django-admin-honeypot Documentation

Release 1.1.0

Derek Payton

May 14, 2018

Contents

1 Installation **3**
- 1.1 Requirements . . . 3
- 1.2 Install . . . 3

2 Usage **5**
- 2.1 Basic setup . . . 5
- 2.2 The `honeypot` signal . . . 5
- 2.3 Customizing the login template . . . 6

3 Reference **7**
- 3.1 Settings . . . 7
- 3.2 Signals . . . 7

4 Frequently Asked Questions **9**
- 4.1 Why can't I delete login attempts from the Django admin? . . . 9
- 4.2 Why is the IP address logged as 127.0.0.1? . . . 9

5 Testing **11**
- 5.1 Test requirements . . . 11
- 5.2 Running the tests . . . 11

6 Indices and tables **13**

django-admin-honeypot is a fake Django admin login screen to log and notify admins of attempted unauthorized access. This app was inspired by discussion in and around Paul McMillan's security talk at DjangoCon 2011.

Author Derek Payton

Version 1.1.0

License MIT

Source github.com/dmpayton/django-admin-honeypot

Documentation django-admin-honeypot.readthedocs.org

Contents

CHAPTER 1

Installation

1.1 Requirements

- Python 2.6+ or Python 3.3+
- Django 1.3+

1.2 Install

django-admin-honeypot is on PyPI and can be installed with pip:

```
pip install django-admin-honeypot
```

CHAPTER 2

Usage

2.1 Basic setup

1. Add `admin_honeypot` to `INSTALLED_APPS` in settings.py:

```
INSTALLED_APPS = (
    ...
    'admin_honeypot',
    ...
)
```

2. Update urls.py:

```
urlpatterns = patterns('')
    ...
    url(r'^admin/', include('admin_honeypot.urls', namespace='admin_honeypot')),
    url(r'^secret/', include(admin.site.urls)),
    ...
)
```

2.2 The `honeypot` signal

Every time a login attempt occurs, the `admin_honeypot.signals.honeypot()` signal is fired off. You can setup listeners to this in order to send out any custom notifications or logging.

A default listener, `admin_honeypot.listeners.notify_admins()`, will send an email to all site administrators (`ADMINS` in your site settings) with the details. This can be disabled by setting `ADMIN_HONEYPOT_EMAIL_ADMINS` to false in your site settings.

2.3 Customizing the login template

The template rendered on the honeypot is `admin_honeypot/login.html`. By default this template simply extends `admin/login.html`, but you may want to change it if, e.g., you've customized the Django admin and want to display the stock admin login form.

CHAPTER 3

Reference

3.1 Settings

ADMIN_HONEYPOT_EMAIL_ADMINS

Default: `True`

Used to determine whether or not to email site admins on login attempts. Set to `False` to disable admin emails.

3.2 Signals

`admin_honeypot.signals.honeypot()`

Sent on every login attempt with the following arguments:

instance The LoginAttempt object created

request The current request object

CHAPTER 4

Frequently Asked Questions

4.1 Why can't I delete login attempts from the Django admin?

The delete permsission has been set to false for all users – including superusers – as an added security precaution. This is done so that, in the event that an attacker does make it into your admin, it will be harder to cover up their tracks if they had previously tried to break in through the honeypot.

4.2 Why is the IP address logged as 127.0.0.1?

Django-admin-honeypot pulls the users IP address from the `REMOTE_ADDR` request header. If your Django app is behind a load balancer or proxy web server, this may not be set and instead you will have an `HTTP_X_FORWARDED_FOR` header which contains the IP address in a comma-separated string.

The simple solution is to use a middleware to automatically set `REMOTE_ADDR` to the value of `HTTP_X_FORWARDED_FOR`, like so:

```
class RemoteAddrMiddleware(object):
    def process_request(self, request):
        if 'HTTP_X_FORWARDED_FOR' in request.META:
            ip = request.META['HTTP_X_FORWARDED_FOR'].split(',')[0].strip()
            request.META['REMOTE_ADDR'] = ip
```

See also: http://docs.webfaction.com/software/django/troubleshooting.html#accessing-remote-addr

CHAPTER 5

Testing

Continuous integration provided by Travis CI.

master (latest stable): build passing

develop (bleeding edge): build passing

5.1 Test requirements

See *requirements.txt*:

```
pytest
pytest-cov
pytest-django
pytest-pep8
pytest-pythonpath
```

5.2 Running the tests

Once your requirements are installed, the unit tests can be run with:

```
$ py.test tests/ --cov admin_honeypot --cov-report term-missing --pep8 admin_honeypot

...

=============== 5 passed, 12 skipped in 0.46 seconds ===============
```

For testing against different Python versions, we use Tox. Please be aware that this only tests against the latest Django release.

```
$ tox

...

_________________ summary _________________
py27: commands succeeded
py33: commands succeeded
congratulations :)
```

CHAPTER 6

Indices and tables

- genindex
- modindex
- search