ASSURED

SECURITY CONSULTANTS

Report

Guardian app API verification test

Emilie Barse, Wictor Olsson, Albin Eldstål-Ahrens

Project	Version	Date
DNS004	1.2	2024-03-19

ASSURED
SECURITY CONSULTANTS

REPORT

Project	Version	Date
DNS004	1.2	2024-03-19

Executive summary

Between 2024-02-12 and 2024-02-26 Assured Security Consultants performed a security assessment of the Guardian app backend API on behalf of DNSfilter. Between 2024-03-11 and 2024-03-13, Assured was tasked with verifying the fixes and mitigations put in place as a result of the original penetration test.

The backend API(s) Housekeeping and SGW were in scope.

The assessments were performed using white-box methodology, where Assured consultants were provided with source code to review and a standalone test environment to run dynamic tests against.

In summary Assured consultants made several observations. Common issues which are linked to business logic abuse, authentication, access control, input validation, security configuration and best practices. None of the issues are rated critical and the majority of the findings are of low risk rating.

The original penetration test uncovered issues with the following risk severity assessments (number of issues):

Critical 0 High 1 Medium 8 Low 10

A majority of the reported vulnerabilities were mitigated, with work underway to follow recommendations on two findings. Three findings were partially mitigated, with their remaining risk accepted by Guardian.

Assured would like to thank Constantin Jacob for the support during this penetration test and the subsequent verification test. We are happy to answer any questions and provide further advice.

Contents

1 Observations

3.1 HIGH FIXED Infinite 3-day trial tokens

Likelihood: MEDIUM (5), Impact: HIGH (6)

Verification Note: The reported issue has been mitigated. Free trials are now based on an in-app purchase, and app store receipts are verified before providing the user with a trial token.

An API endpoint was identified which did not require authentication and could be used for abusing the business logic. In this case it would result in the attacker gaining free access to the Guardian app service, which could be considered fraudulent.

3.2 MED FIXED Merge of day passes enables manipulation of total balance

Likelihood: MEDIUM (4), Impact: MEDIUM (5)

Verification Note: The reported issue has been mitigated. Day pass accounting tokens are now checked before redemption, and rejected if merged into another DPAT.

Day passes can be merged to get an added total balance on the first day pass code. It is possible to use redemption with merged day pass codes to get a higher balance than what was purchased.

3.3 MED FIXED Create a free day pass token

Likelihood: LOW (1), Impact: HIGH (6)

Verification Note: The hardcoded master token has been disabled in the production environment.

A hardcoded test value was identified in the code which can be abused to bypass typical logic validation. An attacker with knowledge of (or the ability to guess) the secret text string can create unlimited free day pass tokens.

3.4 MED FIXED Unauthenticated Partner account creation

Likelihood: MEDIUM (5), Impact: MEDIUM (4)

Verification Note: The affected endpoints have been disabled entirely.

An unauthenticated endpoint for partner administrative user creation was identified. An attacker can create a partner user without any form of validation which in turn could be abused within the system. Being able to create a user of another role might not directly be harmful but will in most cases open up additional functionality to the attacker which in turn could be abused.

3.5 MED FIXED No password complexity requirement

Likelihood: HIGH (6), Impact: LOW (2)

Verification Note: Passwords are now required to be between 14 and 72 characters and contain at least one letter and one number.

Password complexity requirements is common to strengthen credential based authentication. There is no password complexity policy in place in the API which can lead to weak credentials.

3.6 MED REMAINING No multi-factor authentication

Likelihood: HIGH (6), Impact: LOW (2)

Verification Note: MFA support has not been implemented, in part due to privacy concerns with phone-based methods. Guardian is investigating other mechanisms such as TOTP or FIDO2.

No multi-factor authentication was identified to be in use in the application. Multi-factor authentication is an electronic authentication method in which a user is granted access to a website or application only after successfully presenting two or more pieces of evidence to an authentication mechanism. This is a mechanism to combat credential based attacks which are common.

3.7 MED FIXED Publicly accessible hidden/secret functions

Likelihood: MEDIUM (3), Impact: MEDIUM (3)

Verification Note: The majority of the endpoints have been removed from public access, and instead attached to an internal network interface. The remaining endpoints are not deemed secret or sensitive.

Several interesting endpoints were identified which seem to be used for debugging or similar development functions. Exposing such endpoints in production might be tempting but could expose sensitive functionality publicly.

3.8 MED FIXED Unsanitized input parameters causing partial SSRF

Likelihood: MEDIUM (3), Impact: MEDIUM (3)

Verification Note: The reported vulnerabilities have been mitigated by adding input validation to the vulnerable parameters. Additional recommendations can be found at the bottom of this section.

A partial SSRF (Server-Side Request Forgery) vulnerability was found at several different locations in the code. Unsanitized user input parameters are added in the URL path in third party API requests, and can be used for changing the requested URL in an unintended way. Server-side request forgery is a web security vulnerability that allows an attacker to cause the server-side application to make requests to an unintended location. In this case, the requested host cannot be changed, but the URL path can be changed.

Additional recommendations: The mitigation put in place is specific to SSRF vulnerabilities by path traversal, i.e. including an element such as `../../target` in the path portion of a URL. This filtering does not protect against injection into other parts of a URL, such as query parameters.

We recommend applying the strictest possible limits to the format of each individual user-provided parameter, informed by the expected data format. For example, if a parameter is expected to contain a username and usernames are limited to alphanumeric characters, the parameter should also be rejected if it contains non-alphanumeric characters.

If a field must support all characters (e.g. a free text input field), encode the data in a safe manner before using it, e.g. using URL encoding or base64.

3.9 LOW ACCEPTED Input validation of strings missing

Likelihood: MEDIUM (4), Impact: LOW (2)

Verification Note: The reported endpoints accept input parameters with no format validation. No security impact has been demonstrated in the current use. The remaining risk has been accepted by Guardian.

Strings with unrestricted content can in some cases cause security issues such as causing injection attacks in user interfaces (XSS), injection in databases (SQL injection), when concatenated with other strings, or when sent to third-party systems (e.g. email, log management and alerting, ticketing systems). Also, strings of unrestricted length can be used for denial-of-service attacks or increasing cloud usage costs by consuming a lot of resources.

Almost all input strings are accepted without any further checks. An exception is the email address which is validated in some request.

3.10 LOW REMAINING Weak random number generators

Likelihood: LOW (2), Impact: MEDIUM (3)

Verification Note: Work is underway replacing the random number generator with `crypto/rand`, where relevant. At the time of verification, a number of security-related random numbers were still generated using the insecure `math/rand`.

Use of weak random number generators will lead to low or in worst case predictable entropy. If used in relation with security sensitive or business logic functionality it could create an opportunity for an attacker to predict or circumvent these functions.

In several places in the codebase sub-par (not suitable for sensitive use-cases) random functions are used (math/rand). A practical attack has not been constructed during this test.

3.11 LOW FIXED Binding socket to all interfaces

Likelihood: MEDIUM (3), Impact: LOW (2)

Verification Note: The HTTP service has been reconfigured to only listen on the loopback interface.

Binding serving sockets to all interfaces is usually the default behaviour of many applications. Without application external limitations such as firewall rules this will expose functionality on all available interfaces. The service will be exposed on all interfaces which might not have been intended and could result in an attacker being to access to service in a unintended way.

3.12 LOW ACCEPTED WriteFile permissions

Likelihood: LOW (1), Impact: MEDIUM (4)

Verification Note: One reported function has been removed. The remaining two locations are scheduled for deprecation.

The application writes files with lax filesystem permissions, these writes are deemed low risk due to the current access model of the application servers. An attacker in the right context could potentially abuse the privileges to access or modify the content of the files.

3.13 LOW FIXED Vulnerable libraries in use

Likelihood: MEDIUM (3), Impact: LOW (2)

Verification Note: All reported dependencies have been updated to versions without known vulnerabilities.

Third party libraries/components which are used in the application were identified to have known vulnerabilities. Unpatched issues could potentially be exploited with the right preconditions by an attacker. The vulnerabilities are a HTTP DoS, JWT validation bypass, header manipulation and a partial protocol downgrade which could lead to partial traffic interception/modification.

The issues identified in the libraries are not deemed to be easily exploitable.

3.14 LOW FIXED User enumeration

Likelihood: MEDIUM (3), Impact: LOW (2)

Verification Note: A number of affected endpoints have been removed. The remaining reported vulnerabilities have been mitigated by normalizing responses and introducing random timing noise.

Exposed functions which allow user enumeration through timing and deterministic response messages. In the timing scenario a SQL query is executed to lookup the user which will return user data to application on success and continues operation to validate password. Measured difference on test system was 40ms for not existing email and 400ms for existing during this operation. In the other scenario a few endpoints were identified which responds with a message indicating the existence of a user.

These functions could be abused by both an authenticated as well as unauthenticated attacker to enumerate users present in the system. Attackers will typically abuse this to build lists of users which can be targeted through credential stuffing or spraying attacks.

3.15 LOW ACCEPTED Rate limiting missing in Housekeeping API

Likelihood: LOW (2), Impact: MEDIUM (4)

Verification Note: At the time of verification, the Housekeeping API had no rate limiting applied. However, rate limiting is deprioritized in order to maintain usability for new users during normal surges.

Rate limiting on API requests is a valuable tool to prevent a single user/IP from using too much resources (network, CPU, memory, and storage).

There seems to be no rate limiting implemented for any of the API requests in the housekeeping API. This means that an attacker can send many requests in a short time which may consume a lot of CPU resources. Some request, which also stores data in the database may cause a lot of storage to be used, like the request for creating partner apps.

3.16 LOW FIXED User signup endpoint allows arbitrary email

Likelihood: LOW (2), Impact: MEDIUM (3)

Verification Note: Signup is limited to the email address associated with a signup token.

The endpoint `/users/signup` can be used for setting an arbitrary email on the created user account.

The limitation is that the email cannot already be in the users database. Also, the account does not seem very useful without a subscription linked to the user ID. However, it may still have some security impact that an attacker can create an account with an unverified email address.

3.17 LOW FIXED Invalidate credential call fails without notifying user

Likelihood: LOW (2), Impact: MEDIUM (3)

Verification Note: The API has been updated to ensure a token older than 24h is not accepted.

The version 1.2 endpoint for invalidating the subscriber-credential for the SGW API fails to do anything, but still returns the HTTP code 200 OK. Running the same request with the version 1.3 endpoint changes the `invalidated` flag in the database.

3.18 MED FIXED Authentication token expiry missing

Likelihood: MEDIUM (3), Impact: MEDIUM (5)

Verification Note: The API has been updated to ensure a token older than 24h is not accepted.

Rating Note: The finding was originally given a Low severity by Assured. Upon reporting, Guardian increased the severity to Medium.

The authentication token (`auth_token`) does not have an expiry. The user gets it at signup or login. It is valid as long as the user does not logout using the `/api/v1/users/web-sign-out` endpoint.

If an attacker gets hold of the user token, it will be valid indefinitely. It is of similar severity as if the attacker gets hold of the users email and password. Though, the authentication token is not very useful unless the user have a subscription.

3.19 LOW FIXED Go profiling endpoint exposed

Likelihood: MEDIUM (3), Impact: LOW (2)

Verification Note: The sensitive endpoint has been moved to an internal interface, and is no longer exposed.

A Golang profiling endpoint is exposed under /debug/pprof.

An attacker who can access pprof functionality can use it to make the server leak information about things like function names, file paths, business sensitive information as well as open up for DoS attacks.