IMMUNIWEB MOBILE APP SECURITY TEST

Table of Contents

General Overview

API Specifications

Field Name	Value
Protocol	HTTPS
Request Type	GET/POST
Entry-point URL	https://www.immuniweb.com/mobile/api/

POST Specifications

Field Name	Value
malware_check	Possible values: • 0 – the application will not be scanned with VirusTotal. • 1 – the application will be scanned with VirusTotal.
hide_in_statistics	Possible values: • 0 - the test results will be displayed in Latest Tests section on ImmuniWeb's website. • 1 - the test results will be hidden.
app_id	Indicates the tested application's ID.
store_id	Indicates the store from which the app should be downloaded. Possible "store_id" values: • "googleplay" for Google Play • "appgallery" for "Huawei AppGallery • "fdroid" for F-Droid • "rejail" for ReJail • "mainrepo" for Mainrepo
api_key	The optional secret token which you can use to unlock API's full functionality and test results. Can be purchased from https://immuniweb.com/mobile/limits/ page.

# How-To Upload APK/IPA file and start the test:

Request (the “api_key” parameter is optional and can be omitted):

```
curl -F "malware_check=0" -F "hide_in_statistics=0" -F "api_key=YOUR-API-KEY" -F
"file=@MyApplication.apk" "https://www.immuniweb.com/mobile/api/upload"
```

Response example:

```
  [{
    "status": "success",
    "message": "Validation successful",
    "device_type": "android",
    "sha256": "1e238b9adbe768ead29e65f30cdc518d8fa85fc2bc65030552ec970",
    "total_files": 2483,
    "total_size": 33458748,
    "core_size": 7439820
  },
  {
    "status": "success",
    "id": "string",
    "short_id": "string",
    "app_developer": "",
    "ts_start": 1583748106,
    "package_total_files": 2483,
    "package_total_size": 33458748,
    "package_core_size": 7439820,
    "show_test_results": 1,
    "malware_check": 0,
    "test_source": "upload" }]
```

# How-To Get the test results:

In the previous example, if the app is found and test is started, you will get a test ID in the response. Once you have the test ID, you can query API for test results. You can query either by the full ID (id) or by short ID (short_id).

Request (the “API_KEY” is optional, include it in the request to receive the full data):

```
curl https://www.immuniweb.com/mobile/api/test_info/id/[TEST_ID]/[API_KEY]
```

The response will be covered in details later in the document.

# How-To Delete the test (possible only for manually uploaded APK/IPA files):

An example of the request and the structure are as follows:

Request:

```
curl https://www.immuniweb.com/mobile/api/delete/id/[TEST_ID]/[GSESS_VALUE]
```

where GSESS_VALUE should be replaced by “gsess” cookie’s value that you can get after logging in to your account on ImmuniWeb Portal.

Response:

```
{ "status": "true", "message": "Test has been deleted." }
```

# How-To Download the app from Google Play and start the test:

Request (the “api_key” parameter is optional):

```
curl -d "app_id=com.viber.voip&store_id=googleplay&api_key=YOUR-API-KEY"

"https://www.immuniweb.com/mobile/api/download_apk"
```

The response is covered in details in the “Download from Play Store” section.

# How-To Refresh the test by redownloading (possible only for APKs downloaded from Google Play)

Request:

```
 curl -d "api_key=YOUR-API-KEY"
"https://www.immuniweb.com/mobile/api/refresh/id/[TEST_ID]"
```

# How-To Download the PDF report

```
curl -d "api_key=YOUR-API-KEY"
https://www.immuniweb.com/mobile/gen_pdf/[test_id]/ > report.pdf
```

The returned data will be different for finished and in progess tests of the application:

In progress: the application is in the process of uploading and scanning, contains details of the API level of the uploader, the number of tests made, and the total estimated time of completion.

“In progress” status

Corresponds to details for when the application is in the process of uploading and testing, such as API details, test ETA and intermittent SAST data. Those section will be covered in details later in the document. An example:

```
{
  "status": "in_progress",
  "message": "Total estimated time before report delivery is [ETA] minutes.",
  "is_deletable": {
    "status": true,
    "message": "All is ok!",
    "message_id": 36
  },
  "is_refreshable": {
    "status": false,
    "message_id": 26,
    "message": "Test can not be refreshed because it’s manually uploaded."
  },
  "test_eta": 1501,
  "test_elapsed": 314,
  "test_debug": false,
  "test_source": "upload",
  "test_app_info": false,
  "test_sast": false,
  "test_dast": [],
  "test_behaviour": false,
  "test_sca": false,
  "test_apis": false,
  "scores": {},
  "data": {
    "mitm": false,
    "video": false,
    "malware_check": false,
    "show_test_results": true
  },
  "is_cutted": false,
  "cutted_router": "user_not_logged_in" }
```

Finished: the output of a successfully uploaded and processed application will contain:

- meta-information: status, the number of tests made, the number of tests in the queue, and other high-level information.
- is deletable/is refreshable sections, specifying whether the test report can be deleted or refreshed
- scores: contains the scores of the test
- data: contains the main data of the test

All of those sections will be detailed later in this document starting with the meta-information section.

Processed response example:

```
{ "status": "finished",
  "message": "Test is finished.",
  "test_eta": 0,
  "test_elapsed": 0,
  "test_debug": false,
  "test_source": "upload",
  "test_app_info": true,
  "test_sast": true,
  "test_dast": [],
  "test_behaviour": true,
  "test_sca": true,
  "test_apis": true,
  "is_cutted": false,
  "cutted_router": "user_not_logged_in",
  "is_deletable": {...},
  "is_refreshable": {...},
  "scores": {...},
  "data": {...} }
```

1. Meta-information

Contains the details of when the application has been uploaded, if the test can be deleted, if it’s refreshable, the status, position in the queue and SAST information. SAST findings will be populated intermittently as the test progresses. The structure is as follows:

Field Name	Type	Always present	Description
status	string	Yes	Details the current test status.
message	string	Yes	Indicates the test status (e.g. if it is done or not).
test_eta	integer	Yes	Indicates an ETA for the test.
test_elapsed	integer	Yes	Indicates the time spent of the test.
test_debug	bool	Yes	Indicates if test debug information is shown.
test_source	string	Yes	Indicates the source of the application.
test_app_info	bool	Yes	Indicates if application information is shown.
test_sast	bool	Yes	Indicates if SAST has been performed.
test_dast	array	Yes	*Deprecated.*
test_behaviour	bool	Yes	Indicates if test behavior is shown.
test_sca	bool	Yes	Indicates if SCA has been performed.
test_apis	bool	Yes	Indicates if APIs have been included in the test.
is_cutted	bool	Yes	“true” if not all results are shown.
cutted_router	string	Yes	Describes the route action to make hidden results available.

is_deletable	object	Yes	Indicates if the test can be deleted. Syntax:
			`{`
			`"status": "bool",`
			`"message": "string",`
			`"message_id": "integer"`
			`}`

is_refreshable	object	Yes	Indicates if the test is refreshable. Syntax:
			`{`
			`"status": "bool",`

			`    "message": "string",`
			`    "message_id": "integer"`
			`}`

scores	object	Yes	Contains the number of found APIs, number of OWASP Top 10 issues, number of found SCA components, and the number of behavior permissions. Basic syntax:
			`{`
			`      "description": "6 issues found",`
			`      "class": "fs-bar-red"`
			`}`

data	object	Yes	Holds application and test information. Will be detailed in the “2. Mobile Application Data” section.

2. Mobile Application Data

The following "data" object holds the application and test information. The structure is:

Field Name	Type	Always present	Description
show_test_results	bool	Yes	Show test results on the IW website?
malware_check	bool	Yes	Indicates if the app was tested for known malware.
video	bool	Yes	Indicates if the test utilized video.
mitm	string	Yes	Details MitM data.
app_info	object	Yes	An object that holds information on the uploading app, such as name, version, device type. Will be detailed in the "2.1. Application info" section.
test_dast	array	Yes	*Deprecated.*
test_sast	object	Yes	An object that holds information about the SAST results of the test, such as intel and vulnerabilities. Will be detailed in the "2.2. SAST test results" section.

test_behaviour	object	Yes	Holds information on the device type and android specific areas in which tests are carried out, such as storage, location and contacts. Basic syntax:
			`{"device_type": "android",`
			` "data": ["camera", "storage", "location"]}`

test_sca	object	Yes	A structure that contains results returned from the software composition analysis. Will be detailed in the "2.3. Software Composition Analysis results" section.
test_apis	object	Yes	A structure that corresponds to results returned from other free service APIs. Will be detailed later in the "2.4. Community Edition Security Test Results" section.

An example of the “data” object:

```
"data": {
    "app_info": {...},
    "test_sast": {...},
    "test_dast": [],
    "test_behaviour": {...},
    "test_sca": {...},
    "test_apis": {...},
    "mitm": "string",
    "video": false,
    "malware_check": true,
    "show_test_results": true
  }
```

2.1. Application info

The “app_info” object contains information on the app, such as name, version, device type. Comprised of the following fields:

Field Name	Type	Always present	Description
app_name	string	Yes	The name of the application.
app_id	string	Yes	The ID of the application.
app_version	string	Yes	The version of the application.
app_developer	string	Yes	The developer of the application.
package_core_size	string	Yes	The size of the core package.
package_total_size	string	Yes	The total size of the package.
device_type	string	Yes	The OS type the app will run on.
ts_start	integer	Yes	When the test was started.
ts_stop	integer	Yes	When the test was stopped.
test_id	string	Yes	The reference ID of the test.
test_short_id	string	Yes	The short reference ID of the test.
test_icon	string	Yes	Indicates if a test icon is present.
test_icon_thumb	string	Yes	Indicates if a test icon thumbnail is present.

2.2. SAST test results

The next section, "test_sast" is a list that holds information about the SAST results of the test, such as intel and vulnerabilities. The basic structure is as follows:

```
test_sast": {
  "app_info": {...},
  "behaviour_data": {...},
  "sca_data": {...},
  "vulns": {...},
  "intel": {...} }
```

Field Name	Type	Always present	Description

app_info	object	Yes	An object that holds information on the uploading app, such as name, version, device type. Syntax:
			`{     "name": "string",`
			`      "package": "string",`
			`      "version": "string"    }`

behavior_data	object	Yes	Details the device type and data areas such as storage, location and contacts. Syntax:
			`  "behaviour_data": {`
			`    "data": {`
			`      "storage": "string",`
			`      "location": "string",`
			`      "contacts": "string"`
			`    },`
			`    "device_type": "string"   }`

sca_data	object	Yes	Contains results returned from the software composition analysis. The structure is detailed in the “2.3. Software Composition Analysis” section.
vulns	object	Yes	A list containing details on the found SAST vulnerabilities within the application. Detailed in the “2.2.1. Vulnerabilities (SAST)” section.

intel	object	Yes	Each discovery is presented in a list with the elements of "information" and "intel". Basic syntax:
			`"intel": {`
			`      "mitm_hosts": {`
			`        "intel": […],`
			`        "information": {…} }`
			Detailed in the “2.2.2. Intel (SAST)” section.

2.2.1. Vulnerabilities (SAST)

The following section of "test_sast" is "vulns", which is a list of vulnerabilities discovered by the service.

Please note: most of the vulnerabilities have the following structure, unless specified otherwise:

Field Name	Type	Always present	Description
proof	object	Yes	The "proof" section is a list of evidence of the found vulnerability.
information	object	Yes	Details information such as CVSSv4 score, an example of the insecure code, OWASP details and warning type, as shown below.

The possible discovered vulnerabilities are:

Field Name	Type	Always present	Description
create_temp_file	object	Yes	Lists details on the application's usage of creating temporary files.
deserialize_ object	object	Yes	Lists details on the applications usage of object deserialization.
execute_raw_ sql_with_inputs	object	Yes	Lists details on the applications usage of SQL with raw input.

hardcoded_info	object	Yes	Lists details on the applications usage of hardcoded data. The structure is as follows:
			`"hardcoded_info": {`
			`    "information": {},`
			`    "vulns": {`
			`      "emails": [],`
			`      "https://": {},`
			`      "passwords": {},`
			`      "usernames": {}`
			`    }`

load_code_ dynamically	object	Yes	Lists details on the applications usage of dynamically loading code.

missing_ antiemulation	object	Yes	Lists details on the applications usage of missing antiemulation.
network_security _configuration_ not_ present	object	Yes	Lists details on the applications usage of network security configuration.
rw_external_ storage	object	Yes	Lists details on the applications usage of reading writing to external storage, such as SD cards.
tapjacking_ protection	object	Yes	Lists details on the applications usage of TapJacking protection.
information_ exposure	object	Yes	Lists details on the applications information exposure.
use _of_backup	object	Yes	Lists details on the applications usage of backup.
use_of_ hardcoded_ credentials	object	Yes	Lists details on the applications usage of usage of hardcoded credentials.
use_of_implicit_ intent	object	Yes	Lists details on the applications usage of usage of implicit intent.
use_of_intent_filt er	object	Yes	Lists details on the applications usage of usage intent filters.
use_of_ unencrypted_ network_ protocols	object	Yes	Lists details on the applications usage of unencrypted network protocols.
use_of_weak_ crypto_algorithm	object	Yes	Lists details on the applications usage of weak cryptography.
use_of_weak_iv	object	Yes	Lists details on the applications usage of weak initialization vectors.
use_of_weak_rng	object	Yes	Lists details on the applications usage of weak random number generators.
use_of_weak_ hash_algorithm	object	Yes	Lists details on the applications usage of weak hash algorithm.
vulnerable_ providers	object	Yes	Lists details on the applications usage of vulnerable providers.
vulnerable_ receivers	object	Yes	Lists details on the applications usage of vulnerable receivers.
vulnerable_ activites	object	Yes	Lists details on the applications usage of vulnerable activities.
vulnerable_ services	object	Yes	Lists details on the applications usage of vulnerable services.

webview_ javascriptcors_ enabled	object	Yes	Lists details on the applications usage of webview with JavaScript CORS enabled.
webview_ javascript_ enabled	object	Yes	Lists details on the applications usage of webview with JavaScript enabled.
webview_load_ remote_url	object	Yes	Lists details on the applications usage of webview loading a remote URL.
webview_use_of_ setpluginstate	object	Yes	lists details on the applications usage of webviews with usage of setPluginState.

2.2.2. Intel (SAST)

The next element of "test_sast" is "intel", which details issues discovered in the app that are classified as 'intel':

Please note: most of the Intel objects have the following structure, unless specified otherwise:

Field Name	Type	Always present	Description
intel	object	Yes	Contains details on the found intel.
information	object	Yes	Details information such as CVSSv4 score, an example of the insecure code, OWASP details and warning type, as shown below.

Intel (SAST) has the following structure:

Field Name	Type	Always present	Description
interact_with_ trustmanger”	object	Yes	Lists details about TrustManger interactions.
interesting_files	object	Yes	Details on the found interesting files.
list_ciphers	object	Yes	Contains the list of the ciphers.
list_libraries	object	Yes	Contains the list of the libraries.
network_methods	object	Yes	Contains the list of the network methods.
rw_internal_ storage	object	Yes	Lists details on the application's file system storage.

use_of_antiemulation	object	Yes	Lists details on the application’s anti-emulation measures.
use_of _base64	object	Yes	Lists details on the application’s usage of base64.
use_of_command	object	Yes	Lists details on the application’s usage of system commands.
use_of_keystore	object	Yes	Lists details on the application’s usage of keystore.
use_of_socket	object	Yes	Lists details on the application’s usage of sockets.
use_of_sql	object	Yes	Lists details on the application’s usage of SQL.
use_of_webview	object	Yes	Lists details on the application’s usage of WebView
webview_can_access_providers	object	Yes	Lists details on the application’s usage of WebView and its access to providers
webview_dom_storage_enabled	object	Yes	Lists details on the application’s usage of webview and its dom storage implementation.
webview_file_uri_can_access_filesystem	object	Yes	Lists details on the application’s usage of webview and its filesystem access implementation.
application	object	Yes	Contains android specific application configuration details, such as allowBackup and vmSafeMode.
components	object	Yes	A list of android components such as activities, services, receivers, keys, etc. Will be detailed later in the “2.2.3. Components” section.
permissions	object	Yes	Contains information on application’s permissions Will be detailed later in the “2.2.4. Permissions” section.

2.2.3. Components

"components" is a list in the "intel" section of android components such as activities, services, receivers, keys, etc. The structure is as follows:

Field Name	Type	Always present	Description

activities	object	Yes	A list of the observed android activities. Syntax:
			`{`
			`  "application": {`
			`    "activity": […],`
			`    "meta-data": […],`
			`    "uses-library": {…},`
			`    "provider": […],`
			`    "service": […],`
			`    "receiver": […]      }`

information	object	Yes	Details such as CVSSv4 scoring, warning type, reference links and OWASP details.

intel	object	Yes	The "intel" part is the section, that contains discovered intel issues in activities, providers, receivers and services. The structure is as follows:
			`  "intel": {`
			`    "activities": [{…}],`
			`    "providers": [{…}],`
			`    "receivers": [{…}],`
			`    "services": [{…}]`
			`  }`

keys	object	Yes	Lists found keys regarding to activity, provider, receiver and service.
			`{ "keys":`
			`        ["activity",`
			`         "provider",`
			`         "receiver",`
			`         "service"]     }`

providers	array	Yes	A list of the observed providers and details.
receivers	array	Yes	A list of the observed receivers and details.
services	array	Yes	A list of the observed services and details.

2.2.4. Permissions

The next section of “intel” is “permissions”. The structure is as follows:

Field Name	Type	Always present	Description
android	array	Yes	Details of android permissions.
custom	array	Yes	Details of custom permissions.
information	object	Yes	Contains the found groups.
groups	array	Yes	Details of found groups.
intel	object	Yes	Details of found instances.
keys	array	Yes	Details of found keys, e.g. permission, permission-group and users-permission.

2.3. Software Composition Analysis results (test_sca)

“test_sca” is a structure that contains results returned from the software composition analysis. The structure is as follows.

Field Name	Type	Always present	Description
proof	array	Yes	The "proof" section contains a list of external libraries.
internal	array	Yes	The “internal” section contains a list of internal libraries.
information	object	Yes	Details information such as CVSSv4 score, an example of the insecure code, OWASP details and warning type, as shown below.

2.4. Community Edition Security Test Results (test_apis)

"test_apis" is a structure that corresponds to results returned from the free service APIs. It contains arrays for Domain, Website and SSL Security Tests respectively. Their basic structure is as follows:

```
"test_apis": {
  "radar": [{}],
  "ssl": [{}],
  "websec": [{}]
}
```

2.4.1. Domain Security Test Results (RADAR)

The first structure is **"radar"**, which is as follows:

Field Name	Type	Always present	Description
id	string	Yes	The ID corresponding to the RADAR test.
short_id	string	Yes	The short ID of the test.
server_ip	string	Yes	The IP address of the server.
country	string	Yes	The country of the detected domain.
ts	float	Yes	The timestamp of the test.
lat	float	Yes	The latitude of the domain.
lng	float	Yes	The longitude of the domain.
orig_url	string	Yes	The original URL of the domain.
phishing	integer	Yes	Indicates if any potential phishing domains were returned.
typosquatting	integer	Yes	Indicates if any potential typosquatting domains were returned.
cybersquatting	integer	Yes	Indicates if a potential cybersquatting domain was returned.

2.4.2. SSL Security Test Results

The second structure is **"ssl"**, which is as follows:

Field Name	Type	Always present	Description
id	string	Yes	The ID corresponding to the RADAR test.
short_id	string	Yes	The short ID of the test.
ts	float	Yes	The timestamp of the test.
has_ssl_tls	bool	Yes	Does the tested domain have SSL/TLS.
grade	string	Yes	The grade of the tested domain.
score	integer	Yes	The scoring indicator of the test.
ip	string	Yes	The IP address of the domain.
port	integer	Yes	The port that the test was carried out through.
hostname	string	Yes	Indicates the hostname of the domain.
reverse_dns	string	Yes	The reverse DNS record of the domain.
http_response	string	Yes	The returned HTTP response.
server_signature	string	Yes	Indicates the server's signature.
nist	bool	Yes	Indicates if the domain is compliant with NIST.
hipaa	bool	Yes	Indicates if the domain is compliant with HIPAA.
pci_dss	bool	Yes	Indicates if the domain is compliant with PCI DSS.

2.4.3. Website Security Test Results (WebSec)

The third structure is **"websec"**, which is as follows:

Field Name	Type	Always present	Description
id	string	Yes	The ID of the test.
short_id	string	Yes	The short ID of the test.
server_ip	string	Yes	The IP address of the server.
grade	string	Yes	The grade of the test.
reverse_dns	string	Yes	The reverse DNS record of the domain.
tested_url	string	Yes	The tested URL.
http_response	string	Yes	The returned HTTP response.
server_signature	string	Yes	Indicates the server’s signature.
redirect_to	string	Yes	Indicates where the server redirects to.
ts	float	Yes	The timestamp of the test.
hostname	string	Yes	Indicates the hostname of the domain.

Download from Play Store

This section describes the process of using the “download from play store” feature present on the Hi-Tech Bridge Mobile App Scanner interface. The output consists of an array with boolean values indicating success of validation of the application package.

Upon success, a list of details such as the device type, sha256 hash, total number of files, and file size.

Upon an unsuccessful validation, the details will show the id, the message and the status.

Below are the corresponding details in relation to a successful validation:

Field Name	Type	Always present	Description
status	string	Yes	Indicates if the validation was successful.
id	string	Yes	The id of the test.
short_id	string	Yes	The short ID of the test.
message	string	Yes	The presented message of a successful validation.
device_type	string	Yes	The application’s device type.
sha256	string	Yes	The SHA256 hash of the application.
total_files	integer	Yes	the total amount of files.
total_size	integer	Yes	the total size of the application package.
core_size	integer	Yes	the core size of the application package.

Below are the corresponding details in relation to an unsuccessful validation:

Field Name	Type	Always present	Description
status	string	Yes	Indicates if the validation was successful.
error_id	integer	Yes	The ID of the error.
message	string	Yes	The error message.
id	string	Yes	The id of the test.
short_id	string	Yes	The short ID of the test.
dbg	object	Yes	Debug information for the unsuccessful test. Will be detailed later in the document.

The following section is the debug information for the unsuccessful test:

Field Name	Type	Always present	Description
id	string	Yes	The id of the test.
short_id	string	Yes	The short ID of the test.
test_source	string	Yes	The source of the application.
score	string	Yes	The scoring of the test.
grade	string	Yes	The grading of the test.
app_name	string	Yes	The name of the application.
app_id	string	Yes	The application ID.
app_version	string	Yes	The version of the application.
device_type	string	Yes	The application's device type.
test_sast	string	Yes	SAST test information for the application.
test_dast	array	Yes	*Deprecated.*
test_behaviour	string	Yes	The behavior information of the test.
test_apis	string	Yes	The test APIs.
test_debug	string	Yes	Debug information regarding the test.

package_total_ files	string	Yes	Total number of files in the package.
package_total_ size	string	Yes	The total size of the package.
package_core_ size	string	Yes	The core size of the package.
show_test_ results	string	Yes	If the test results will be shown on IW website.
ts_start	string	Yes	A timestamp of when the test was started.
ts_stop	string	Yes	A timestamp of when the test was stopped.
user_ip	string	Yes	The IP address of the uploader.
user_agent	string	Yes	The user agent of the uploader.
user_city	string	Yes	The city of the uploader.
user_country	string	Yes	The country of the uploader.
vebose_audit	string	Yes	The verbosity of the audit.

Appendix 1: List of Messages and Error values

ID	Value
0	This file was uploaded previously and is pending test.
1	There was an error in uploading file.
2	The uploaded application is corrupted. Please double check file integrity.
4	The uploaded application is too large and cannot be processed.
5	The uploaded file is not a valid application. Please double check and try again.
6	Package not runnable.
7	No file provided for upload.
8	Error in upload.
9	Unable to create destination file.
10	Error saving file chunk.
20	Not logged in.
21	Unknown error, please try again.

22	Test can't be deleted.
25	Test cannot be refreshed. Test is locked.
26	Test cannot be refreshed because it's manually uploaded.
27	Test information cannot be refreshed because the test is not finished yet.
30	Error in deleting, please try again later.
32	This E-Mail has been already added.
33	Error adding e-mail to this test.
34	Error sending e-mail.
35	E-mail has been sent.
36	All is ok!
37	E-mail added to this test.
38	All fields must be numeric.
40	Sorry, our systems are very busy now, we are working on the issue. Please try again in a few minutes.
41	You have tried to perform N tests in the last 3 minutes. Please try again a bit later.
42	You have tried to perform N tests in the last 24 hours. Please try again a bit later.
43	Your IP is blacklisted.
44	Sorry, your API key is invalid or has expired. Please double-check it or contact us.
45	You reached the limit of N concurring running tests. Please wait until at least one of them is finished.
46	The application will be eligible for a re-scan after N hours.