Legislative Assembly for the
Australian Capital Territory
Standing Committee on Environment,
Planning, Transport and City Services

Submission Cover Sheet

Inquiry into the procurement and delivery of MyWay+

Submission number: 056

Submitter: Patrick Reid

Date authorised for publication: 13 March 2025

MyWay+ has had at least two disclosed security vulnerabilities within the first two weeks of public operation. Within this submission, I will:

1. What are the two disclosures and their technical cause
2. Industry best practices that could have prevented them
3. Prior warning and the Swiss cheese model
4. Calls to action
5. Personal conclusions

The two disclosures

On the launch day of MyWay+, it was discovered [1] (in part by me) that the balance transfer mechanism was missing critical (server side) checks, including:

1. If the MyWay card had already been claimed
2. If the MyWay card balance matched the balance to be imported

In practice, this meant that with sufficient technical skill (or by following this screenshot [2]) you could ask MyWay+ for as much free money as you want. As noted in [1], this was reported to the ASD within hours of a proof of concept, and the system was temporarily suspended.

The second security venerability was found and fleshed out over the next two weeks. Shaun goes into detail on the timeline and scope [3]. For the sake of the reader, the following details were leaked on a **public, unauthenticated** endpoint:

- Full (first and last) names
- Phone numbers
- Email address
- Physical (home) address
- Password hash and salt (this is not as severe as you think [4])
- Full MyWay+ Card number(s), CVV(s), and other info
- First 6 and last 4 numbers of credit/debit card(s) (see [3] for risk evaluation)

Not only were all of these details on the public internet, with no authentication, they were also very easy to index. All account numbers are serial (with small gaps). This means that within the range $[600, 800]$, there is at least 30 users information. Scraping this is very easy. Just ask for user 1, followed by user 2, and so on until you have every registered user. This is why the federal government recommends the usage of random identifiers [5].

At the time of writing, we do not know if this attack has been used by a malicious actor. The information laid out above could, at the very least, be used for a very convincing phishing scam. At worst, there is a chance it could be used for mass identity theft.

Best practices and prevention

Each of these vulnerabilities show us a best practice that NEC's developers skimped out on. Specifically:

- They trusted the user's browser
- They did not follow standard API structure (like REST)

It is relatively common knowledge [6] that server side validation is important. For any number of reasons, the site running in the user's browser may misbehave. They may have a flaky internet connection, leading to the request being re-tried. This is what tipped me off to the first security vulnerability. Or, some malicious actor may choose to exploit unchecked requests for personal gain.

The second standard that NEC chose to ignore is REST (or any other form of request structure). REST is a set of principles intended to make APIs easier for developers to model, memorize, and secure. Let's say I am a developer at NEC and I want to create an endpoint that lists a user's fare media (e.g. debit card, MyWay+ card). I might write an API that is structured something like this:

```
GET api.mywayplus.au/account/{accountid}/fare-media
```

If you ask a developer 'what should you protect for this API?', they will be able to quickly tell you that everything after `/account/{accountid}` should only be accessible by the user. By comparison, the developers payed by NEC wrote the following endpoint:

```
GET api.mywayplus.au/abt-account-management/abt/account/faremedia/fare-media-list
```

Ignoring how messy this API call is, can you spot what I should be using to protect this API? The answer is that there is nothing. The account ID is passed in as a query parameter (`?accountNo={accountid}`). This is particularly bad because the framework that NEC uses to build MyWay+ (spring) does not support authenticating query parameters out of the box[7]. It does support authenticating REST style APIs, like the example I provided above, because those are the industry standard, written by most[8] developers.

Prior warning

I would like to quote a document that was sent to TCCS by PTCBR on my behalf on late November:

> They also fail to follow industry technical standards. For example, communication between the client and server is done using http(s) requests. Normally, these would follow a standardised structure, known as REST. This is good because:
> - They reduce the number of mistakes that developers make
> - Make it easier to apply access control and validation rules consistently
> - Makes systems more reusable

I hate to say 'I told you so,' but I did. The lack of access control or authentication lead to a lot of ACT residents having personally identifying information placed, publicly available, on the internet by NEC and TCCS.

Calls to action

Whilst TCCS and NEC have patched the leak of personal information onto the internet, there are still actions that need to be taken:

1. I believe that Shaun's vulnerability counts as a Notifiable Data Breach. The ACT government MUST notify those effected under Australian law.
2. The MyWay+ system should have a security audit, ideally by a Canberra based company with employees using the PT network.

For future large software projects, the ACT government should:

1. Perform a security audit, ideally by a local company who has skin in the game
2. Be ready to delay a project if it is not ready. A late project is better than one that leaks partial credit card numbers

Personal conclusions

It is now my turn to level a bit of criticism at the terms of reference of the Inquiry. The MyWay+ system is not 'bespoke product'. NEC (at least prior to the launch) expects to resell MyWay+ as a generic account based ticketing (ABT) solution[9]. In fact, they claim that they are operating this system in three other cities already. A bespoke solution made by a Canberra based company both would have been of higher quality, and would have been more aware of the quality expectations of Canberrans. A bespoke system would have been perfect, MyWay+ is not bespoke.

Refocusing on NEC, their developers seem like they are incompetent. On launch day of MyWay+ there were significant accessibility failures, amateur design, and leaking developer phone numbers. These continue to this day, with focus selectors around tabs being broken, an enormous amount of visual inconsistency and unclear design. Even the developer's phone number (+91-7838012934) is still included in the strings file after three months[10].

Footnotes

[1] https://bob-from-canberra.neocities.org/

[2] https://devtoolstips.org/tips/en/replay-xhr/
[3] https://sfulham.github.io/blog/mywayplus-vulnerabilities
[4] https://www.youtube.com/watch?v=8ZtInClXe1Q
[5] https://www.counterfraud.gov.au/fraud-countermeasures/unique-identifiers
[6] https://stackoverflow.com/a/15855799
[7] https://docs.spring.io/spring-security/reference/servlet/authorization/authorize-http-requests.html
[8] I am using 'most' here because other query structures (graphql, grpc, etc) exist. However, these are generally used in applications with a higher performance sensitivity than MyWay+
[9] https://www.nec.com.au/industries/transportation/nec-mobility-platform
[10] https://mywayplus.transport.act.gov.au/assets/i18n/en.json