DoHClient Module Documentation

Introduction

This DNS over HTTPS (DoH) client is designed for reverse DNS lookups. It uses the Quad9 DoH endpoint by default and provides caching capabilities for the lookups to optimize repetitive queries.

Setup

Prerequisites

- Python 3.x
- `requests` library: You can install it using `pip install requests`.

Installation

1. Save the module in a directory accessible to your project.
2. Ensure the required libraries are installed.

How to Use

Basic Usage

1. **Initialization**

   First, create an instance of the `DoHClient` class:

   ```
   pythonCopy code
   client = DoHClient()
   ```

2. **Performing a Reverse DNS Lookup**

   Use the `reverse_lookup` method of the client:

```
pythonCopy code
domain = client.reverse_lookup("8.8.8.8")
print(domain)
```

Advanced Usage

Cache Configuration

The client uses a cache system to avoid redundant lookups. By default, the cache duration is set to 5 minutes, and it can store up to 100 entries. However, you can adjust these values:

- **Custom Cache Duration**: To set a custom cache duration (e.g., 10 minutes):

```
pythonCopy code
client = DoHClient(cache_duration=600)
```

- **Custom Cache Size**: To set a custom cache size (e.g., 200 entries):

```
pythonCopy code
client = DoHClient(max_cache_size=200)
```

Logging

The module uses Python's built-in logging system. To see detailed logs:

1. Setup logging in your main script:

```
pythonCopy code
import logging
logging.basicConfig(level=logging.DEBUG, format='%(asctime)s - %(name)s - %(levelnam
e)s - %(message)s')
```

2. Now, when you use the client, you'll see detailed logs on the console.

Understanding the Code

Cache System

The `Cache` class is responsible for caching the domain results to optimize lookup speed. It's designed as an LRU (Least Recently Used) cache. When the cache reaches its max size, it will evict the oldest entries. The cache also evicts entries that are stale, based on the duration you set.

DNS Query Formation

The module uses the `_build_dns_query` method to create a DNS query in binary format. This binary query is then sent to the DoH endpoint using the HTTP POST method.

DNS Response Parsing

The response from the DoH endpoint is in binary format. The `_parse_dns_response` method is responsible for parsing this binary response to extract the domain name.

Error Handling and Retries

The client includes error handling mechanisms for possible issues like timeouts, connection errors, and malformed responses. By default, it will retry the request 3 times if there's a connection error or timeout.