npm レジストリ URL と認証トークンの使用

重要ポイント

- npm レジストリ URL を使用するための認証トークンを設定するには、通常 .npmrc ファイルに設定し、トークンをレジストリ URL に関連付けることが推奨されています。
- 研究によると、標準的な方法は、.npmrc ファイルに //registry.url.com/:_authToken=your_token を追加することです。ここで、registry.url.com はレジストリ URL で、your_token は認証トークンです。
- 証拠は、認証トークンを URL に直接含める（HTTP リクエストなど）のではなく、.npmrc ファイルを使用することを示唆しています。

直接の回答

概要　npm レジストリ URL を使用して認証トークンを使用するには、通常、特別なファイルである .npmrc に設定します。このファイルは、npm コマンドラインツールに特定のパッケージレジストリ（公共の npm レジストリやプライベートレジストリ）へのアクセス時に認証する方法を教えます。以下に、初心者向けに簡単に設定する方法をステップバイステップで説明します。

設定手順

1. **レジストリ URL を見つける**: 使用するレジストリを決定します。例えば、registry.npmjs.org は公共の npm レジストリで、privateregistry.com はプライベートレジストリです。
2. **認証トークンを取得**: レジストリ提供者から認証トークンを取得します。これは、個人用アクセストークンや組織が提供する他の種類のトークンです。
3. **.npmrc ファイルを編集**: .npmrc ファイルを開いたり作成したりします。このファイルは、プロジェクトフォルダにあるプロジェクト固有の設定用、または Unix システムのホームディレクトリ（例：~/.npmrc）にあるユーザー全体の設定用です。
    - 以下のように行を追加します: //registry.url.com/:_authToken=your_token。ここで、registry.url.com をレジストリ URL に、your_token を実際のトークンに置き換えます。
    - 例えば、公共の npm レジストリの場合は、以下のようになります: //registry.npmjs.org/:_authToken=abc123。
4. **ファイルをセキュアにする**: .npmrc ファイルがあなただけが読み取りおよび書き込みできるようにして、トークンを安全に保護します。Unix システムでは、chmod 600 ~/.npmrc コマンドを使用して権限を設定します。
5. **動作を確認**: npm install などの npm コマンドを実行して、レジストリに問題なくアクセスできるか確認します。

予期せぬ詳細　URL に直接認証トークンを入れる（例：https://registry.url.com?token=your_token）と考えるかもしれませんが、これは npm の標準的な方法ではありません。代わりに、npm は .npmrc ファイルを使用して、認証を背景で処理し、より安全で管理しやすくします。

詳細については、ここの公式 npm ドキュメントの .npmrc ファイルの設定についてを参照してください。

調査ノート: npm レジストリ URL での認証トークンの使用に関する詳細な探求

このセクションでは、npm レジストリ URL で認証トークンを使用する方法について、直接の回答に追加のコンテキスト、技術的な詳細、例を含めて包括的な分析を提供します。これにより、異なるレベルの専門知識を持つユーザーがすべての討論された側面を理解することができます。

npm と認証の紹介　Node Package Manager (npm) は、JavaScript 開発者にとって重要なツールで、パッケージと依存関係を管理します。npm は、公共のレジストリ（例：registry.npmjs.org）や、組織がホストするプライベートレジストリとやり取りします。プライベートレジストリやパッケージの公開などの特定のアクションには、通常認証が必要であり、これは設定ファイルに保存された認証トークンを通じて処理されます。

.npmrc ファイルは、npm の設定の核心であり、レジストリ URL、認証方法などの設定をカスタマイズすることができます。このファイルは、プロジェクトごと（プロジェクトルート）、ユーザーごと（ホームディレクトリ、例：~/.npmrc）、またはシステム全体（例：/etc/npmrc）に存在することができます。このファイルは INI 形式を使用し、キーと値が npm の動作方法を定義します。

.npmrc での認証トークンの設定　特定のレジストリ URL とトークンを関連付けるには、.npmrc ファイルを設定して、トークンをそのレジストリに関連付けます。標準的な形式は以下の通りです。

```
registry.url.com/:_authToken=your_token
```

ここで、registry.url.com はレジストリのベース URL（例：registry.npmjs.org は公共レジストリ、privateregistry.com はプライベートレジストリ）で、your_token はレジストリによって提供される認証トークンです。:_authToken キーは、これはトークンベースの認証であることを示し、npm は HTTP リクエストをレジストリに送信する際に Authorization ヘッダーを Bearer your_token として設定します。

例えば、公共の npm レジストリに対してトークン abc123 を持っている場合、.npmrc のエントリは以下のようになります。

```
registry.npmjs.org/:_authToken=abc123
```

この設定により、registry.npmjs.org に対するすべての npm コマンドは、認証のためにトークンを含むようになり、トークンのスコープによってはプライベートパッケージへのアクセスや公開機能が可能になります。

スコープ付きパッケージの処理　スコープ付きパッケージ（例：@mycompany/mypackage）については、そのスコープに対して異なるレジストリを指定することができます。まず、スコープのレジストリを設定します。

```
@mycompany:registry=https://mycompany.artifactory.com/
```

次に、そのレジストリにトークンを関連付けます。

```
mycompany.artifactory.com/:_authToken=your_token
```

この設定により、すべての `@mycompany` パッケージに対するリクエストは指定されたプライベートレジストリにルーティングされ、提供されたトークンを使用して認証されます。これは、組織が内部パッケージのために自社の npm レジストリをホストする企業環境で特に有用です。

`.npmrc` の場所とセキュリティ `.npmrc` ファイルは、以下の場所に存在することができます。

- **プロジェクトごと**: プロジェクトルートにあります（例：`package.json` の横）。これは、プロジェクト固有の設定とグローバル設定の上書きに最適です。
- **ユーザーごと**: ユーザーのホームディレクトリにあります（例：Unix の `~/.npmrc`、Windows の `C:\Users\<Username>\.npmrc`）。これは、そのユーザーのすべての npm 操作に影響を与えます。
- **グローバル**: `/etc/npmrc` にあり、または `globalconfig` パラメータによって指定され、通常はシステム全体の設定に使用されます。

`.npmrc` ファイルには、認証トークンなどの機密情報を含むことが多いため、セキュリティは重要です。ファイルは、所有者のみが読み取りおよび書き込みできるようにする必要があります。Unix システムでは、`chmod 600 ~/.npmrc` コマンドを使用して、所有者のみが読み取りおよび書き込みできるように設定します。

代替の認証方法 トークンベースの認証は一般的ですが、npm は基本認証もサポートしており、`.npmrc` ファイルにユーザー名とパスワードを含めることができます。

```
registry.url.com/:username=your_username
registry.url.com/:_password=your_password
```

しかし、セキュリティの観点から、トークンベースの認証が推奨されます。トークンは取り消し可能であり、スコープ付きの権限を持つため、平文のパスワードを保存するよりもリスクが低いです。

URL に直接含めることは可能ですか？ 「npm レジストリ URL に auth または authtoken を使用する」という質問は、トークンを URL に直接含めることを示唆しているかもしれません（例：`https://registry.url.com?token=your_token`）。しかし、研究によると、これは npm の標準的な方法ではありません。npm レジストリ API ドキュメントや関連リソース（例：NPM レジストリ認証 | Rush）は、認証のために `.npmrc` ファイルを使用し、トークンを `Authorization` ヘッダーとして `Bearer your_token` として渡すことを強調しています。

URL にトークンをクエリパラメータとして含めることは、標準の npm レジストリではサポートされておらず、動作しない可能性があります。HTTP ヘッダーレベルで認証が処理されるためです。一部のプライベートレジストリではカスタム URL ベースの認証をサポートしているかもしれませんが、これは公式の npm レジストリではドキュメント化されていません。例えば、基本認証では URL のような `https://username:password@registry.url.com` を許可していますが、これはトークンベースの方法に比べて非推奨であり、セキュリティが低いです。

実用例と使用例　以下に、いくつかのシナリオを示します。

- **公共レジストリでトークンを使用**: 公共の npm レジストリにパッケージを公開する必要がある場合、以下のように追加します。

  ```
  registry.npmjs.org/:_authToken=abc123
  ```

  そして、`npm publish` を実行してパッケージをアップロードし、npm は認証のためにトークンを使用します。

- **スコープ付きパッケージのプライベートレジストリ**: 会社が `@company` パッケージのためにプライベートレジストリを `https://company.registry.com` にホストしている場合、以下のように設定します。

  ```
  @company:registry=https://company.registry.com/
  company.registry.com/:_authToken=def456
  ```

  これで、`@company/mypackage` をインストールすると、プライベートレジストリに対してトークンを使用して認証されます。

- **CI/CD 統合**: CI 環境では、トークンを環境変数（例：`NPM_TOKEN`）として保存し、`.npmrc` ファイルで動的に使用します。

  ```
  registry.npmjs.org/:_authToken=${NPM_TOKEN}
  ```

  このアプローチは、Using private packages in a CI/CD workflow | npm Docs に詳細があり、トークンがハードコードされず、セキュアです。

トラブルシューティングとベストプラクティス　認証が失敗した場合は、以下を確認してください。

- レジストリ URL が正しく、アクセス可能である。
- トークンが有効であり、必要な権限（例：インストールには読み取り、公開には書き込み）を持っている。
- `.npmrc` ファイルが正しい場所にあり、適切な権限を持っている。

ベストプラクティスには以下が含まれます。

- トークンを含む `.npmrc` をバージョン管理にコミットしない。`.gitignore` に追加してください。
- CI/CD パイプラインで環境変数を使用してトークンを保存し、セキュリティを強化します。
- トークンを定期的に回転させ、使用されていないトークンを無効にしてリスクを最小限に抑えます。

認証方法の比較分析　以下の表は、npm でのトークンベース認証と基本認証を構造化した概要を提供します。

方法	.npmrc の設定	セキュリティ	使用例
トークンベース（推奨）	`registry.url.com/:_authToken=your_token`	高（取り消し可能、スコープ付き）	プライベートレジストリ、CI/CD
基本認証	`registry.url.com/:username=your_username registry.url.com/:_password=your_password`	低（平文パスワード）	レガシーシステム、公共レジストリ

この表は、特に現代的なワークフローにおいてトークンベースの認証が推奨される理由を示しています。

結論 npm レジストリ URL を使用して認証トークンを使用する主な方法は、.npmrc ファイルを設定してトークンをレジストリに関連付けることです。この方法は標準的であり、セキュアで、公式の npm レジストリとの相互作用に広くサポートされています。URL にトークンを直接含めることは、npm の標準的な方法ではありませんし、動作しない可能性があります。.npmrc メソッドは、公共およびプライベートレジストリの相互作用に対して強力なソリューションを提供します。

さらに詳しくは、configuring npmrc と registry authentication に関する公式 npm ドキュメントを参照してください。

主要な引用

- NPM レジストリ認証 Rush ドキュメント
- registry-auth-token npm パッケージの詳細
- Globally configure NPM with a token Stack Overflow 質問
- Configure authentication Artifact Registry Google Cloud
- 公式 npm ドキュメントの npmrc ファイル
- Using private packages in CI/CD workflow npm Docs