Less is More

Quantifying the Security Benefits of Debloating Web Applications

Babak Amin Azad Pierre Laperdrix Nick Nikiforakis

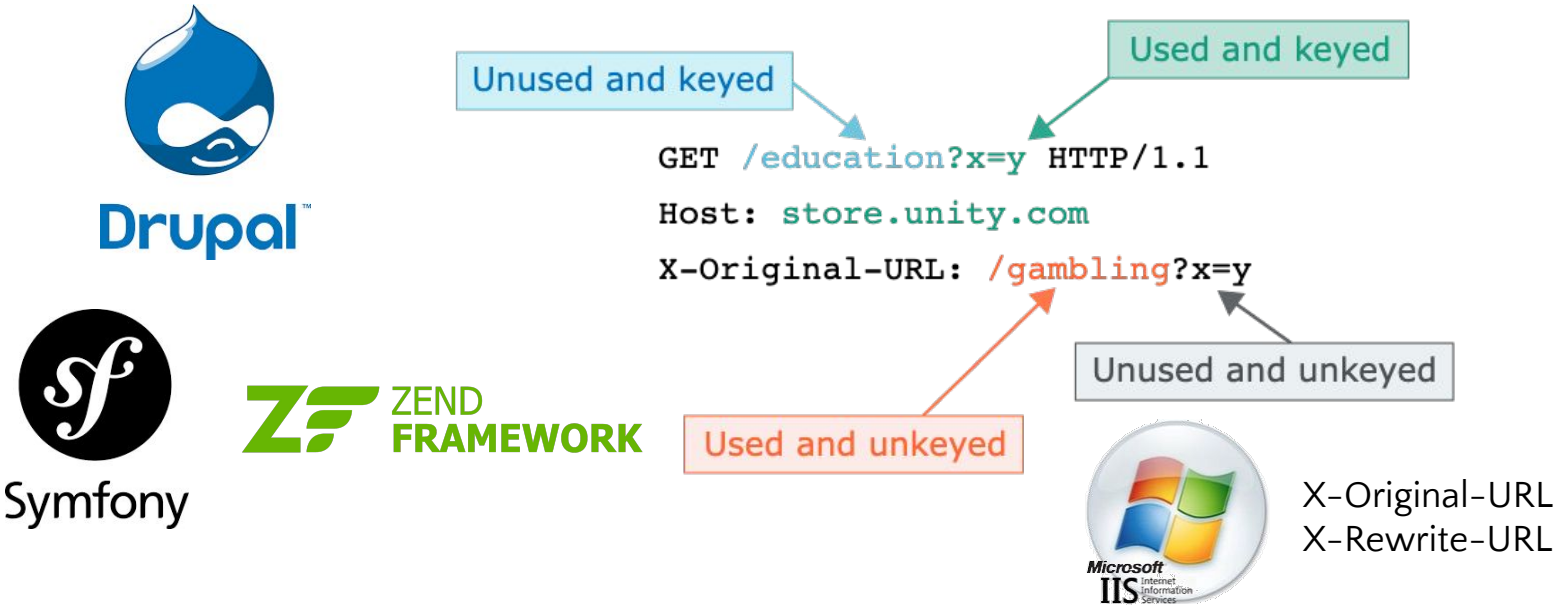
Stony Brook University

What is software debloating?

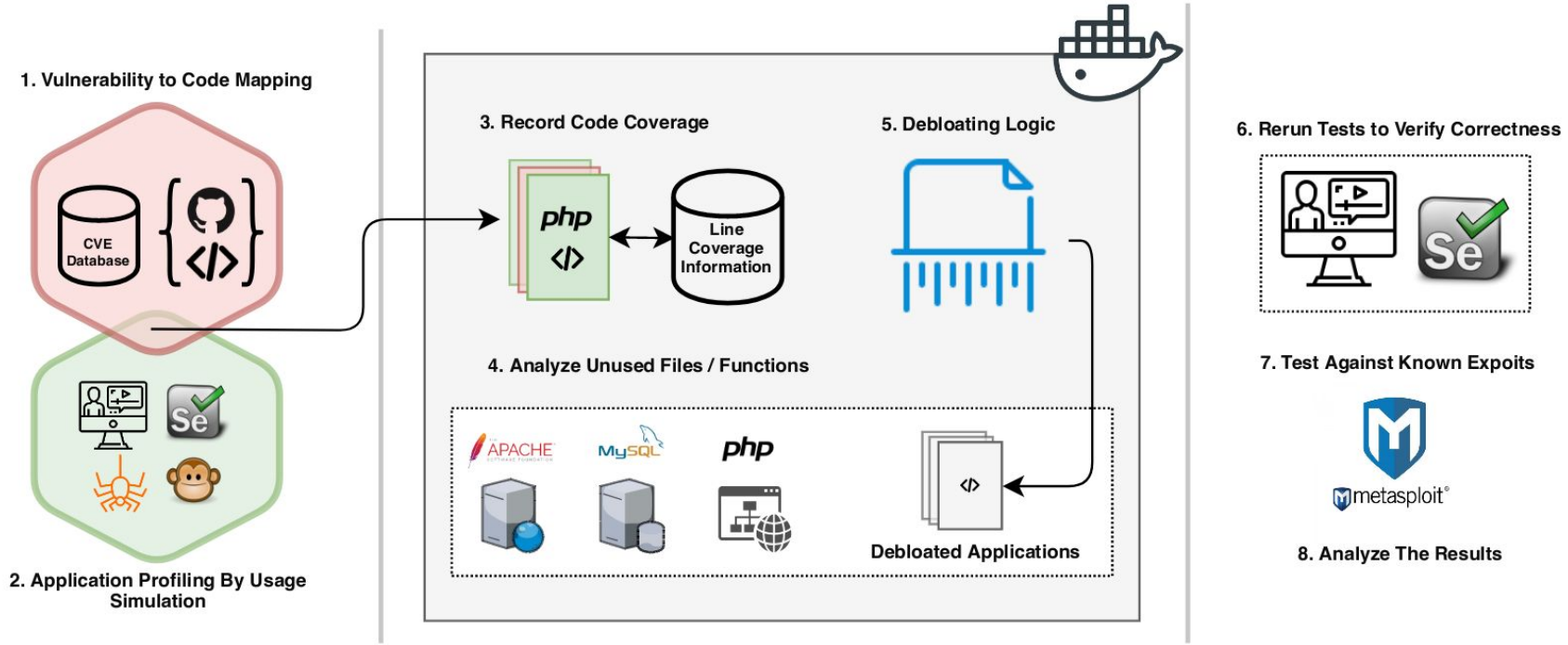
*"Reducing the **attack surface** by removing pieces of code that **are not required** by users."*

You're vulnerable, but do you have to be?

Web Cache Poisoning vulnerability on Drupal https://portswigger.net/blog/practical-web-cache-poisoning



Debloating Pipeline



Identifying important functionalities of an application

- Find tutorials for these applications
- Automate them using Selenium

Tutorials

Example of tasks covered by tutorials

1. Login
2. Create a database
3. Create tables
4. Run queries
5. Drop database
6. …

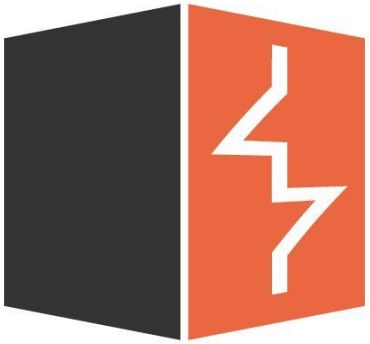
What's not covered by tutorials

1. Some pages on the front of the application
2. Error handlers

Expanding the breadth of coverage

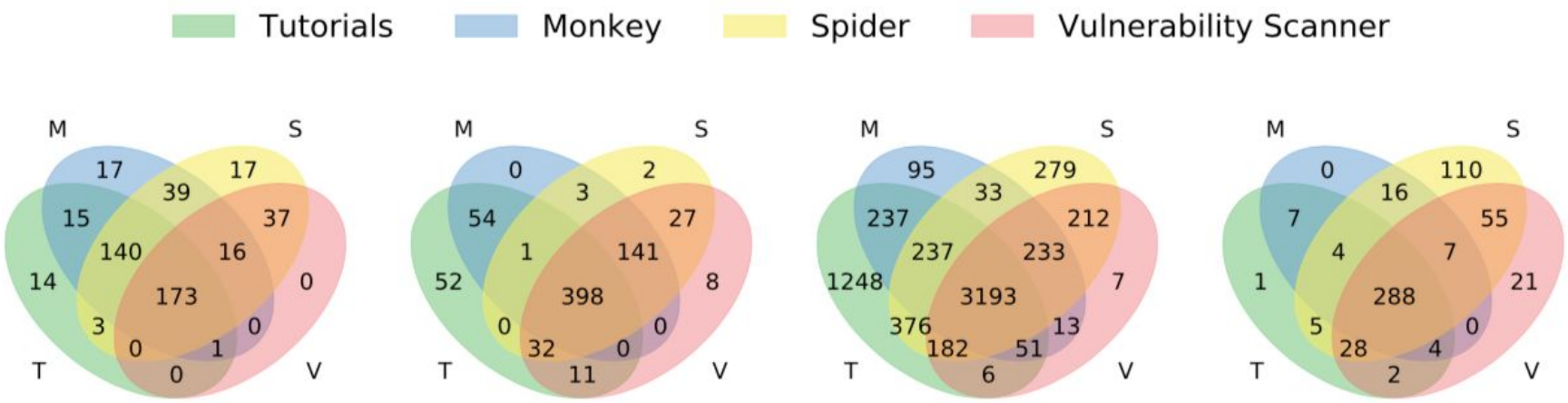
Monkey Testing

Spider



Vulnerability Scanner

Files covered by each testing tool



(a) phpMyAdmin 4.7.0 *(b) MediaWiki 1.28.0* *(c) Magento 2.0.5* *(d) WordPress 4.7.1*

File & Function level debloating

- Remove the contents of unused files/functions
- Use place holders
  - Log information about execution of removed code
  - Stop the execution flow to prevent entering an unknown state

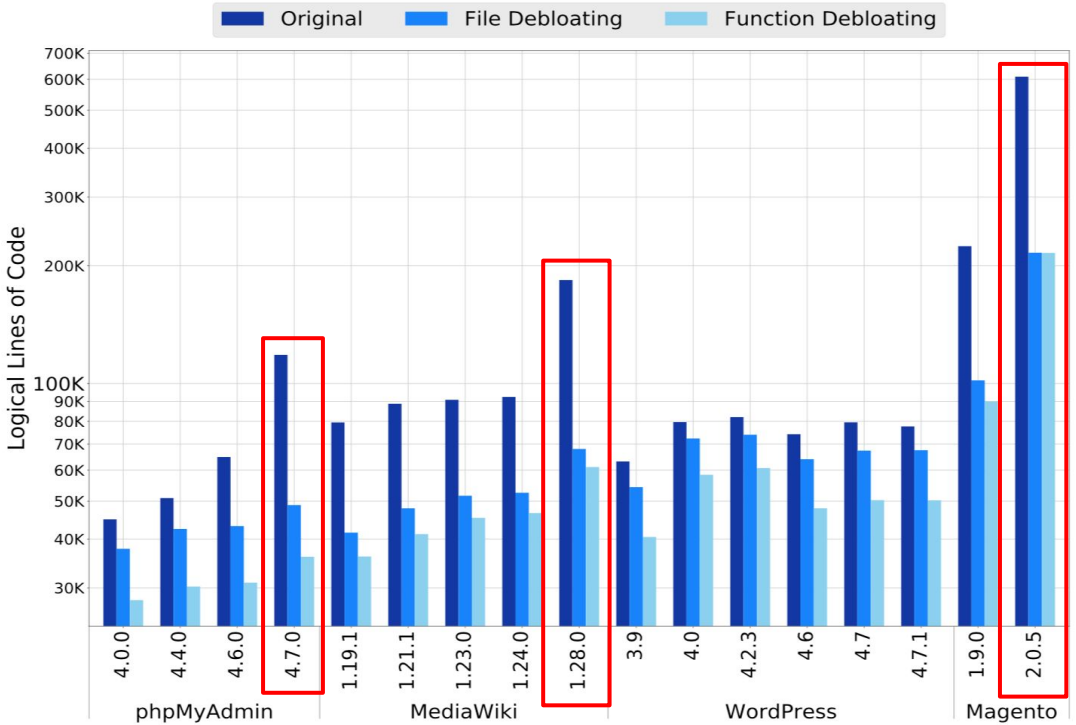
Results #1: Reduction of LLOC after debloating

File Debloating

- Average **33%** reduction
- WordPress: **9%**
- Magento: **65%**
  (400 KLLOC)

Function Debloating

- Average **47%** reduction (**+14%**)
- WordPress: **31%** (**+22%**)
- Magento **71%** (**+6%**)



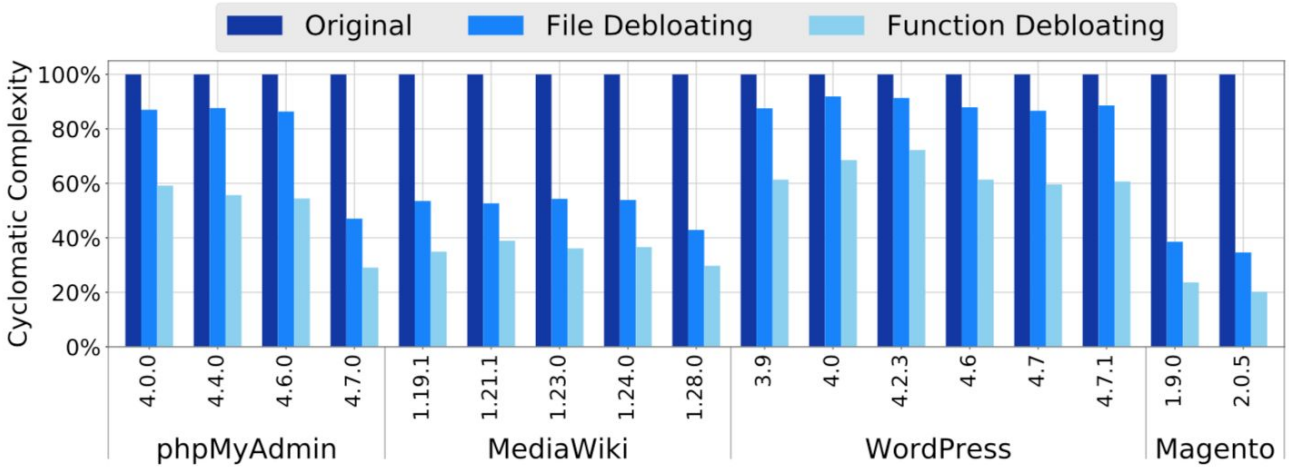
Results #2: Reduction of Cyclomatic Complexity

File Debloating

- Average of **32.5%** reduction
- WordPress: **6%**
- Magento: **74.3%**

Function Debloating

- Average **50.3%** reduction (**+18%**)
- WordPress: **24%** (**+18%**)
- Magento **80.2%** (**+6%**)



Coverage of CVEs based on Usage Profiles

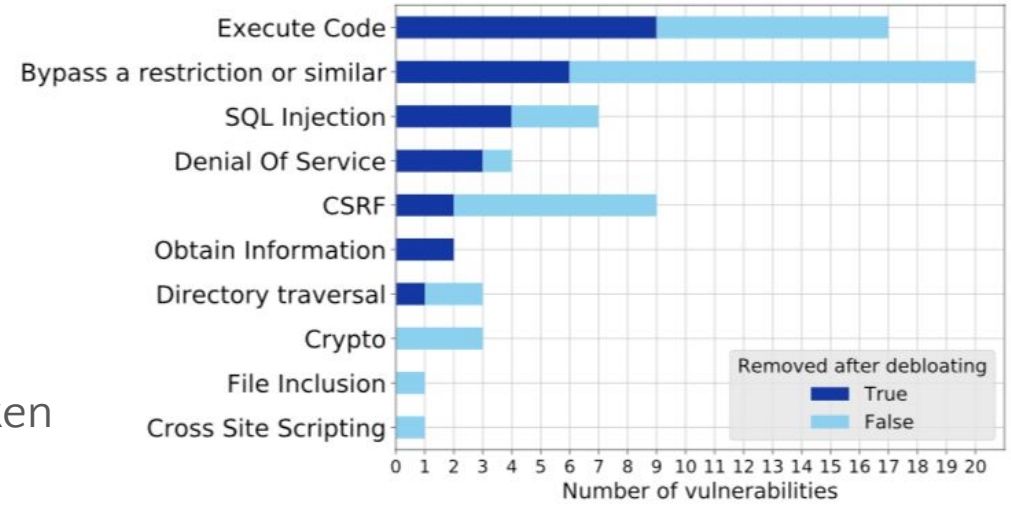
ID	CVE	Software	Version	File Name	Triggered
64	CVE-2014-8959	phpMyAdmin	4.0.0	libraries/gis/pma_gis_factory.php	✖
63	CVE-2013-3240	phpMyAdmin	4.0.0	libraries/plugin_interface.lib.php	✔
24	CVE-2016-6619	phpMyAdmin	4.0.0	libraries/Table.class.php	✔
22	CVE-2016-6609	phpMyAdmin	4.0.0	libraries/plugins/export/ExportPhparray.class.php	✔
21	CVE-2016-9866	phpMyAdmin	4.0.0	prefs_manage.php	✖

Results #3: Reduction of CVEs

Application	Strategy	Total Removed CVEs	
phpMyAdmin	File Debloating	4/20	**20 %**
	Function Debloating	12/20	**60 %**
MediaWiki	File Debloating	8/21	**38 %**
	Function Debloating	10/21	**47.6 %**
WordPress	File Debloating	0/20	**0 %**
	Function Debloating	2/20	**10 %**
Magento	File Debloating	1/8	**12.5 %**
	Function Debloating	3/8	**37.5 %**

Types of vulnerabilities removed by debloating

- Crypto and cookie related vulnerabilities usually can't be removed by debloating.
- CSRF vulnerabilities are only removed when the underlying feature is removed.
- Code execution vulnerabilities can either be removed or broken by removing the POI gadgets.



Effect of external dependencies on software bloat

	Before debloating		After function-level debloating	
Application	*LLOC in main App*	*LLOC in packages*	*LLOC in main App*	*LLOC in packages*
phpMyAdmin 4.7.0	36k	82k	26k (**-26.2 %**)	10k (**-88.3 %**)
MediaWiki 1.28.0	133k	51k	54k (**-58.8%**)	6k (**-87.7 %**)
Magento 2.0.5	396k	213k	182k (**-54.2 %**)	34k (**-84.0 %**)

Statistics about removed external packages

	Before debloating	After function-level debloating	
Application	# Packages	# packages completely removed	# packages with < 30 % of lines removed
phpMyAdmin 4.7.0	45	38 (**84 %**)	4
MediaWiki 1.28.0	40	24 (**60 %**)	12
Magento 2.0.5	71	58 (**82 %**)	2

But if a package is never used, does it contribute to the attack surface?

PHP Object Injection (POI) attacks

- Unsafe object deserialization vulnerability is the target of this exploit.
- Attacker can control value of properties on injected objects.
  (Also known as Property Oriented Programming, POP)
- But the attacker cannot control execution of functions.
- The chain is made based on magic functions.
- The chain usually ends with a write to file system
  or a database transaction.

Magic functions:
__construct()
__toString()
__destruct()
__wakeup()
...

Results #4: Reduction of object injection gadgets

Application	Package	Removed by Debloating	
		File	Function
phpMyAdmin 4.7.0	Doctrine	✔	✔
	Guzzle	✔	✔
MediaWiki 1.28.0	Monolog	✔	✔
Magento 2.0.5	Doctrine	✔	✔
	Monolog	✘	✔
	Zendframework	✘	✔

Source code and the artifacts are publicly available

- Debloating pipeline to evaluate and debloat custom applications
- Debloated web applications
- Source code coverage information
- CVE to source code mappings & Exploits

https://debloating.com

Conclusion

- Debloating can reduce web applications attack surface significantly
  - Up to **71 %** reduction in **LLOC**
  - Up to **60 %** reduction in **CVEs**
  - Up to **100 %** removal of **POI Gadgets**
- Web vulnerabilities & their exploitation is different, as a result web debloating is different (Targeting actual vulnerabilities rather than dead code)
- We also need to focus on usability and performance of debloating schemes
- Artifacts and debloated applications are available at: https://debloating.com