APT JSON Hookを試してみた話

更新したらまずいパッケージを回避したい

Kentaro Hayashi

ClearCode Inc.

2023年11月 東京エリア・関西合同Debian勉強会

スライドはRabbit Slide Showにて公開済みです

- APT JSON Hookを試してみた話
  - https://slide.rabbit-shocker.org/authors/kenhys/tokyodebian-apt-json-hook-202311

元となる記事も公開済み

- 参考: Apt 1.6で導入されたJSONフックを活用する方法
  - https://www.clear-code.com/blog/2023/10/19/use-apt-json-hook.html

本日の内容

- APT JSON Hookを試してみた話
  - APT JSON Hookはどういうものなのか
  - シェルスクリプトによるフックの説明
    - Rubyによるフックのサンプルも
  - Hookを活用するアイデアについて
    - 最近あった致命的なバグの事例

APT JSON Hookとは(1)

- 参考(元ネタ): https://gihyo.jp/admin/serial/01/ubuntu-recipe/0676
  - 「Ubuntu Weekly Recipe 第676回aptコマンドの最新機能あれこれ」
    - 「Apt CLIの操作結果をJSON RPCで受け取る」にて言及あり

APT JSON Hookとは(2)

- JSON-RPCを使って、APT実行中に任意の処理を割り込み実行させるしくみ
  - 割り込み処理の結果、APTの挙動を変えるものではない
    - フック処理とはハンドシェイクするぐらい
      →対応プロトコルバージョンを返すことしか想定されていない
    - ❌フックで処理した結果をAPTにJSON-RPC経由で伝える
    - ✅フックで通知された情報をもとになにかする

APT JSON Hookの仕様

- 参考: https://salsa.debian.org/apt-team/apt/-/blob/main/doc/json-hooks-protocol.md
  - リクエストとレスポンスはJSON-RPC 2.0に準拠
  - 最新プロトコルバージョンは0.2
  - UNIXドメインソケット(ファイルディスクリプタが渡される)でAPTとやりとり

APT JSON Hookの利用方法

- フックの設定ファイルを配置
  - 例: /etc/apt.conf.d/99apt-json-hook
    - AptCli::Hooks::<name>と対応するコマンドのパスを記載 (<name>はInstallとかUpgradeとか)
- 設定ファイルで言及したパスに仕様に準拠したコマンドを配置
  - 例: /usr/local/bin/apt-json-hook
    - APTと通信するフックスクリプト

フックを設定する方法

- 例: /etc/apt.conf.d/99apt-json-hookに記載する内容
  - ✅実行するコマンドを配置するパスは任意で良い
  - ✅Search,Install,UpgradeといったAPTのサブコマンドに対応したフックを指定できる

```
AptCli::Hooks::Upgrade:: "/usr/local/bin/apt-json-hook";
```

test-apt-cli-json-hooks サンプル

```
#!/bin/bash
trap '' SIGPIPE
while true; do
    read request <&$APT_HOOK_SOCKET || exit 1 # UNIXドメインソケットから読み込み
    if echo "$request" | grep -q ".hello"; then
        echo "HOOK: HELLO"
    fi
    if echo "$request" | grep -q ".bye"; then
        # {"json-rpc":"2.0", "method": "org.debian.apt.hooks.bye", ...}なら終了"
        echo "HOOK: BYE"
        exit 0;
    fi
    echo HOOK: request $request
    read empty <&$APT_HOOK_SOCKET || exit 1 # {"json-rpc":"2.0", "method": ...}\n + \n が送られてくるので空行を読み飛ばす
    echo HOOK: empty $empty
    if echo "$request" | grep -q ".hello"; then
        # 対応するプロトコルバージョンを返す
        printf '{"jsonrpc": "2.0", "result": {"version": "'$TEST_HOOK_VERSION'"}, "id": 0}\n\n' >&$APT_HOOK_SOCKET 2>/dev/null || exit 1
    fi
done
```

- 参考: https://salsa.debian.org/apt-team/apt/-/blob/main/test/integration/test-apt-cli-json-hooks#L29-67

サンプルスクリプトの挙動の説明

- APTから.helloがきたら対応しているプロトコルバージョンを返す
- APTから.byeがきたらスクリプトを終了
- requestにはmethodに応じてパッケージの情報が含まれている

Rubyで書いてみると

```ruby
IO.open(ENV['APT_HOOK_SOCKET'].to_i, "a+") do |io|
  while true do
    request = io.gets
    io.gets
    json = JSON.parse(request)
    method = json["method"]
    if method.end_with?(".bye")
      exit 0
    end
    if method.end_with?(".hello")
      io.puts({"jsonrpc" => "2.0", "result" => {"version" => "0.2"}, "id" => 0}.to_json)
      io.puts
    end
    # ここでなにか処理をする
  end
end
```

フックをどう活用するか?

- Debian sidを常用するけど、やばいバグは踏みたくない
  - 例: GRUB2 2.12~rc1-7 prevent machine to boot
    - https://bugs.debian.org/cgi-bin/bugreport.cgi?bug=1051271

GRUB2 2.12~rc1-7 prevent machine to boot

- https://bugs.debian.org/cgi-bin/bugreport.cgi?bug=1051271
  - タイトルの通り起動しなくなる **critical** バグです
  - 9/5 早朝にunstableに投入された
  - バグ報告されたのが9/5 23:21:02(日本時間)
    - 9/5 日中にアップデート後、夕方シャットダウンしていると次回起動で必ず踏む

やばいバグを回避できないか?

- ✅アップグレードしようとした時点でバグ登録されている
  - 登録されているバグをチェックすればいい?
    - **今日焦点をあてるのはこちら**
- ❌バグ登録されるタイミングが遅ければ不可避
  - Unattended-Upgrade::Package-Blacklistも併用するのがよい
    - See /etc/apt/apt.conf.d/50unattended-upgrades

更新とフックのタイミング

- JSON-RPCのメソッドの種類(アップグレード時)
  - org.debian.apt.hooks.install.pre-prompt
  - org.debian.apt.hooks.install.package-list
  - org.debian.apt.hooks.install.statistics

更新とフックのタイミング(2)

- **org.debian.apt.hooks.install.pre-prompt**
  - 「アップグレードパッケージを検出しています...完了」といったメッセージのあとにフックが実行
- org.debian.apt.hooks.install.package-list
  - 「以下のパッケージはアップグレードされます」といったメッセージのあとにフックが実行

更新とフックのタイミング(3)

- org.debian.apt.hooks.install.pre-prompt
  - 「アップグレードパッケージを検出しています...完了」といったメッセージのあとにフックが実行
- **org.debian.apt.hooks.install.package-list**
  - 「以下のパッケージはアップグレードされます」といったメッセージのあとにフックが実行

更新とフックのタイミング(4)

- **org.debian.apt.hooks.install.statistics**
  - 「アップグレード: 2 個、新規インストール: 0 個、削除: 0 個、保留: 1 個」といったメッセージのあとにフックが実行

バグで泣かないためにできること

- 最終的なY/nを指示する前という意味ではどのフックも大差ない
- .statisticsのタイミングで登録されているバグを検索
  - seriousやcriticalなバグがないか調べる
  - マッチするバグがあればメッセージを表示する

APTはどんな情報をくれるのか？

- 現在インストールしているパッケージとそのバージョン
- アップグレード候補のパッケージとそのバージョン
- アップグレードにてインストールされるはずのパッケージとそのバージョン

APTはどんな情報をくれるのか？

- 現在インストールしているパッケージとそのバージョン
- アップグレード候補のパッケージとそのバージョン
- アップグレードにてインストールされるはずのパッケージとそのバージョン

具体的なもらえる情報のサンプルは?

- {“json-rpc”:”2.0”, … “params”{ “packages”: ….}}に詰めたものがもらえる


```
[{"id"=>3790,
  "name"=>"at-spi2-common",
  "architecture"=>"amd64",
  "mode"=>"upgrade",
  "automatic"=>true,
  "versions"=>
   {"candidate"=>
     {"id"=>1455,
      "version"=>"2.49.91-2",
      "architecture"=>"all",
      "pin"=>500,
      "origins"=>[{"archive"=>"unstable", "codename"=>"sid", "origin"=>"Debian", "label"=>"Debian", "site"=>"deb.debian.org"}]},
    "install"=>
     {"id"=>1455,
      "version"=>"2.49.91-2",
      "architecture"=>"all",
      "pin"=>500,
      "origins"=>[{"archive"=>"unstable", "codename"=>"sid", "origin"=>"Debian", "label"=>"Debian", "site"=>"deb.debian.org"}]},
    "current"=>{"id"=>71628, "version"=>"2.49.91-1", "architecture"=>"all", "pin"=>100, "origins"=>[]}}},]
```

どうやって登録されているバグを検索するか

- Use UDD!
  - 参考: https://wiki.debian.org/UltimateDebianDatabase
  - 一般向けに UDD mirrorも公開されているよ
    - psql "postgresql://udd-mirror:udd-mirror@udd-mirror.debian.net/udd"
    - たまに接続できないこともある 😢

どうやって登録されているバグを検索するか(2)

- やばげなバグは？
  - bugsテーブルのseverityカラムを参照する (serious, critical, ...)
- 該当するパッケージを絞るには？
  - bugs_packagesテーブルのpackageカラムを参照する
- 該当するバージョンで絞り込むには?
  - bugs_found_inテーブルのversionを参照する

ざっくりとしたクエリの例

```
SELECT * FROM bugs LEFT JOIN bugs_packages ON bugs.id=bugs_packages.id
              LEFT JOIN bugs_found_in on bugs_packages.id=bugs_found_in.id
              WHERE bugs.severity IN ('serious', 'critical') AND
              (bugs_packages.package = 'パッケージ名'
                AND bugs_found_in.version LIKE '%' || '対象バージョン' || '%' )
```

もしあのときすでにバグ登録されていたら回避できたか？

- 実験の条件
  - sidのコンテナイメージにフックを設定済み
  - バグの検索はarchivedを対象にする(もうすでに修正済みのため)
  - **grub2 2.06-13からのアップグレード**

もしあのときすでにバグ登録されていたら回避できたか？(2)

- 実験の手順
  - snashot.debian.orgのアーカイブを利用し 2023/09/04の時点に巻き戻す
    - https://snapshot.debian.org/archive/debian/20230904T000000Z
  - 問題のなかったgrub2 2.06-13をインストール
  - **snashot.debian.orgのアーカイブを利用し 2023/09/05のメタデータを使ってアップグレード**
    - https://snapshot.debian.org/archive/debian/20230905T120000Z

実験してみる

- 注:
  - snapshot.d.oへのアクセスは時間がかかるので、grubインストールまでは実施ずみ

さいごに

- APTから渡されるのはバイナリパッケージ名なので、ソースパッケージ名も含めてバグを検索しないとヒットしない
- APTのフックにより既知の致命的なバグの回避は限定的ながら有効そう
  - ✅UDDのバグ登録データを利用すると事前に機械的なチェックが実現可能👍
  - ✅Debian sidを常用しても怖くなくなる(かもしれない)
  - ❌未登録なバグはどうしても不可避(今回の範囲外)