Malware URL Lookup Exercise

For this exercise, we would like to see how you solve a coding challenge with architecting for the future in mind.

One of our key values in how we develop new systems is to start with simple implementations and progressively make them more capable, scalable, and reliable. You are encouraged to get something that meets the base requirements working ASAP, and then iterate to improve on it. We ask that you use a Git-based repository (Bitbucket, GitHub, etc.) to commit your updates. It's up to you how frequently you commit and what you decide to include in each push, but we are particularly curious about your development workflow and how you handle revision control.

Please also include some unit tests for your project, and detailed instructions on how to get the application up and running. Assume we know nothing about how it needs to be run. You can use any languages/technologies/platforms you like.

Here's what we would like you to build:

Malware URL lookup service
We have an HTTP proxy that is scanning traffic, looking for malware URLs. Before allowing HTTP connections to be made, this proxy asks a service that maintains several databases of malware URLs if the resource being requested is known to contain malware.

Write a small web service, preferably in Go or Python, that responds to GET requests where the caller passes in a URL and the service responds with some information about that URL. The GET requests would look like this:

`GET /v1/urlinfo/{resource_url_with_query_string}`

The caller wants to know if it is safe to access that URL or not. As the implementer, you get to choose the authorization, response format and structure. Please document the API in the README. These lookups are blocking users from accessing the URL until the caller receives a response from your service.

Give some thought to the following. Write-up the design, if you do not have time to code.

- The size of the URL list could grow infinitely, how might you scale this beyond the memory capacity of the system? Bonus if you implement this.
- The number of requests may exceed the capacity of this system, how might you solve that? Bonus if you implement this.
- What are some strategies you might use to update the service with new URLs? Updates may be as many as 5000 URLs a day with updates arriving every 10 minutes.