The case for the Robots Exclusion Protocol

Gary Illyes, Google Switzerland GmbH

Introduction

In 1994, Martijn Koster (a webmaster himself) came up with the idea of robots.txt after automatic clients (crawlers) were overwhelming his site. With more input from other webmasters, the Robots Exclusion Protocol was born, and it was adopted by search engines and other crawler operators to help website owners manage their server resources easier. It functioned as a de facto standard for over 20 years.

In 2022 the REP became a proposed standard under RFC 9309, and in 2024 a new I-D was submitted to the IETF to standardize the REP's resource level controls (REPext) which have been around since 1996.

Together, the REP and REPext offer service owners a simple and mature protocol to supply access directives to automatic clients, and how the fetched bytes may be used if at all. To service owners the REP and REPext provide a mature and well-tested, already highly adopted, and simple opt out mechanism that fits seamlessly in any system, be that traditional IR, ML training, or RAG.

These directives are honored by the vast majority of current crawler operators and thus robots.txt and its resource level counterparts are used by over 4 billion hosts Google knows of.

For crawler operators it's also an attractive way to let service owners opt out or otherwise control crawling. The protocol's simplicity, maturity and its high adoption rate, and flexibility allows crawler operators to offer opt out mechanisms in a format that's straightforward, extremely simple yet familiar to service owners

REP(ext) Adoption and Maturity

The REP alone is used by over 4 billion hosts that Google keeps track of in its robots.txt cache. The number of robots.txt files is linearly increasing with the number of new hosts suggesting continuous support for the protocol. The REPext is used by the vast majority of sites that Google indexes, and the number of HTTP or HTML robots directives shows an increase in usage.

The protocols themselves haven't changed substantially in syntax since their introduction. This offers high quality opt out data for crawlers. Moreover, there are well established robots.txt parsers released open source, such as Google's, which allows new crawler operators to quickly adopt the REP without the need to write new parsers.

REP(ext) Extensibility

While the current REP and REPext provide a limited number of directives, they allow for new directives provided the directives don't interfere with the standardized ones. For example, the robots.txt "sitemap: <URL>" record is not standardized but rather a record that search engines started to support in 2006.

This means that crawler operators may add support for new records in their parsers. For example, a new record for disallowing "ai crawl" in a certain URI path might look like

```
disallow-ai: /foo/bar/*/cheese
```

The value of the user-agent field is relatively loosely defined in RFC9309, which may also allow for extensibility. For example, while the "user-agent" record is supposed to hold a product token (crawler name) or a global token (*), it might be able to also hold a crawler category like "SEARCH" or "GENAI" which would target crawlers falling in that category:

```
user-agent: SEARCH
disallow: /foo/bar/*/cheese
```

Closing

The REP and REPext are working based on an honor system; they depend on crawler operators respecting them. This applies to every other file based crawl directive system as well, such as ads.txt. The REP won't ever function as an access authorization system, however due to its adoption and simplicity, it can continue to exist as a control mechanism for well behaved crawlers; for other crawlers, we will always have firewalls.