

**NAME****npm-hook** – Manage registry hooks**Synopsis**

```
npm hook ls [pkg]
npm hook add <entity> <url> <secret>
npm hook update <id> <url> [secret]
npm hook rm <id>
```

Note: This command is unaware of workspaces.

**Description**

Allows you to manage npm hooks <https://blog.npmjs.org/post/145260155635/introducing-hooks-get-notifications-of-npm>, including adding, removing, listing, and updating.

Hooks allow you to configure URL endpoints that will be notified whenever a change happens to any of the supported entity types. Three different types of entities can be watched by hooks: packages, owners, and scopes.

To create a package hook, simply reference the package name.

To create an owner hook, prefix the owner name with `~` (as in, `~youruser`).

To create a scope hook, prefix the scope name with `@` (as in, `@yourscope`).

The hook **id** used by **update** and **rm** are the IDs listed in **npm hook ls** for that particular hook.

The shared secret will be sent along to the URL endpoint so you can verify the request came from your own configured hook.

**Example**

Add a hook to watch a package for changes:

```
$ npm hook add lodash https://example.com/ my-shared-secret
```

Add a hook to watch packages belonging to the user **substack**:

```
$ npm hook add ~substack https://example.com/ my-shared-secret
```

Add a hook to watch packages in the scope **@npm**

```
$ npm hook add @npm https://example.com/ my-shared-secret
```

List all your active hooks:

```
$ npm hook ls
```

List your active hooks for the **lodash** package:

```
$ npm hook ls lodash
```

Update an existing hook's url:

```
$ npm hook update id-deadbeef https://my-new-website.here/
```

Remove a hook:

```
$ npm hook rm id-deadbeef
```

**Configuration**

```
<!-- AUTOGENERATED CONFIG DESCRIPTIONS START --> <!-- automatically generated, do not
edit manually --> <!-- see lib/utils/config/definitions.js -->
```

**registry**

- Default: "https://registry.npmjs.org/"
- Type: URL

The base URL of the npm registry. <!-- automatically generated, do not edit manually --> <!-- see lib/utils/config/definitions.js -->

**otp**

- Default: null
- Type: null or String

This is a one-time password from a two-factor authenticator. It's needed when publishing or changing package permissions with **npm access**.

If not set, and a registry response fails with a challenge for a one-time password, npm will prompt on the command line for one. <!-- automatically generated, do not edit manually --> <!-- see lib/utils/config/definitions.js -->

<!-- AUTOGENERATED CONFIG DESCRIPTIONS END -->

**See Also**

- "Introducing Hooks" blog post <https://blog.npmjs.org/post/145260155635/introducing-hooks-get-notifications-of-npm>