FERMILAB-POSTER-23-222-STUDENT

Beholder: Updating and Maintaining Cybersecurity Software

Lucas D’Antonio - Purdue University Northwest, Under the Mentorship of Jason Ormes & Jeny Teheran

Security and Emergency Management Division, Cybersecurity Team, Fermi National Accelerator Laboratory, Batavia, Illinois 60510

Understanding Beholder

Beholder is the main engine for Fermilab’s cybersecurity defense systems, which integrates detectors, vulnerability scanners, and blocking mechanisms. Beholder takes the output from diverse sensors, routes appropriate actions to protect the network against existing threats and integrates with blocking and reporting mechanisms. Beholder is written on the Ruby on Rails web framework. It currently runs on Rails 6.0 which has reached its end-of-life for security updates. Ruby utilizes a package manager referred to as “gems”. These gems allow a Ruby program to interface in a multitude of ways not possible in the programming language, from web integration to support for SQL databases.

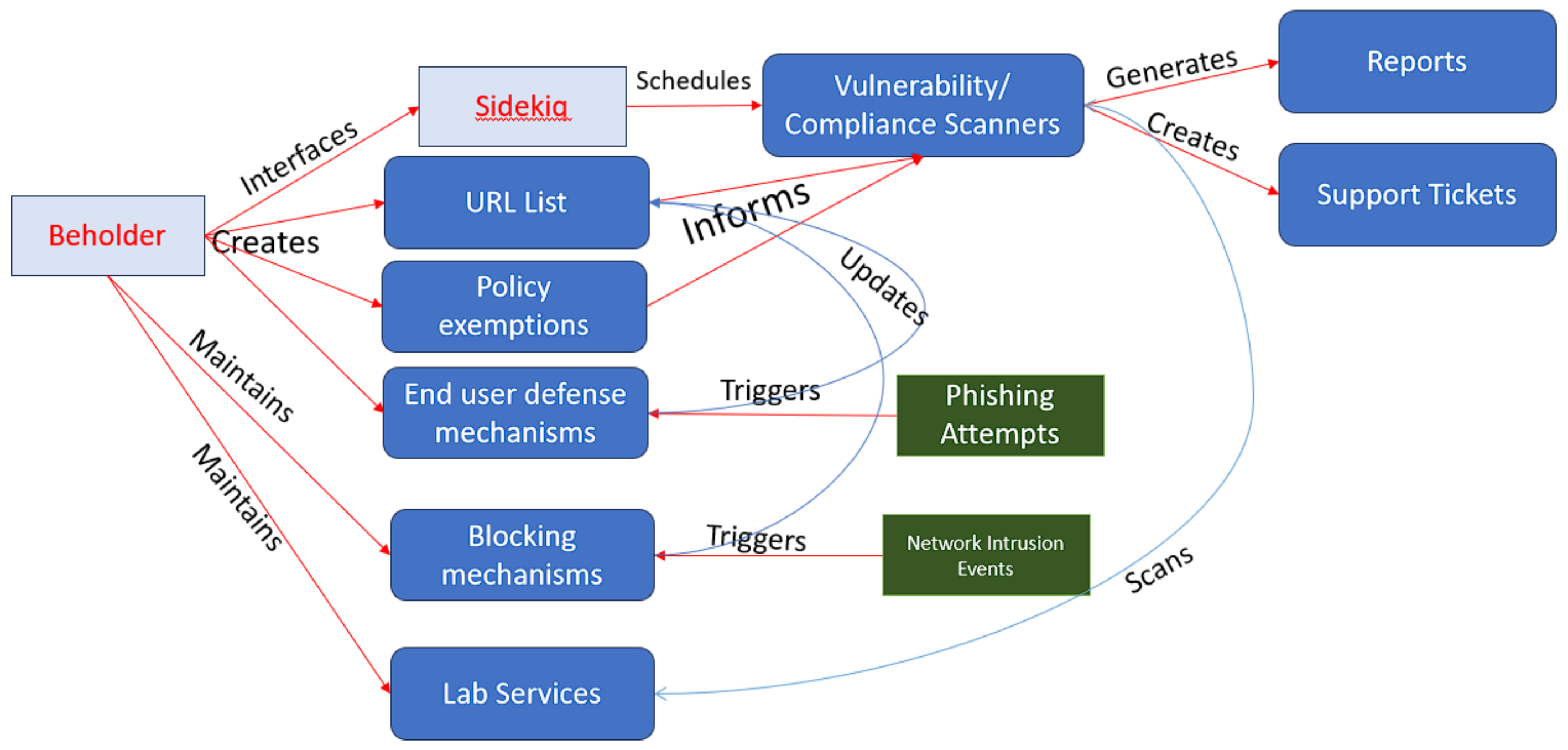


Figure 1 – Beholder’s Workflow. Jobs are scheduled through Sidekiq to scan end devices on Fermilab’s network, and the results are returned and stored in datatables using Redis.

As major features of Ruby on Rails are released, older versions of the gems and their dependencies become deprecated, leaving them vulnerable to unpatched exploits at a future date. At the beginning of the upgrade process from Rails 6.0, an active vulnerability existed in the Sidekiq job scheduler gem which was at a version which was dependent on Rails 6.0 to work. This showed that in addition to changing the Rails version Beholder ran on, other gems would need to be updated as well.

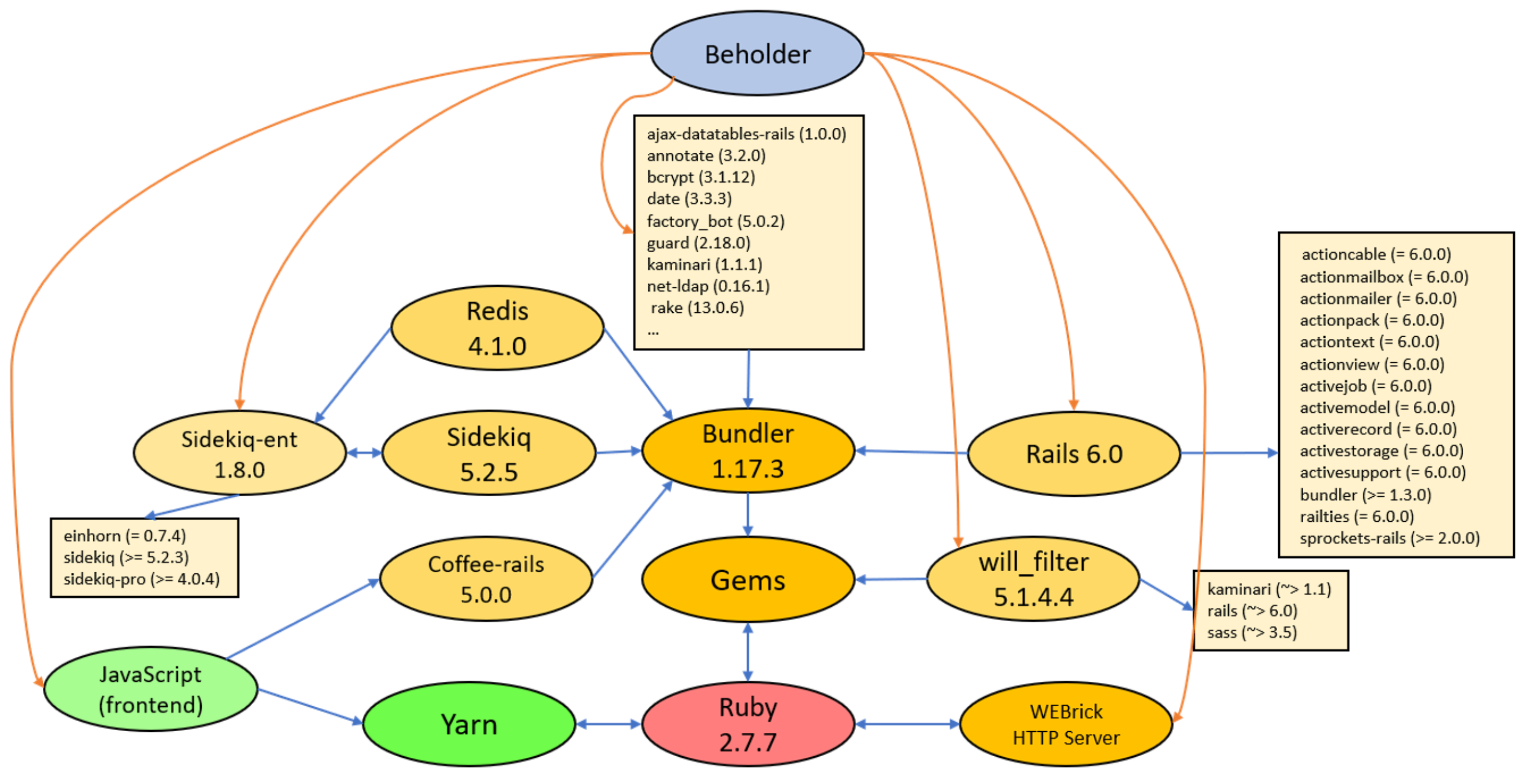


Figure 2 - Beholder’s dependencies on Rails 6.0 before any upgrades. The Bundler gem maintains most of the gems installed on Beholder, with the notable exception of will_filter, which is an internally maintained gem. This is a non-exhaustive diagram.

Upgrading Dependencies

A copy of Beholder was first deployed in a sandbox computer environment. Then, updates were made to the configuration files of Rails to accommodate a version upgrade. Using the Ruby gem “bundler”, one can simply switch the version of Rails (or any other gem) by editing the list of installed gems found in the Gemfile and running a command to make the installed gems compliant. Once this was done, the version of Rails had been changed. Bundler will also change the gem version number for any other dependencies where necessary.

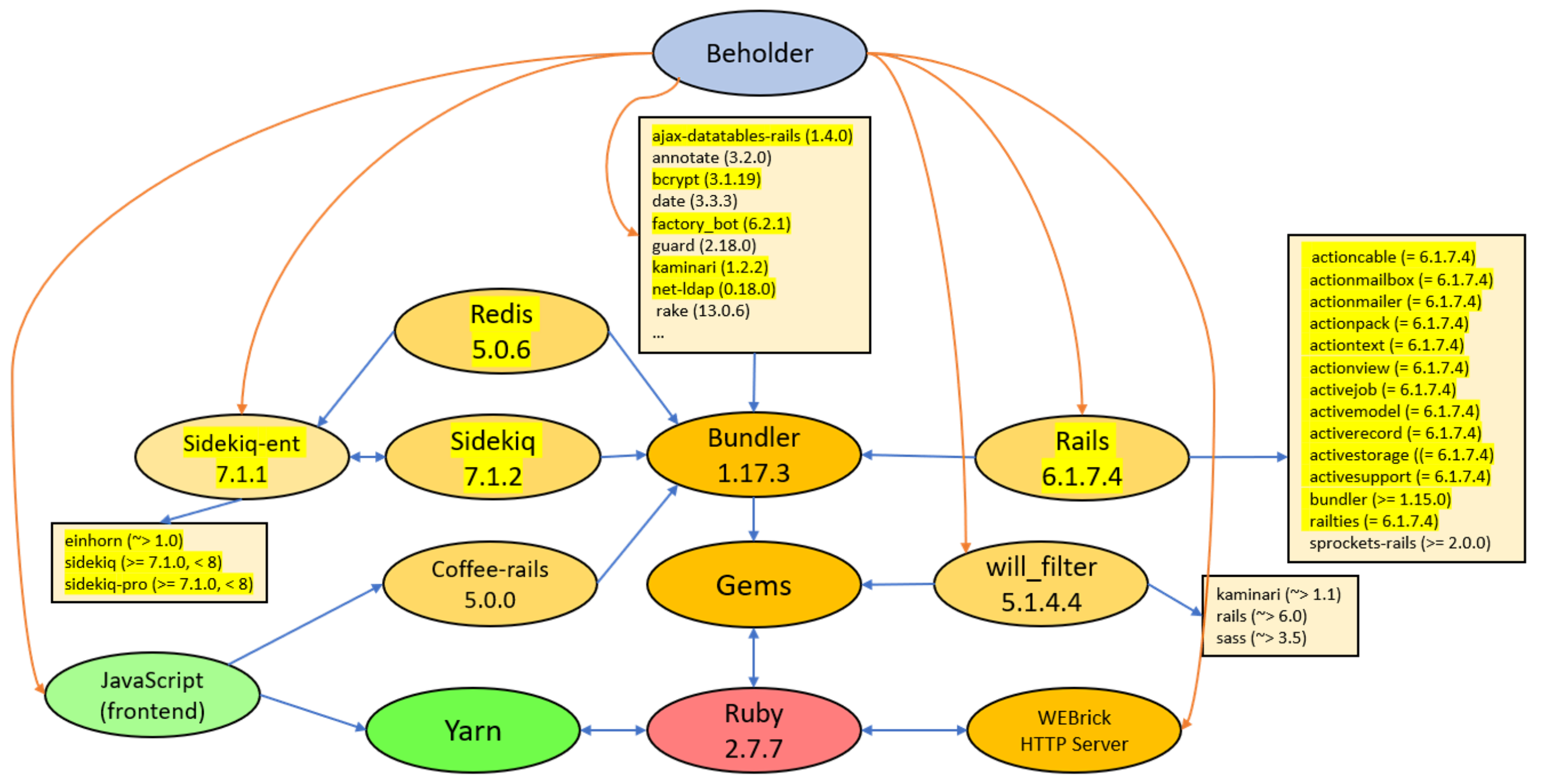
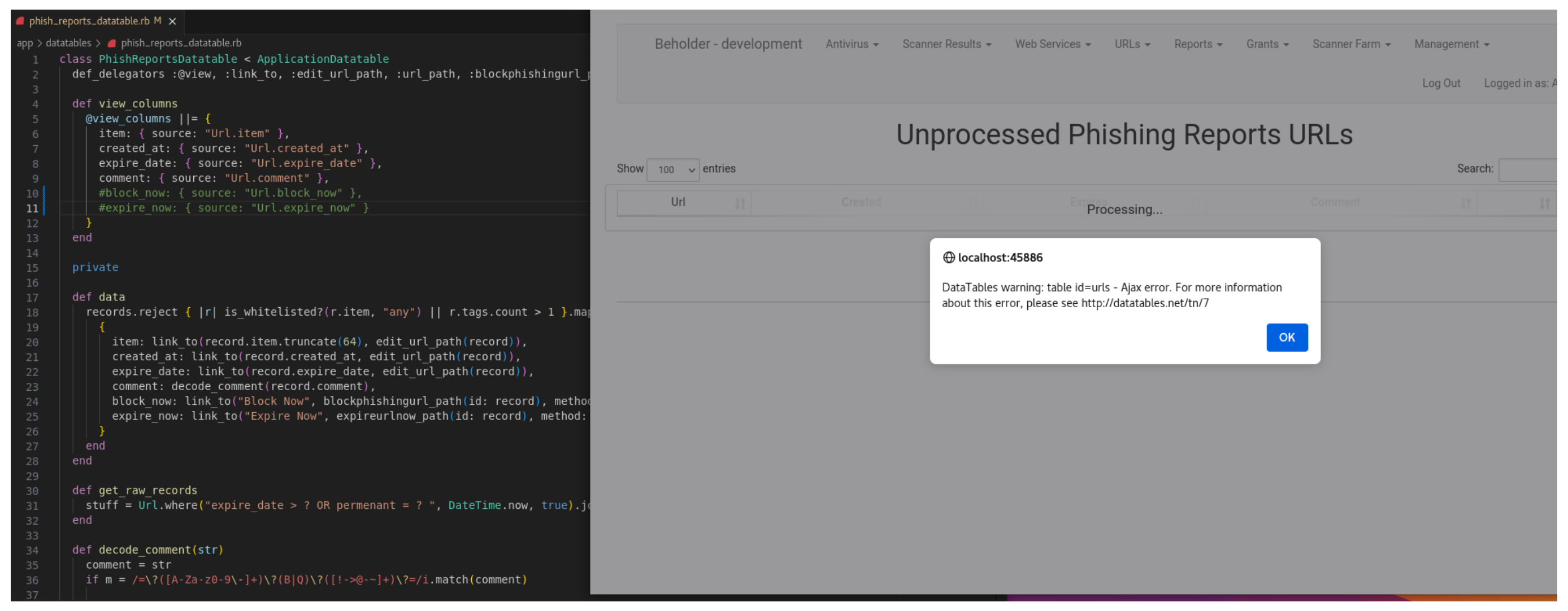


Figure 3 – Beholder after upgrading to Rails 6.1.7.4. Changes have been highlighted. Not every dependency required an update at this time.

At this time, deprecations must be removed for the application to start. Rails 6.1 removed several outdated methods in use on Beholder for establishing transport layer security. These were made redundant and were simply removed. In addition, several gems now required their current version to be pointed to in the Gemfile such as Redis, the database used by the Sidekiq job scheduler and ajax-datatables-rails. Upgrading Rails brought Beholder back into a window of security support by the maintainers of Ruby.

While the application may be running, certain features may no longer work properly due to deprecations between the gem version changes. There are two ways to address this: manual testing in the sandbox environment and Rspec testing. Using Beholder normally can help uncover errors not immediately apparent in the code due to a deprecation. By simply attempting to complete tasks in Beholder and observing the results, fixes can be deployed so the program can resume normal operation. Rspec testing in contrast, automates this process by creating a series of scripts written in Ruby test individual features of Beholder. Once testing of the new version of Beholder has demonstrated satisfactory performance, it can be deployed into production. Using the version control system git, a copy of the changes made to Beholder could then be deployed into the production environment.



```
1   class PhishReportsDatatable < ApplicationDatatable
2     def_delegators :@view, :link_to, :edit_url_path, :url_path, :blockphishingurl_
3
4     def view_columns
5       @view_columns ||= {
6         item: { source: "Url.item" },
7         created_at: { source: "Url.created_at" },
8         expire_date: { source: "Url.expire_date" },
9         comment: { source: "Url.comment" },
10        #block_now: { source: "Url.block_now" },
11        #expire_now: { source: "Url.expire_now" }
12      }
13    end
14
15    private
16
17    def data
18      records.reject { |r| is_whitelisted?(r.item, "any") || r.tags.count > 1 }.ma
19        {
20          item: link_to(record.item.truncate(64), edit_url_path(record)),
21          created_at: link_to(record.created_at, edit_url_path(record)),
22          expire_date: link_to(record.expire_date, edit_url_path(record)),
23          comment: decode_comment(record.comment),
24          block_now: link_to("Block Now", blockphishingurl_path(id: record), metho
25          expire_now: link_to("Expire Now", expireurlnow_path(id: record), method:
26        }
27      end
28    end
29
30    def get_raw_records
31      stuff = Url.where("expire_date > ? OR permenant = ? ", DateTime.now, true).j
32    end
33
34    def decode_comment(str)
35      comment = str
36      if m = /=\?([A-Za-z0-9\-]+)\?(B|Q)\?([!->@-~]+)\?=/i.match(comment)
37
```

Unprocessed Phishing Reports URLs

localhost:45886

DataTables warning: table id=urls - Ajax error. For more information about this error, please see http://datatables.net/tn/7

Figure 4 – Error encountered when attempting to view unprocessed phishing report URLs. Two lines (10 & 11) were deleted from the file that manages the view on the webpage. It was determined that bundler had deleted these lines, creating the error seen on the right.

Ongoing Maintenance

Upgrading Beholder to Rails 6.1 was a straightforward task, albeit with some challenging aspects to make all the features work properly. This served as an exercise in streamlining the process for maintaining Beholder for years to come. By increasing Rspec test coverage in addition to documenting the upgrade process (including resources that were used to assist in the upgrading process), a future Ruby on Rails upgrade to Rails 7 for Beholder should run smoothly for the cybersecurity staff at Fermilab.

This work was produced by Fermi Research Alliance, LLC under Contract No. DE-AC02-07CH11359 with the U.S. Department of Energy. Publisher acknowledges the U.S. Government license to provide public access under the DOE Public Access Plan.

This research was supported in part by the U.S. Department of Energy (DOE), Omni Technology Alliance Internship Program. The program is championed by the DOE’s Office of Chief Information Officer (OCIO) and represents a partnership with the leadership of the Office of Economic Impact and Diversity, the Office of Science, the Office of Nuclear Energy, and the National Nuclear Security Agency. The program is administered by the Oak Ridge Institute for Science and Education

Oak Ridge Institute for Science and Education

DOE Omni Technology Alliance Internship Program

de omnibus dubitandum

Be suspicious of everything

PNW

Purdue University Northwest

Fermi National Accelerator Laboratory

Fermilab

U.S. Department of Energy