unable to find an appropriate contact for the agency or the vendor. After reporting the vulnerability, the research team reviewed several Florida transit agency websites and were unable to find publicly available contacts for any

transit agency in the state of Florida. As transit agencies continue to employ new technologies such as mobile fare payment, the number of discovered vulnerabilities is expected to increase. By providing a clear vulnerability disclosure policy, vulnerabilities may be patched more quickly and communication between agencies, researchers, and vendors can be improved.

A common challenge for successful vulnerability disclosure is the lack of experience for both the vendors and security researchers with regard to accepting or providing vulnerability reports (*28*, pp. 7–8). Smaller organizations, such as transit agencies, may be unprepared to accept vulnerability reports and, because of the sensitive nature of vulnerabilities, inexperienced researchers may be overly aggressive when discussing a timeline with an agency. Providing a clear vulnerability policy will provide a smoother experience for both the agency employees and security researchers.

Providing a prominent location for the vulnerability policy, such as the agency website, will allow vulnerability researchers and vendors to easily discover the policy, especially if the page can be found using a search engine. This prominent location should have a single point of contact listed and should ideally specify what information should be included in the report.

The point of contact could be an online form instead of an email or other form of communication. This will allow agencies more control over the information that is submitted along with the report. Agencies have deployed similar online forms for safety reporting (*29*). These online forms can be programmed to provide other advantages, such as forwarding alerts to all relevant personnel (*29*).

Florida Rule 14-90 Policy Review. As part of the Florida DOT-sponsored Enhancing Security in Public Transportation project (*30*), the research team held monthly working group meetings aimed at increasing communication between security researchers and transit professionals. During a working group meeting held on December 12, 2018, the state of Florida safety and security regulatory infrastructure for bus transit systems, Florida Rule 14-90, was discussed. Currently, Rule 14-90 does not specifically address cybersecurity. The participants discussed potential cybersecurity-related additions to Security Program Plans (SPP) for Rule 14-90. Following this meeting, the research team provided Florida DOT with the following suggested language for disclosure policies. This addition would require Florida agencies to include the following system activities in Section 3 of their SPP:

- (l) A public cybersecurity vulnerability disclosure policy that includes:
  a. a single, public point of contact at the bus transit system for disclosure of vulnerability reports.
  b. expeditious notification of any and all potentially affected or in-danger parties, including users of the system.
  c. practical and timely steps to mitigate and recover from known vulnerabilities.
  d. a location for prominent public display of the policy (e.g., on the agency's website).
  e. compliance with the Florida Information Protection Act of 2014 (25).
- (m) Contractual templates used by the bus transit system to engage contractors and vendors that require these entities to comply with the bus transit system public cybersecurity vulnerability disclosure policy described in Section (3) (l). Contractors and vendors shall report all known vulnerabilities to the bus transit system in a timely manner and shall describe in the contract practical and timely steps to mitigate and recover from known vulnerabilities without additional charge to the bus transit system (e.g., as part of a maintenance agreement).

Other Cybersecurity Policies

For agencies seeking to deploy in-house solutions, their developers should be sure to follow industry best practices. Care should be taken when handling user input and authenticating users. Organizations such as the Open Web Application Security Project (OWASP) (*31*) provide training and informational resources for developers looking to develop new applications securely. Agencies should also conduct internal security reviews for their in-house applications.

Agencies should also review their cybersecurity processes and training. Management should provide resources to train employees to be wary of potential threats, such as phishing emails. In addition, employees should be provided a process for reporting suspected vulnerabilities or suspicious events and encouraged to report anything they observe.

When available, agencies should employ authentication and encryption with their operational technologies. Requiring authentication and encrypting communications may prevent attacks from less sophisticated attackers. Default passwords for technologies are commonly available on the Internet, and attackers may cause significant damage simply by gaining access to a system. This is especially true for isolated technologies, such as traffic cabinets or roadside signs, where an attacker may easily access the device. For example, attackers can purchase physical keys to traffic cabinets online. An example

policy may require technicians to use unique passwords for deployed technologies and disable any unnecessary applications/protocols such as Telnet that are not used in operations.

In the event of an attack, an agency that deploys proper logging technologies may be able to determine the source of the attack and prevent future attacks from the same vulnerability. Agencies should also maintain backups of systems to help recover from events. Responding to and recovering from a breach is often costly. If the source of the attack is not discovered, the costs may increase further as a result of repeated attacks.

Conclusion

Mobile fare payment apps are a critical nexus between customer and agency financial information and therefore critical to agency operations. This paper introduced a new form of attack on mobile fare payment applications and described a vulnerability susceptible to this attack discovered in a Florida mobile fare payment application. Because of the system architecture in which multiple transit and parking fare payment applications by the same vendor share the same server, this vulnerability may have affected customers in as many as 40 cities across the United States, an estimated 1,554,000 users. Lessons learned from the vulnerability disclosure process followed by the research team as well as recommendations for public agencies seeking to improve the security of these types of applications have been discussed.

Given the increasing number of mobile fare payment applications and the agencies' desire to provide more functionality to increase passenger numbers, as well as vendors pushing to implement payment solutions across multiple modes of transportation for "Mobility as a Service (MaaS)," cybersecurity issues such as the ones discussed in this paper will only become more prevalent in the future.

Acknowledgments

The authors would like to thank Gabrielle Matthews for serving as the Florida DOT project manager.

Author Contributions

The authors confirm contribution to the paper as follows: study conception and design: S. Barbeau, J. Ligatti; data collection: K. Dennis, M. Alibayev; analysis and interpretation of results: K. Dennis, M. Alibayev, S. Barbeau, J. Ligatti; draft manuscript preparation: K. Dennis. All authors reviewed the results and approved the final version of the manuscript.

Declaration of Conflicting Interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This paper is part of the project "BDV25-977-51 Enhancing Cybersecurity in Public Transportation" funded by the Florida Department of Transportation and the National Center for Transit Research.

References

1. Ng, A. *How the Equifax Hack Happened, and What Still Needs to be Done*. CNet, 2018. https://www.cnet.com/news/equifaxs-hack-one-year-later-a-look-back-at-how-it-happened-and-whats-changed/. Accessed July 24, 2019.
2. Ferguson, S. *Atlanta's Ransomware Attack Cost Around $2.6M – Report*. Securitynow, 2018. https://www.securitynow.com/author.asp?section_id=613&doc_id=742502.
3. Babcock, S. *City: Cyber Attack against Baltimore's 911 Computer-Aided Dispatch System was Ransomware*. Technically, Baltimore, 2018. https://technical.ly/baltimore/2018/03/29/city-cyber-attack-baltimores-911-computer-aided-dispatch-system-ransomware/.
4. Schwartz, M. J. *Equifax's Data Breach Costs Hit $1.4 Billion*. Bank Info Security, 2019. https://www.bankinfosecurity.com/equifaxs-data-breach-costs-hit-14-billion-a-12473. Accessed July 31, 2019.
5. Chuang, T. *Ransomware Strikes CDOT for Second Time Even as Agency Still Recovering from First SamSam Attack*. The Denver Post, 2018.
6. Georggi, N. L., S. Barbeau, and A. Joslin. *Assessment of Mobile Fare Payment Technology for Future Deployment in Florida*. Florida Department of Transportation, 2016.
7. Tavilla, E. *Transit Mobile Payments: Driving Consumer Experience and Adoption*. Federal Reserve Bank of Boston, MA, 2015.
8. *First Urges Bus Users to Swap Cash for mTickets*. Passenger Transport, 2016. http://www.passengertransport.co.uk/2016/12/first-urges-bus-users-to-swap-cash-for-mtickets/. Accessed July 20, 2018.
9. QRcode.com. *DENSO WAVE*. http://www.qrcode.com/en/. Accessed July 30, 2018.
10. Finžgar, L., and M. Trebar. Use of NFC and QR Code Identification in an Electronic Ticket System for Public Transport. *Proc., 19th International Conference on Software, Telecommunications and Computer Networks*, Split, Croatia, IEEE, New York, 2011, pp. 1–6.
11. Bluetooth Technology Website. Bluetooth. https://www.bluetooth.com/. Accessed July 25, 2018.
12. Kieseberg, P., M. Leithner, M. Mulazzani, L. Munroe, S. Schrittwieser, M. Sinha, and E. Weippl. QR Code Security.

Proc., 8th International Conference on Advances in Mobile Computing and Multimedia, SBA Research, Vienna, 2010, pp. 430–435.
13. Conger, K. *Rare Malware Targeting Uber's Android App Uncovered*. Gizmodo, 2018. https://gizmodo.com/rare-malware-targeting-ubers-android-app-uncovered-1821753862.
14. Wang, G., B. Wang, T. Wang, A. Nika, H. Zheng, and B. Y. Zhao. Ghost Riders: Sybil Attacks on Crowdsourced Mobile Mapping Services. *IEEE/ACM Transactions on Networking*, Vol. 26, No. 3, 2018, pp. 1123–1136.
15. Brown, D. *Could Uber, Lyft Drivers Trick the Apps to Increase Surge Pricing? Experts say 'Probably'*. USA Today, 2019. https://www.usatoday.com/story/tech/2019/05/15/uber-lyft-drivers-can-probably-manipulate-apps-charge-you-more/3678461002/. Accessed July 30, 2019.
16. *MyJTA Mobile Application*. MyJTA. http://myjta.com/. Accessed April 23, 2019.
17. *Fiddler*. Telerik Fiddler. https://www.telerik.com/fiddler. Accessed April 23, 2019.
18. *Android Studio Download Page*. Android Studio. https://developer.android.com/studio/. Accessed April 23, 2019.
19. *APKTool Docs*. iBotPeaches, APKTool. https://ibotpeaches.github.io/Apktool/.
20. *Sensor Tower Homepage*. Sensor Tower. https://sensortower.com/. Accessed July 31, 2019.
21. *Passport Parking Mobile Application*. Passport Parking. https://ppprk.com/park/. Accessed April 23, 2019.
22. Countermeasures Assessment and Security Experts LLC, and Western Management and Consulting LLC. *NCHRP Web-Only Document 221/TCRP Web-Only Document 67: Protection of Transportation: Protection of Transportation Infrastructure from Cyber Attacks: A Primer*. Transportation Research Board of the National Academies, Washington, D.C., 2016.
23. *How Google Handles Security Vulnerabilities*. Application Security, Google. https://www.google.com/about/appsecurity/. Accessed June 24, 2019.
24. *Cisco Security Threat and Vulnerability Intelligence*. Vendor Vulnerability Reporting and Disclosure Policy, 2014.
25. *Chapter 501 Section 171 - 2014 Florida Statutes*. The Florida Senate. https://www.flsenate.gov/Laws/Statutes/2014/501.171. Accessed June 24, 2019.
26. McDaniel, P., and K. Lipscomb. *Data Breach Laws on the Books in Every State; Federal Data Breach Law Hangs in the Balance*. Security Privacy Bytes, 2018. https://www.securityprivacybytes.com/2018/04/data-breach-laws-on-the-books-in-every-state-federal-data-breach-law-hangs-in-the-balance/. Accessed July 30, 2019.
27. Hennessy, J. J., C. T. Howell, J. L. Rathburn, S. M. Millendorf, A. K. Tantleff, and S. D. Goldstick. *State Data Breach Notification Laws*. Foley, 2019. https://www.foley.com/en/insights/publications/2019/01/state-data-breach-notification-laws. Accessed July 31, 2019.
28. van der Meulen, N., S. Gunashekar, S. Soesanto, and E. A. Jo. *Good Practice Guide on Vulnerability Disclosure: From Challenges to Recommendations*. Enisa, 2015. https://www.enisa.europa.eu/publications/vulnerability-disclosure.
29. Staes, L., and J. Godfrey. *TCRP Research Report 218 Pre-Publiction Draft: Characteristics and Elements of Non-Punitive Safety Reporting Systems for Public Transportation*. Transportation Research Board of the National Academies, Washington, D.C., 2020.
30. *National Center for Transit Research*. Enhancing Cybersecurity in Public Transportation. https://www.nctr.usf.edu/2018/02/enhancing-cybersecurity-in-public-transportation-scope/. Accessed June 2, 2018.
31. *OWASP Overview*. Owasp. https://www.owasp.org/index.php/Main_Page. Accessed July 31, 2018.