

FORMLOCK MANUAL-DEMO

INSTALLATION

FormLock is a Chrome browser extension. It can be installed by going to “chrome://extensions”, enabling “Developer mode”, and selecting the FormLock folder in “Load unpacked extension”.

ACCESSING DEMO

The demo website is located at “<http://anonymous-demo.github.io>”. It provides a simple contact form and demonstrates several ways how entered PII may leak to “<http://attacking-party.appspot.com>”.

VULNARABLE SUBMISSION

Leaking script may try to save PII to send later.
In case of using FormLock, new browsing data is being removed upon releasing the lock.

Cookies:

Storage:

[Clear all](#)

Try to contact us:

Name:

E-mail:

Message:

[Send your message](#)

Entered PII may leak via capturing script on form submission, e.g. by loading an iFrame:

In reality, the iFrame will be invisible, or a "pixel tag" will be used.
In case of using FormLock, the request will be blocked.

Leaking script may try to save PII to send later.
In case of using FormLock, new browsing data is being removed upon releasing the lock.

Cookies:


weblead_email=test@test.com

Storage:

weblead_email, test@test.com

[Clear all](#)

Try to contact us:



The page at anonymous-demo.github.io says:
Transition to the landing page

[OK](#)

[Send your message](#)

Entered PII may leak via capturing script on form submission, e.g. by loading an iFrame:

Third-party received the E-mail:
test@test.com

In reality, the iFrame will be invisible, or a "pixel tag" will be used.
In case of using FormLock, the request will be blocked.

FORMLOCK MANUAL-DEMO

Leaking script may try to save PII to send later.
In case of using FormLock, new browsing data is being removed upon releasing the lock.

Cookies:

```
weblead_email=test@test.com
```

Storage:

```
weblead_email, test@test.com
```

```
weblead_url, http%3A//anonymous-  
demo.github.io/landing.html%3Fuser_name%3D  
Test%26user_email%3Dtest%2540test.com%26u  
ser_message%3DH%2521
```

[Clear all](#)

Thank you!

[Back to Main](#)

The PII may leak via URL that is captured and sent by a script:

```
Third-party received the URL:  
http://anonymous-  
demo.github.io/landing.html?  
user_name=Test&user_email=test@test.  
com&user_message=Hi!
```

The PII may leak via Referer header on loading third-party content:

```
Third-party received the Referer:  
http://anonymous-  
demo.github.io/landing.html?  
user_name=Test&user_email=test%40tes  
t.com&user_message=Hi%21
```

In case of using FormLock, the requests will be blocked.

The leakage resumes because scripts saved the provided email and the captured URL in cookies and in storage. You may click “*Clear all*” to start from scratch.

PROTECTED SUBMISSION

In case of enabled FormLock extension the contact form becomes highlighted with red borders, because it is submitted with GET. To start the “safe” filling and submission, just right-click to call a context menu somewhere inside the form and choose to “Set LOCK”.

Leaking script may try to save PII to send later.
In case of using FormLock, new browsing data is being removed upon releasing the lock.

Cookies:

Storage:

[Clear all](#)

Try to contact us:

Name:

E-mail:

Message:

Entered PII may be leaked via script on form submission, if iFrame:

Back

Forward

Reload

Save As...

Print...

Translate to English

View Page Source

View Page Info

FormLock

Inspect Element

Submit method: get

Submits to: github.io

Explain sharing risks.

Set LOCK

*In reality, the iFrame will be invisible, or a "pixel tag" will be used.
In case of using FormLock, the request will be blocked.*

FORMLOCK MANUAL-DEMO

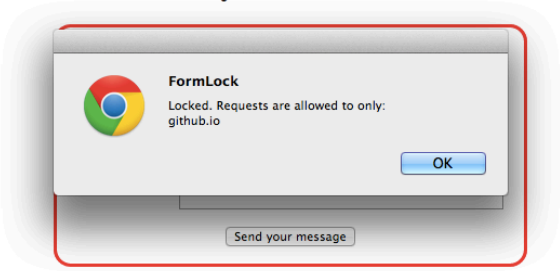
Leaking script may try to save PII to send later.
In case of using FormLock, new browsing data is being removed upon releasing the lock.

Cookies:


Storage:

Clear all

Try to contact us:

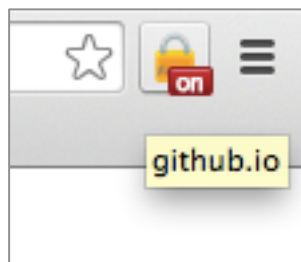


Entered PII may leak via capturing script on form submission, e.g. by loading an iFrame:



*In reality, the iFrame will be invisible, or a "pixel tag" will be used.
In case of using FormLock, the request will be blocked.*

Additionally, the active lock is shown on the FormLock's icon:



Now, leaking requests to the “attacking-party” are blocked:

Leaking script may try to save PII to send later.
In case of using FormLock, new browsing data is being removed upon releasing the lock.

Cookies:

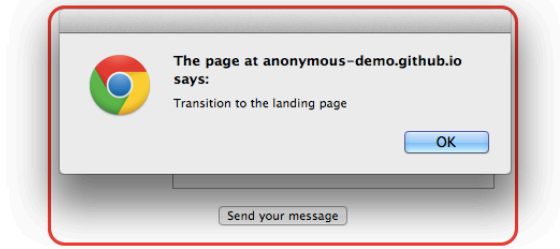
weblead_email=test@test.com

Storage:

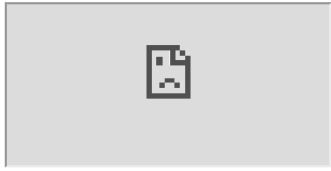
weblead_email, test@test.com

Clear all

Try to contact us:



Entered PII may leak via capturing script on form submission, e.g. by loading an iFrame:



*In reality, the iFrame will be invisible, or a "pixel tag" will be used.
In case of using FormLock, the request will be blocked.*

FORMLOCK MANUAL-DEMO

Leaking script may try to save PII to send later.
In case of using FormLock, new browsing data is being removed upon releasing the lock.

Cookies:

weblead_email=test@test.com

weblead_url=http%3A//anonymous-demo.github.io/landing.html%3Fuser_name%3DTest%26user_email%3Dtest%2540test.com%26user_message%3DHi%2521

Storage:

weblead_email, test@test.com


weblead_url, http%3A//anonymous-demo.github.io/landing.html%3Fuser_name%3DTest%26user_email%3Dtest%2540test.com%26user_message%3DHi%2521

[Clear all](#)

Thank you!


[Back to Main](#)

The PII may leak via URL that is captured and sent by a script:



Requests to the server have been blocked by an extension.

The PII may leak via Referer header on loading third-party content:



In case of using FormLock, the requests will be blocked.

Removing the lock is also done via the context menu (clicking anywhere):

Leaking script may try to save PII to send later.
In case of using FormLock, new browsing data is being removed upon releasing the lock.

Cookies:

weblead_email=test@test.com

weblead_url=http%3A//anonymous-demo.github.io/landing.html%3Fuser_name%3DTest%26user_email%3Dtest%2540test.com%26user_message%3DHi%2521

Storage:

weblead_email, test@test.com


weblead_url, http%3A//anonymous-demo.github.io/landing.html%3Fuser_name%3DTest%26user_email%3Dtest%2540test.com%26user_message%3DHi%2521

[Clear all](#)


Thank you!

[Back to Main](#)

The PII may leak via URL that is captured and sent by a script:



The PII may leak via Referer header on loading third-party content:



In case of using FormLock, the requests will be blocked.

- Back
- Forward
- Reload
- Save As...
- Print...
- Translate to English
- View Page Source
- View Page Info
- FormLock**
- Inspect Element

Submit method: undefined

Submits to: (empty)

Explain sharing risks.

Remove LOCK

The alert provides statistics on third-party requests blocked:

Leaking script may try to save PII to send later.
In case of using FormLock, new browsing data is being removed upon releasing the lock.

Cookies:

weblead_url=http%3A//anonymous-demo.github.io/landing.html

Storage:

weblead_url, http%3A//anonymous-demo.github.io/landing.html

[Clear all](#)

Thank you!

The PII may leak via URL that is captured and sent by a script:

The PII may leak via Referer header on loading third-party content:

Third-party received the Referer:
http://anonymous-demo.github.io/landing.html

In case of using FormLock, the requests will be blocked.

FormLock

Unlocked. Third-party requests blocked 4:

- attacking-party.appspot.com
- attacking-party.appspot.com
- attacking-party.appspot.com
- attacking-party.appspot.com

[OK](#)

FORMLOCK MANUAL-DEMO

The browsing data and the landing URL are safely cleared, so no further leakage is possible:

Leaking script may try to save PII to send later.
In case of using FormLock, new browsing data is being removed upon releasing the lock.

Cookies:

weblead_url=http%3A//anonymous-demo.github.io/landing.html

Storage:

weblead_url, http%3A//anonymous-demo.github.io/landing.html

[Clear all](#)

Try to contact us:

Name:

E-mail:

Message:

[Send your message](#)

Entered PII may leak via capturing script on form submission, e.g. by loading an iFrame:

In reality, the iFrame will be invisible, or a "pixel tag" will be used.
In case of using FormLock, the request will be blocked.

Thank you for trying the FormLock!