1. Dynamic Patching

1.1. Django-19844 (CVE-2019-19844)

Link:
https://github.com/django/django/commit/5b1fbcef7a8bec991ebe7b2a18b5d5a95d72cb70

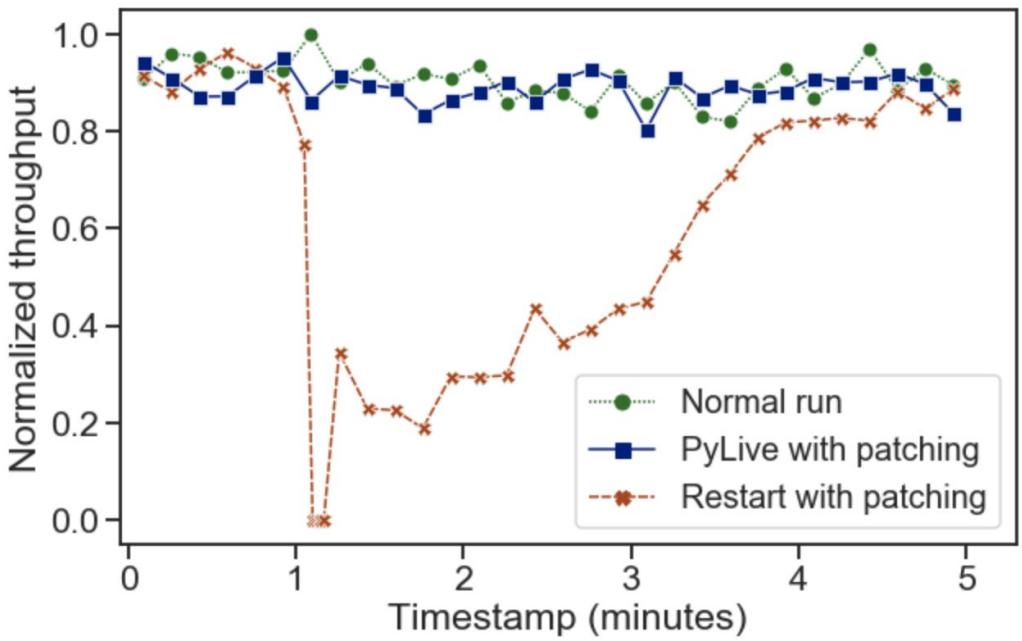
Description:

This patch fixes a security vulnerability, which potentially allows attackers to steal customers' accounts by sending password reset requests. For example, attackers can use an email account mike@ıxample.org to request password reset for a similar account mike@ixample.org and then reset the victim's password.

This is an urgent bug because it can be exploited to steal accounts and personal information.

To fix it, we can make a patch on the fly, which includes a new-defined function and modification to a method body.

Result:



1.2. Oscar_3264

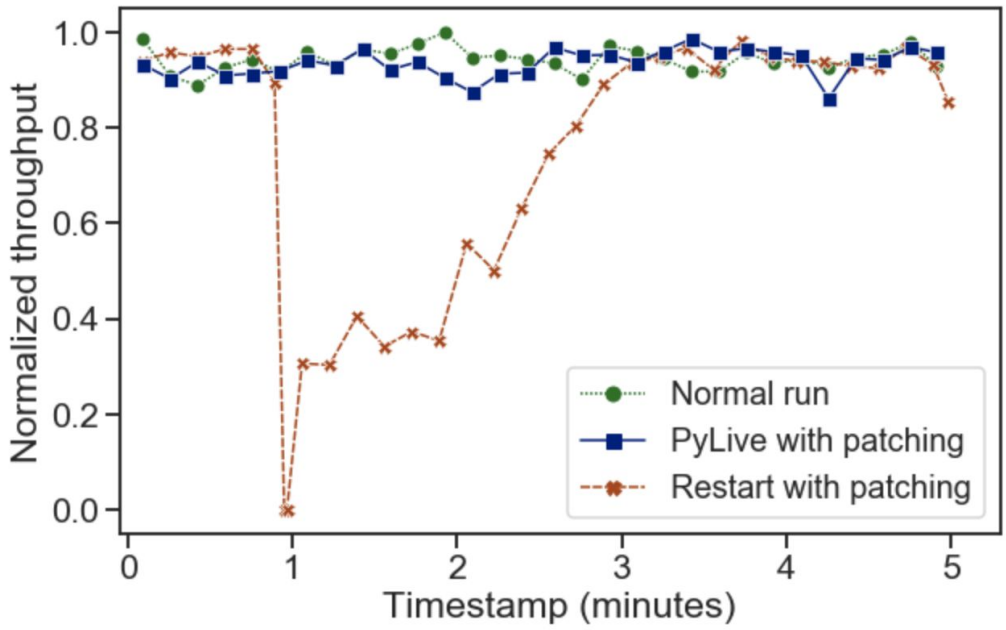
Link: https://github.com/django-oscar/django-oscar/pull/3264

Description:

This bug has the same symptoms as last one. But the difference is that it is not located in Django, instead it's because the save() method in Oscar is not well-written.

To fix it, we make a patch on the fly that modifies the method body of oscar.apps.customer.forms.PasswordResetForm.save().

Result:



1.3. Saleor_4879

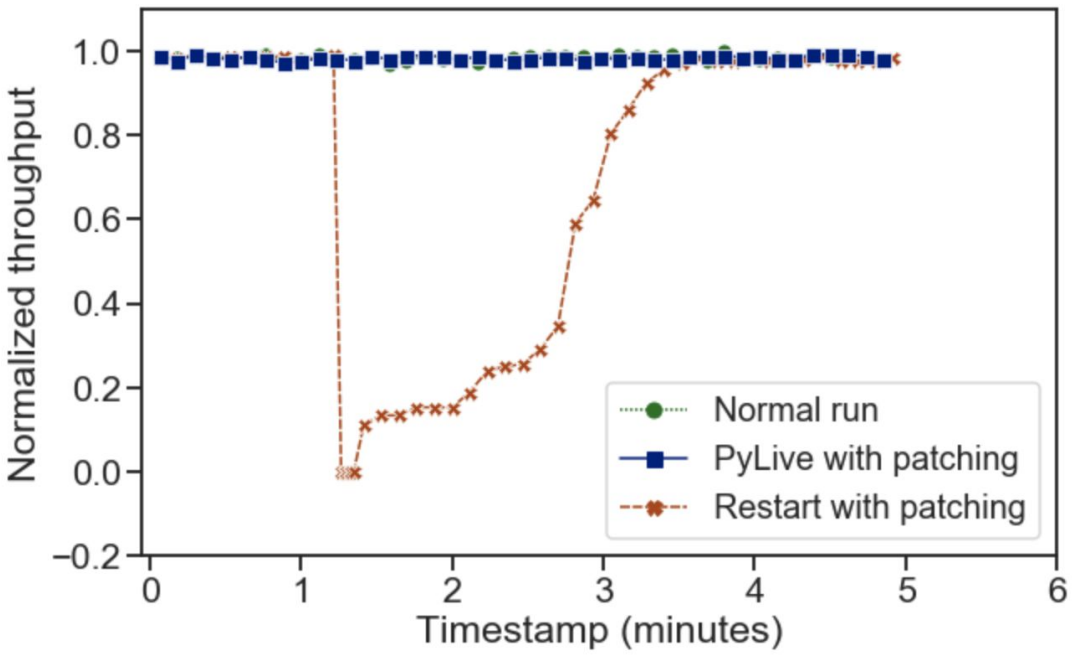
Link: https://github.com/mirumee/saleor/issues/4879

Description:

This is an urgent payment-related issue since users are unable to pay in production. This is because changing the payment gateway from sandbox mode to production mode failed.

In order to patch this issue, we use PyLive to patch two functions in two modules: function "BasePlugin.get_payment_config" in module “saleor.extensions.base_plugin” and function "AvataxPlugin._initialize_plugin_configuration” in module “saleor.extensions.plugins.avatax.plugin”.

Result:



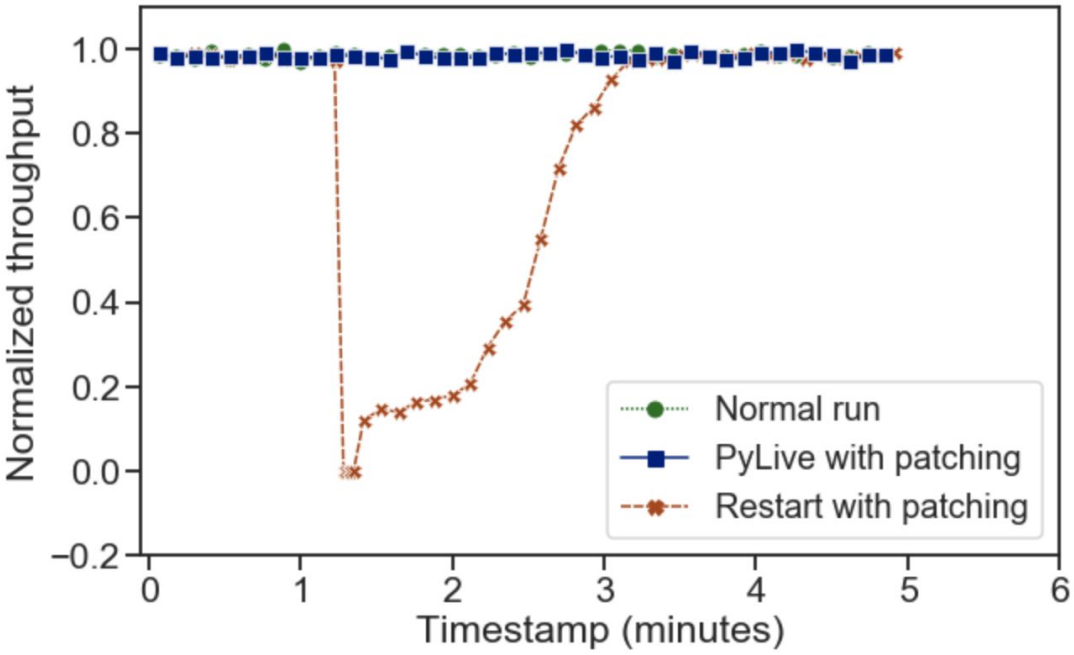
1.4. Saleor_3768

Link: https://github.com/mirumee/saleor/issues/3768

Description:

This is an urgent security issue. Sensitive data such as margin and revenue are exposed to users without permission checking. We use Pylive to patch functions resolve_margin, resolve_price_override, resolve_quantity_ordered, resolve_revenue in module "saleor.graphql.product.types".

Result:



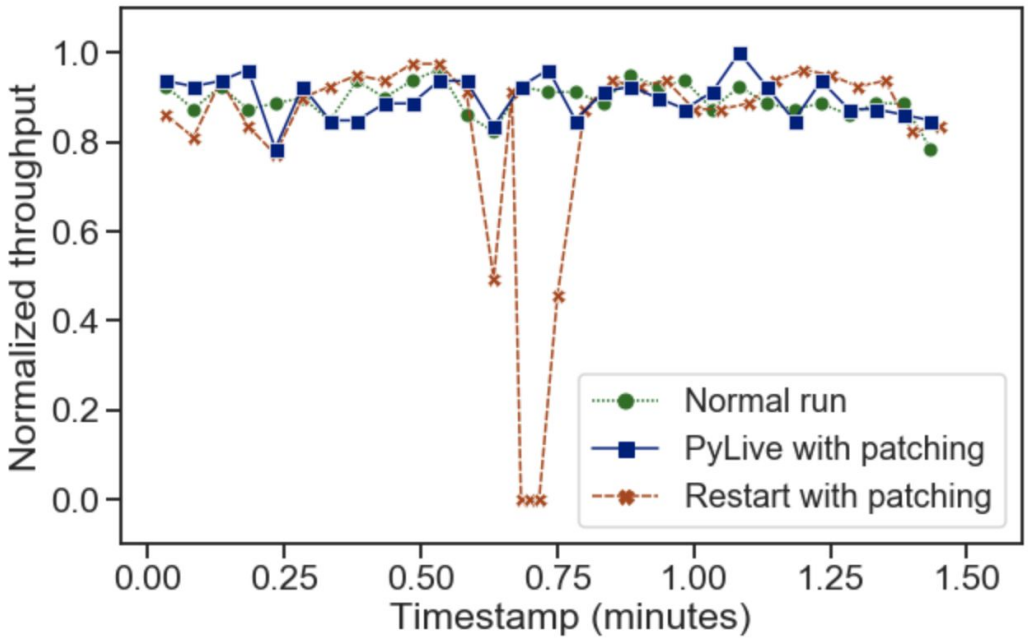
1.5. Odoo_39406

Link: https://github.com/odoo/odoo/issues/39406

Description:

This is an urgent payment-related issue. Users get 500 Error when return to the Odoo shop after making a payment with Paypal and the transaction is not recorded. We use Pylive to patch the function "PaypalController.paypal_validate_data" in module "odoo.addons.payment_paypal.controllers.main".

Result:



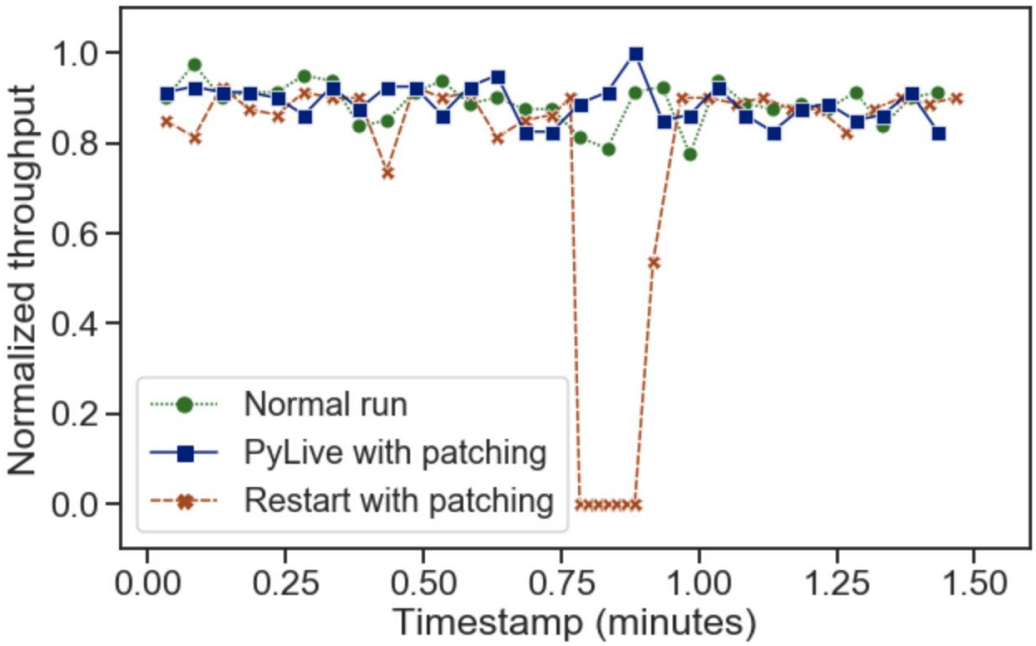
1.6. Odoo_46900

Link: https://github.com/odoo/odoo/issues/46900

Description:

In this issue, validating a large purchase order when there is a large database takes a long time to finish. In order to patch this issue, we use Pylive to patch the function "StockMove._action_done" in module "odoo.addons.stock_account.models.stock_move".

Result:



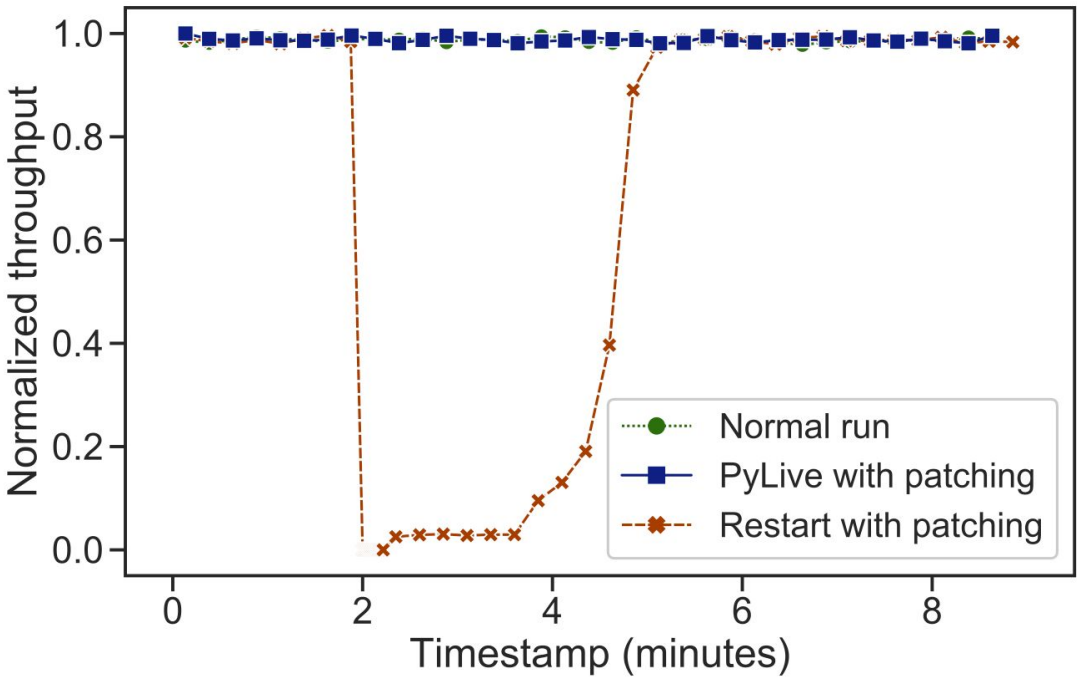
1.7. Pretix_1521

Link: https://github.com/pretix/pretix/issues/1521

Description:

This issue is an authentication bug, which prevents organizers from managing their events and so prevents tickets from being sold. In order to patch this issue, we use Pylive to add and delete attributes in two classes ("ReauthForm.backend.url" in module "src.pretix.base.forms.auth", "EventDeleteForm.user_pw" in module "src/pretix/control/forms/event.py"), add a superclass to multiple classes (add superclass "RecentAuthenticationRequiredMixin" to class "StartShredView", "ShredDownloadView", "ShredExportView", "ShredDoView" in module "src.pretix.control.views.shredder"), and delete one function ("EventDeleteForm.clean_user_pw()" in module "src.pretix.control.forms.event"), making it a non-trivial patch to apply.

Result:



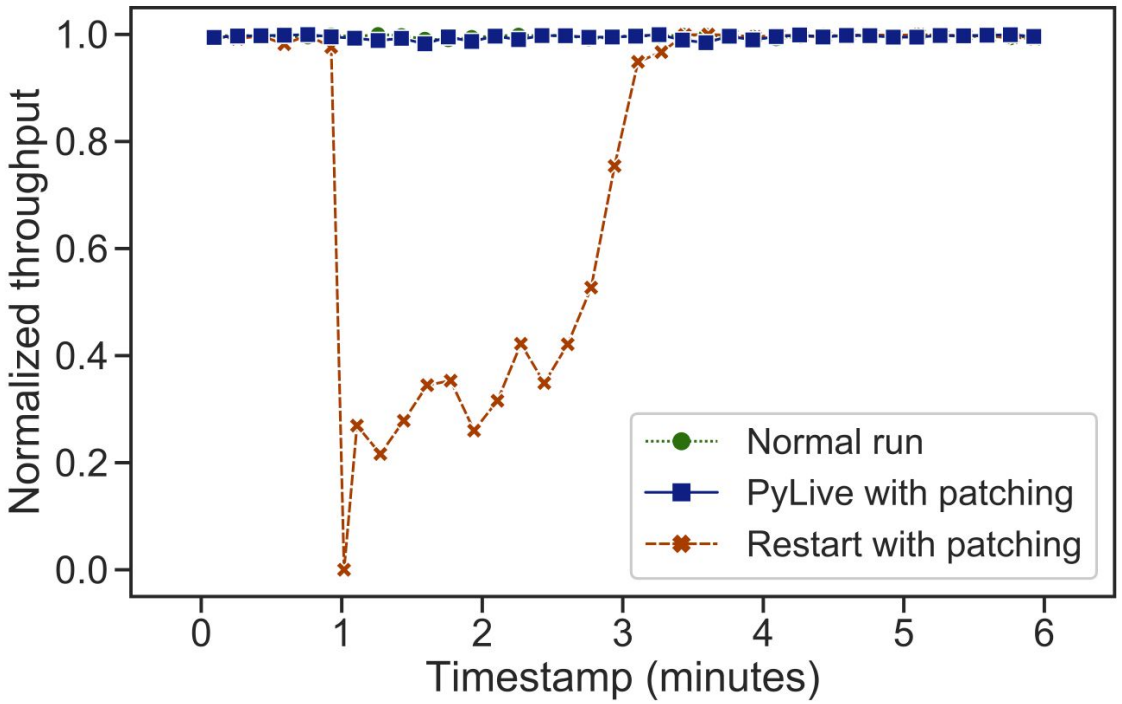
1.8. Shuup_330

Link: https://github.com/shuup/shuup/pull/330

Description:

In this issue, customers cannot add items to their shopping carts. In order to patch this issue, we use Pylive to patch the function "BaseBasket._add_or_replace_line()" in module "shoop.front.basket.objects".

Result:



2. On-the-fly Logging

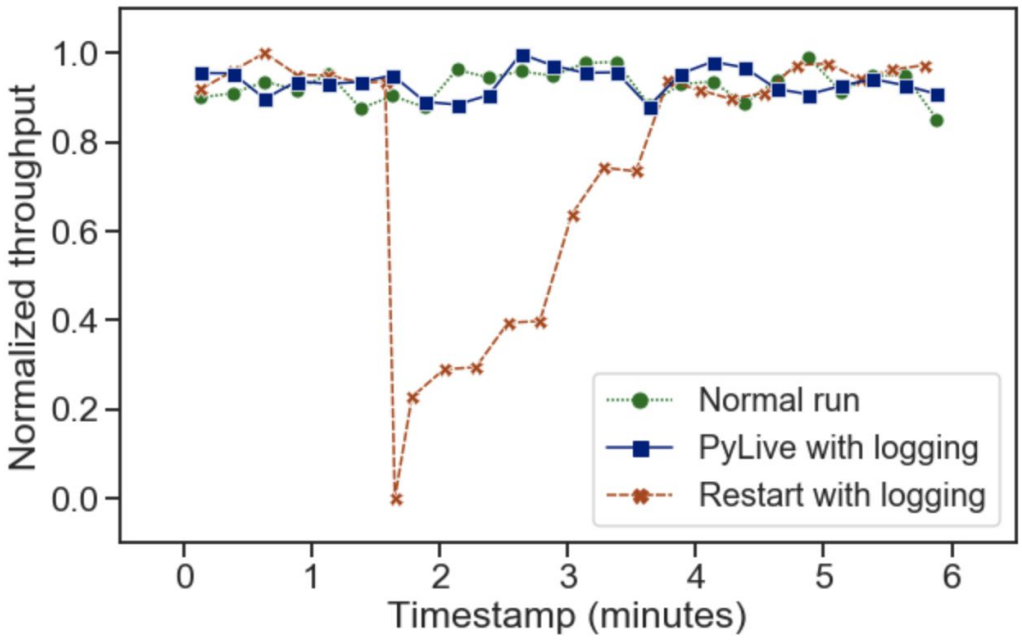
2.1. Django_26504

Link:
https://github.com/django/django/commit/40b69607c751c4afa453edfd41d2ed155e58187e

Description:

In this case, the engineers remove an inappropriate log in a 404 exception handler and add a new log when this exception is not handled by middlewares. To achieve this, we patch the new version to function django.core.handlers.base.BaseHandler.get_response().

Result:



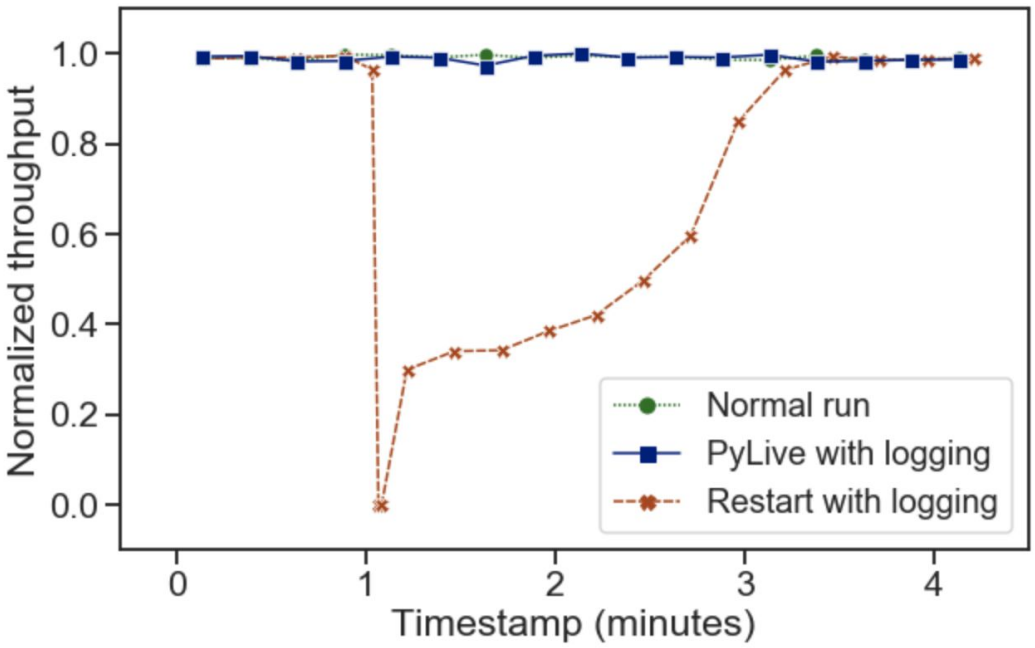
2.2. Oscar_2452

Link:
https://github.com/django-oscar/django-oscar/commit/80d42bb6997fefbf6da04f8233a1c426be422c27

Description:

In this case, a log is added to record the exception that invalid URL names are set to “OSCAR_DASHBOARD_NAVIGATION”. Previously they were silently ignored. The engineers comment that “in Oscar 1.5 this exception was silently ignored which made debugging very difficult.”
To dynamic log, we patch a new version to method oscar.apps.dashboard.nav.Node.default_access_fn().

Result:



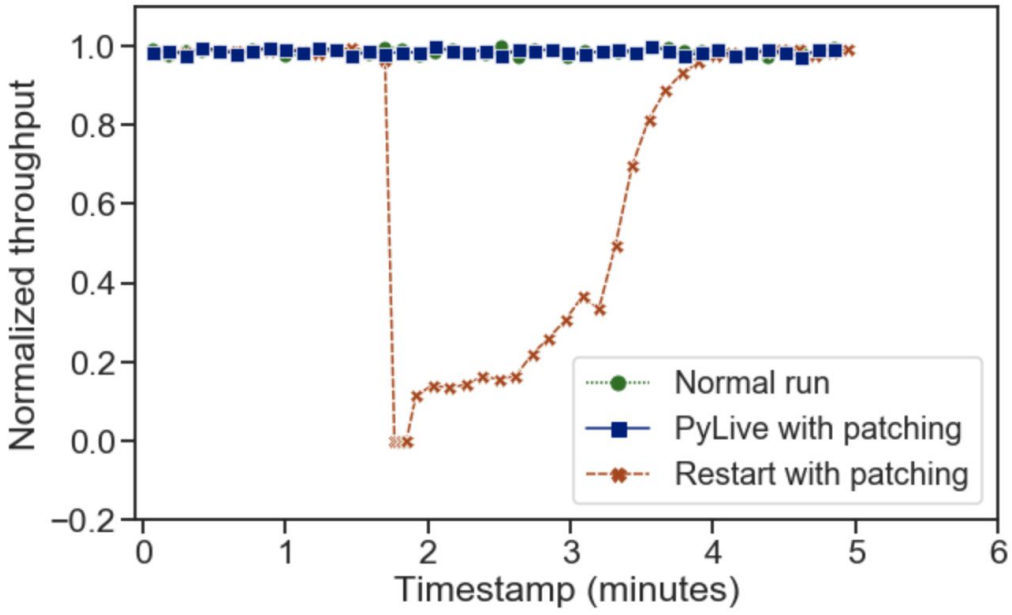
2.3. Saleor_4879_debug

Link: https://github.com/mirumee/saleor/issues/4879

Description:

This is an urgent payment-related issue since users are unable to pay in production. This is because changing the payment gateway from sandbox mode to production mode failed.
In order to debug this issue, we use PyLive to add loggings in functions "ExtensionsManager.list_payment_gateways", "ExtensionsManager.__get_payment_config", "ExtensionsManager.__run_method_on_single_plugin" in module "saleor.extensions.manager"
and function "get_client_token", "get_braintree_gateway" in module "saleor.payment.gateways.braintree".

Result:



3. On-the-fly Profiling

3.1. Oscar_2019

Link: https://github.com/django-oscar/django-oscar/pull/2019

Description:

The dashboard page becomes slow when there are too many customer orders. In order to profile this issue, we use PyLive to profile the module “django.db.models.base.Model”, “oscar” and function “django.db.models.Model.from_db”.

Result: