Toward Black-box Detection of Logic Flaws in Web Applications

Giancarlo Pellegrino
gpellegrino@deeds.informatik.tu-darmstadt.de

Davide Balzarotti
davide.balzarotti@eurecom.fr

San Diego, 25/02/2014

Agenda

- Problem
- Approach
  - Model Inference
  - Behavioral Patterns Extraction
  - Attack Pattern-based Test Case Generation
  - Test Execution and Oracle
- Evaluation
- Conclusion

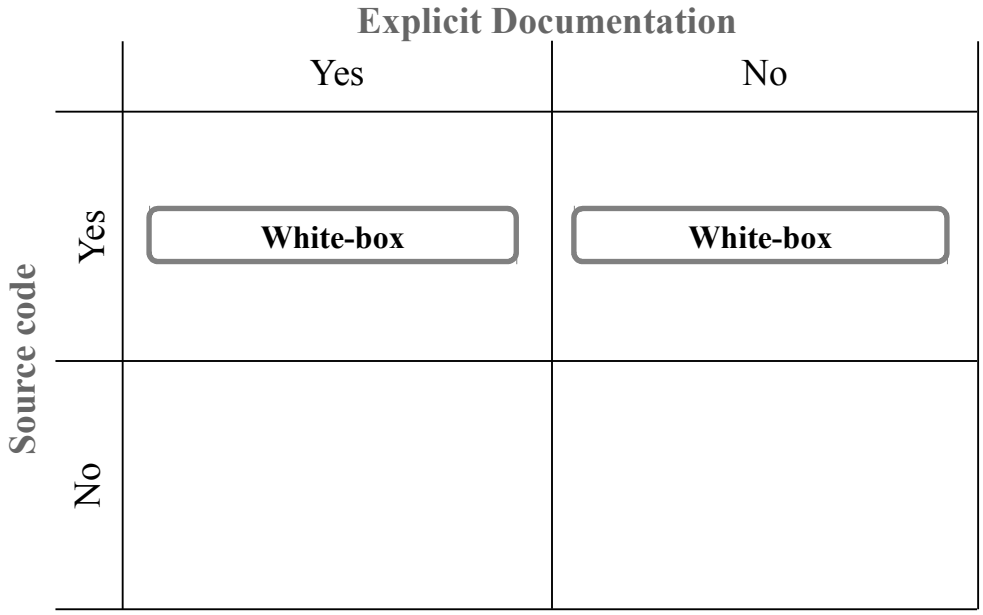
Logic Flaws

- Also known as design flaws/errors, business/application logic errors/flaws
- Lack a formal definition
  - CWE-ID 840: Business logic errors are "w*eaknesses […] that commonly allow attackers to manipulate the business logic of an application*"
- Mainly caused by insufficient validation of the application workflow and data flow
- Can exhibit patterns, e.g.
  - Improper authentication/authorization

Problem

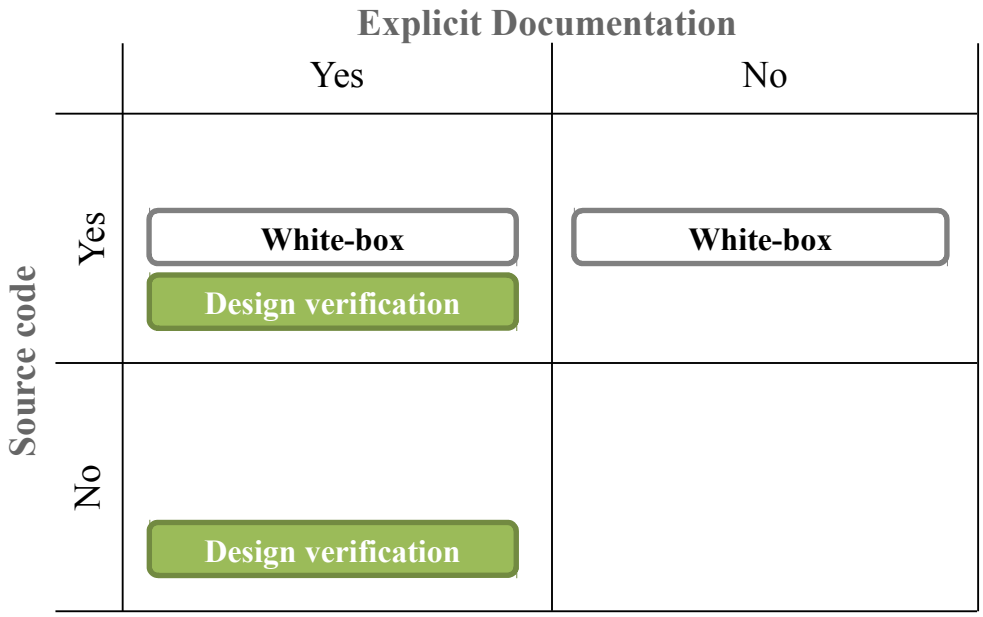
Source code \ Explicit Documentation	Yes	No
Yes		
No		

Problem



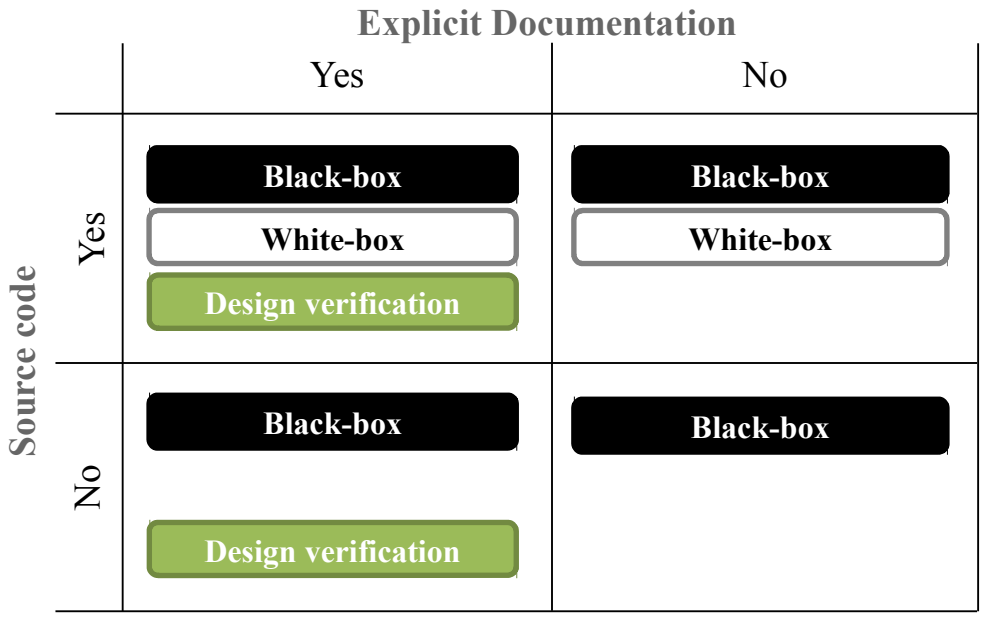
- White-box testing [BalzarottiCCS07, FelmetsgerUSENIX10, ...]
  - Source code of WA may not be available → White-box not applicable!

Problem



- White-box testing [BalzarottiCCS07, FelmetsgerUSENIX10, ...]
  - Source code of WA may not be available → White-box not applicable!
- Design verification [LoweCSF97, ArmandoCSF07, ...]
  - Specification of WA may not be available → DV not applicable!

Problem



- White-box testing [BalzarottiCCS07, FelmetsgerUSENIX10, ...]
  - Source code of WA may not be available → White-box not applicable!
- Design verification [LoweCSF97, ArmandoCSF07, ...]
  - Specification of WA may not be available → DV not applicable!
- Black-box testing, e.g., web scanners [DoupèDIMVA10, WangS&P11, WangS&P12]
  - Cannot automatically detect logic flaws

➔ ***Testing for logic flaws is done manually***

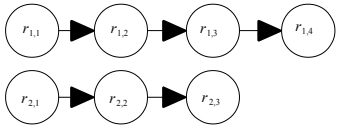
Our Approach

Overview

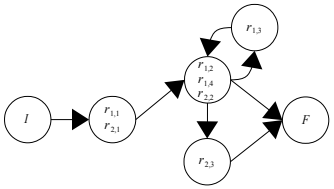
1) Model Inference

74.125.230.240 > 192.168.1.89
192.168.1.89 > 74.125.230.240
74.125.230.240 > 192.168.1.89

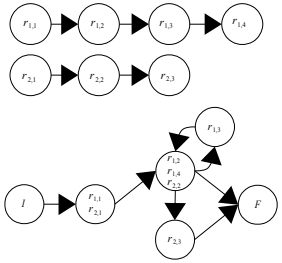
Resource Abstraction



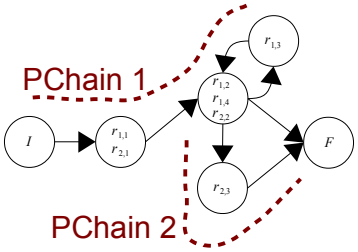
Resource Clustering



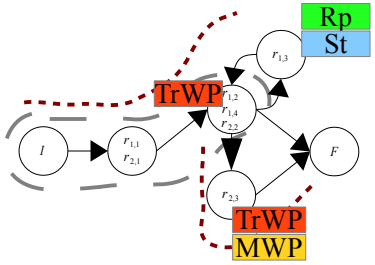
2) Behavioral Patterns



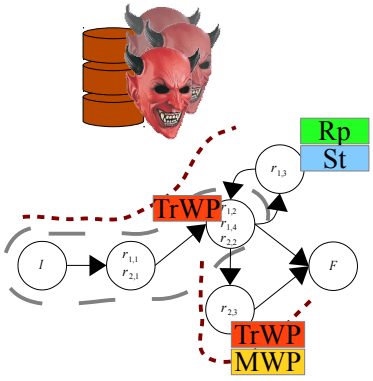
Data flow Patterns



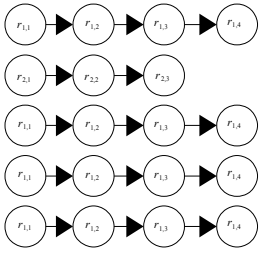
Workflow Patterns



3) Test Cases Generation



Test Cases



4) Test Cases Execution

Execution

74.125.230.240 > 192.168.1.89
192.168.1.89 > 74.125.230.240
74.125.230.24 > 192.168.1.89

Oracle

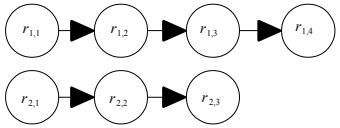
Verdict:
Flaw found in test 1 and 2

Model Inference

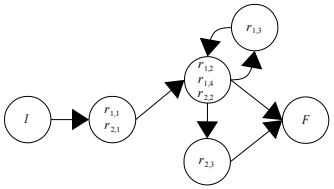
1) Model Inference

74.125.230.240 > 192.168.1.89
192.168.1.89 > 74.125.230.240
74.125.230.240 > 192.168.1.89

Resource Abstraction



Resource Clustering



2) Behavioral Patterns

Data flow Patterns

Workflow Patterns

3) Test Cases Generation

Test Cases

4) Test Cases Execution

Execution

74.125.230.240 > 192.168.1.89
192.168.1.89 > 74.125.230.240
74.125.230.240 > 192.168.1.89

Oracle

Verdict:
Flaw found in test 1 and 2

Behavioral Patterns Extraction

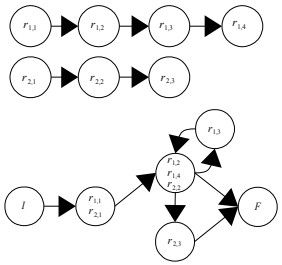
1) Model Inference

74.125.230.240 > 192.168.1.89
192.168.1.89 > 74.125.230.240
74.125.230.240 > 192.168.1.89

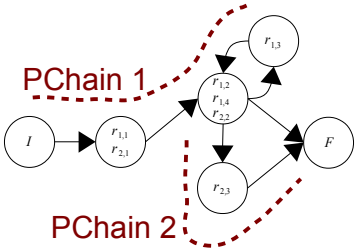
Resource Abstraction

Resource Clustering

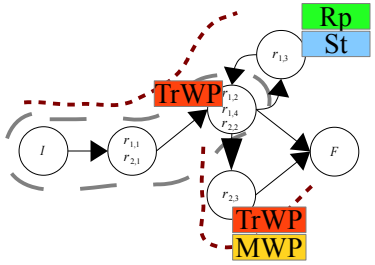
2) Behavioral Patterns



Data flow Patterns



Workflow Patterns



3) Test Cases Generation

Test Cases

4) Test Cases Execution

Execution

74.125.230.240 > 192.168.1.89
192.168.1.89 > 74.125.230.240
74.125.230.24 > 192.168.1.89

Oracle

Verdict:
Flaw found in test 1 and 2

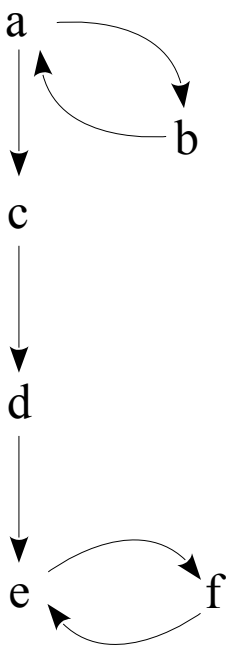
Workflow Patterns

Traces:

$$\pi_1 = \langle a, b, a, c, d, e, f, e \rangle$$

$$\pi_2 = \langle a, c, d, e, f, e \rangle$$

Model:



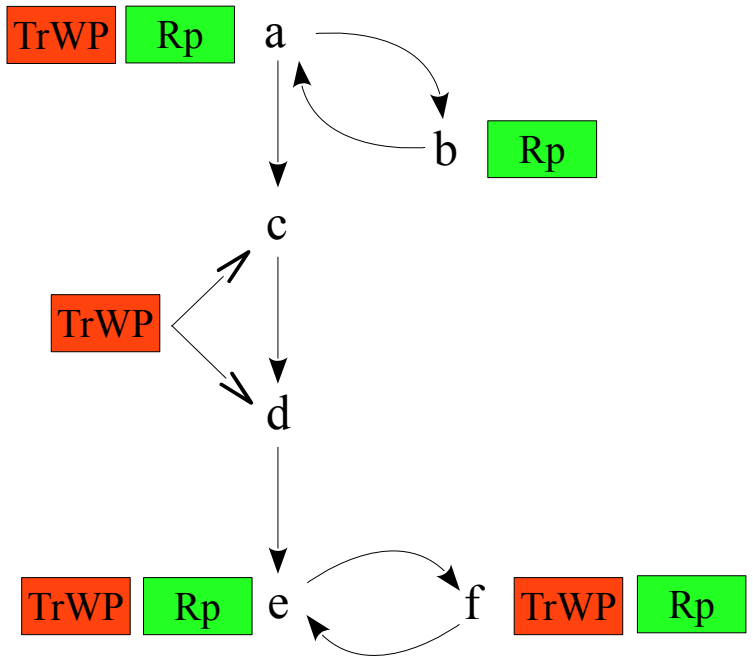
Workflow Patterns

Traces:

$$\pi_1 = \langle a, b, a, c, d, e, f, e \rangle$$

$$\pi_2 = \langle a, c, d, e, f, e \rangle$$

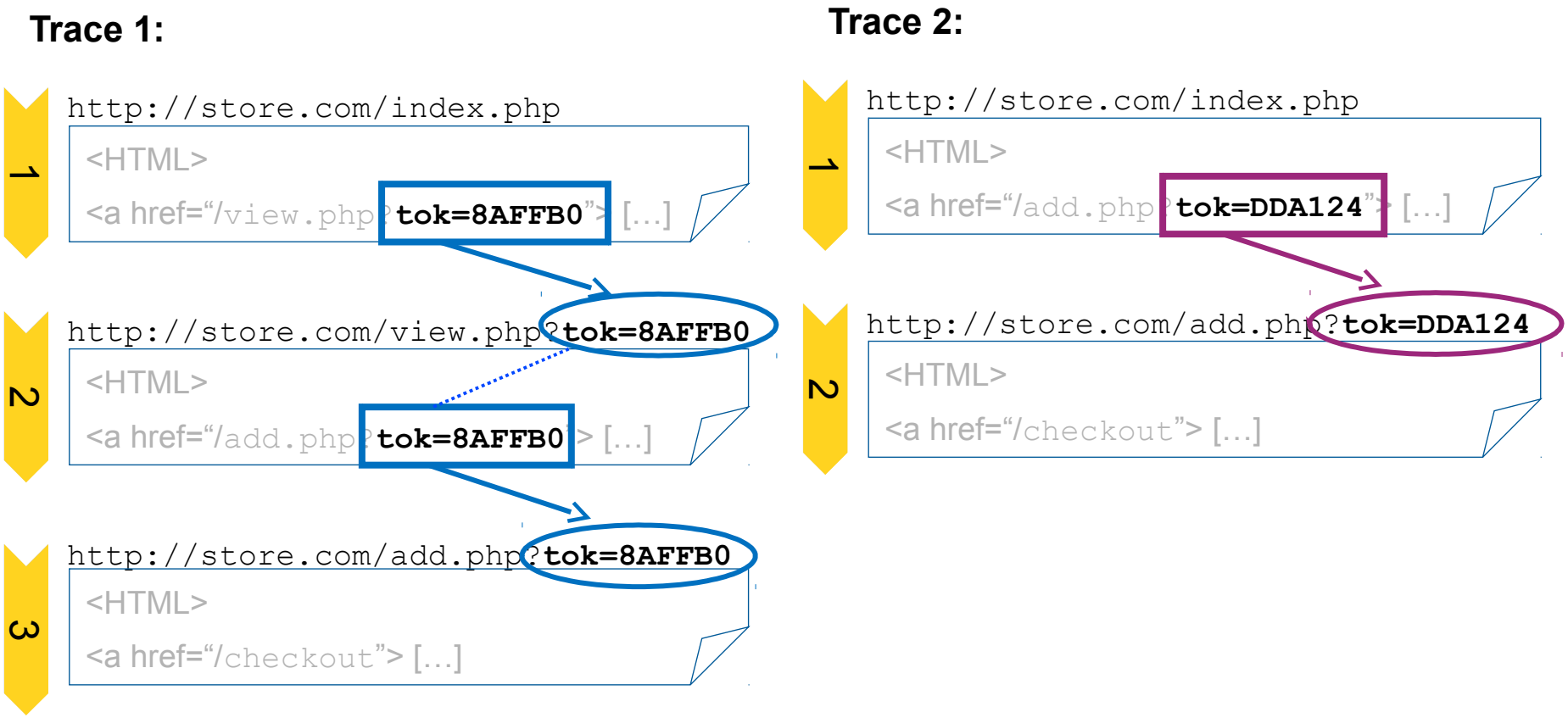
Model:



TrWP : Trace Waypoints

Rp : Repeatable Operations

Data flow Patterns



Test Case Generation

1) Model Inference

74.125.230.240 > 192.168.1.89
192.168.1.89 > 74.125.230.240
74.125.230.240 > 192.168.1.89

Resource Abstraction

Resource Clustering

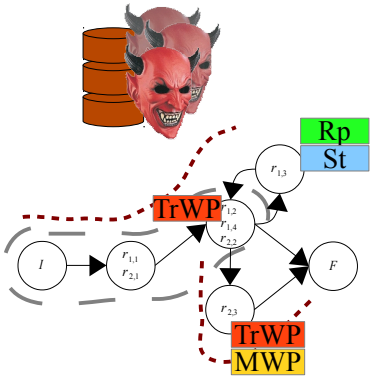
2) Behavioral Patterns

Extraction

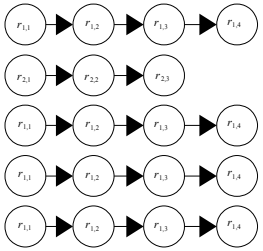
Data flow Patterns

Workflow Patterns

3) Test Cases Generation



Test Cases



4) Test Cases Execution

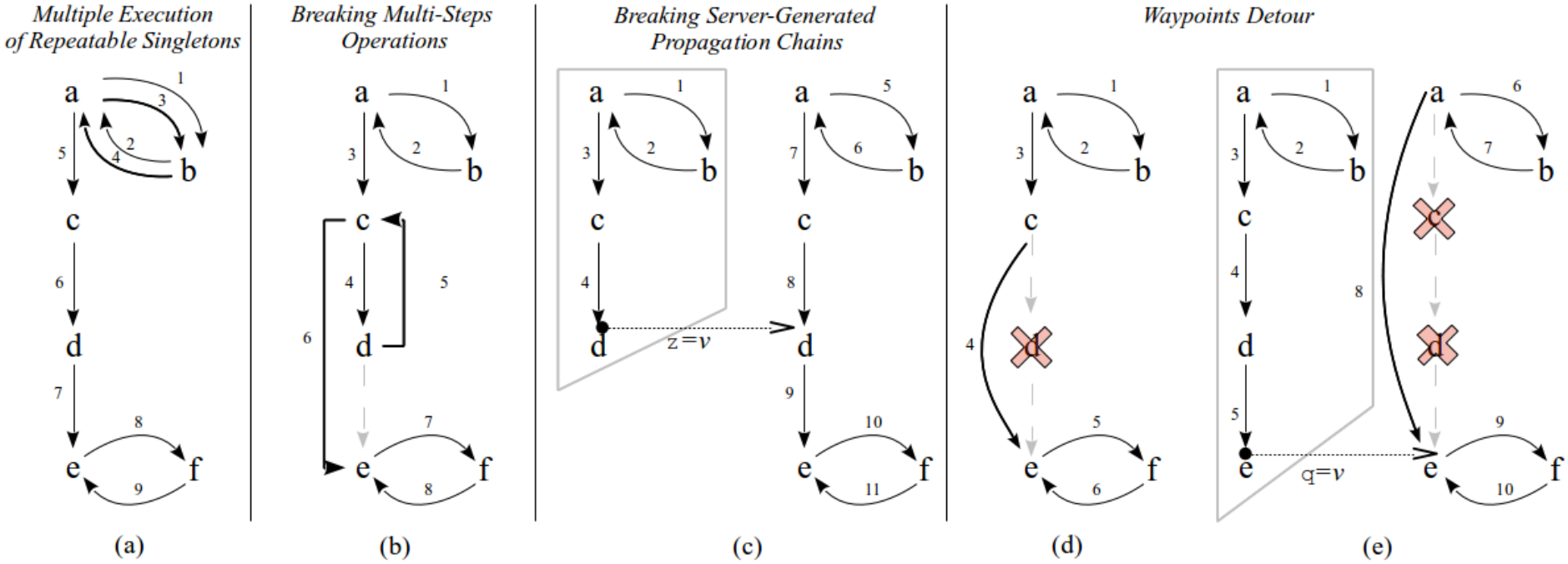
Execution

74.125.230.240 > 192.168.1.89
192.168.1.89 > 74.125.230.240
74.125.230.24 > 192.168.1.89

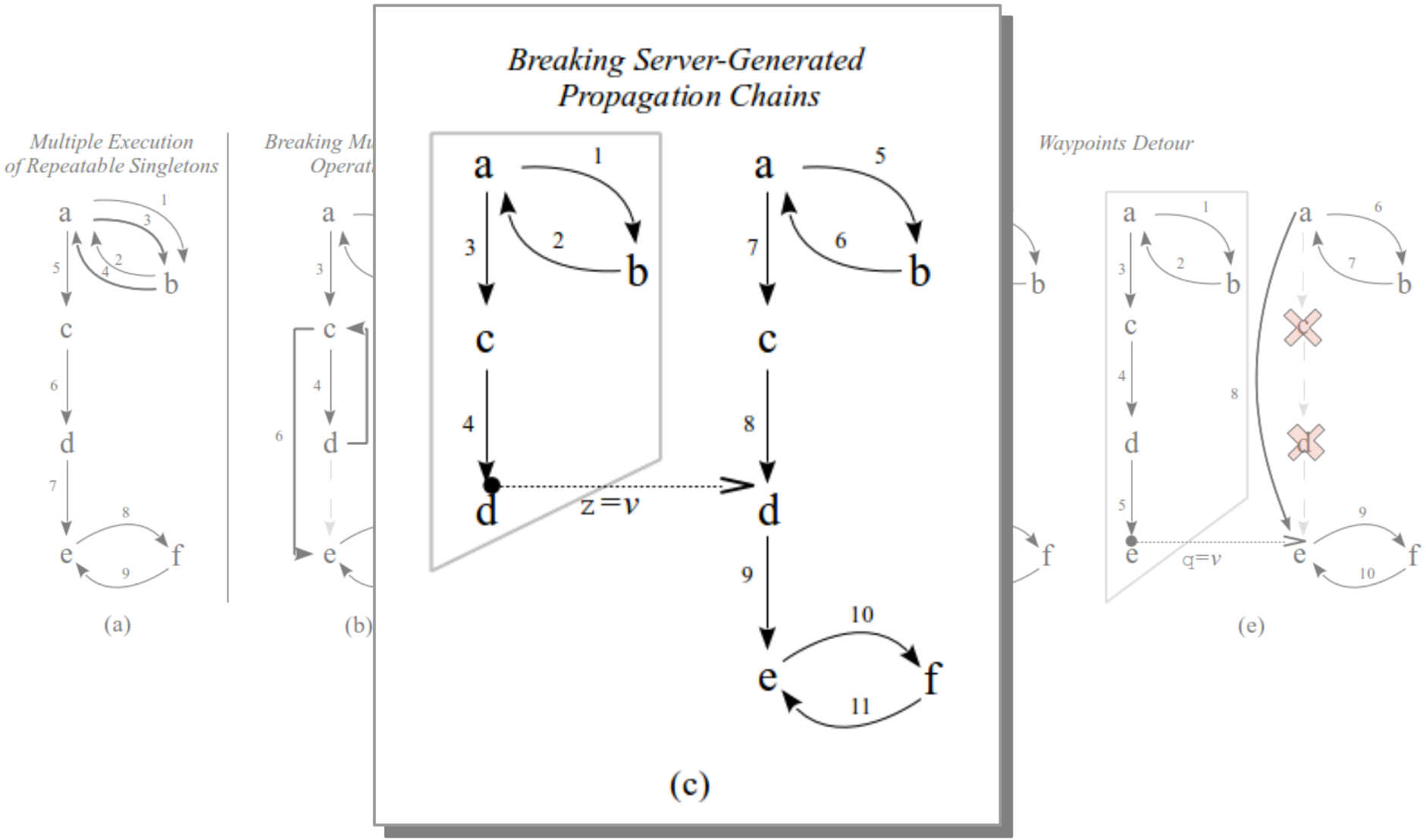
Oracle

Verdict:
Flaw found in test 1 and 2

Attack Pattern-based Test Case Generation



Attack Pattern-based Test Case Generation



Test Case Execution and Oracle

1) Model Inference

74.125.230.240 > 192.168.1.89
192.168.1.89 > 74.125.230.240
74.125.230.240 > 192.168.1.89

Resource Abstraction

Resource Clustering

2) Behavioral Patterns

Data flow Patterns

Workflow Patterns

3) Test Cases Generation

Test Cases

4) Test Cases Execution

Execution

74.125.230.240 > 192.168.1.89
192.168.1.89 > 74.125.230.240
74.125.230.240 > 192.168.1.89

Oracle

Verdict:
Flaw found in test 1 and 2

Test Case Execution and Oracle

1) Model Inference

74.125.230.240 > 192.168.1.89
192.168.1.89 > 74.125.230.240
74.125.230.240 > 192.168.1.89

Resource Abstraction

Resource Clustering

2) Behavioral Patterns

Security Property:

$$ord_{placed} \wedge onStore(S) \implies$$
$$\blacksquare(paid(U,I) \wedge toStore(S) \ \wedge$$
$$\blacksquare(ack(U,I) \wedge onStore(S)))$$

3) Test Cases Generation

Test Cases

4) Test Cases Execution

Execution

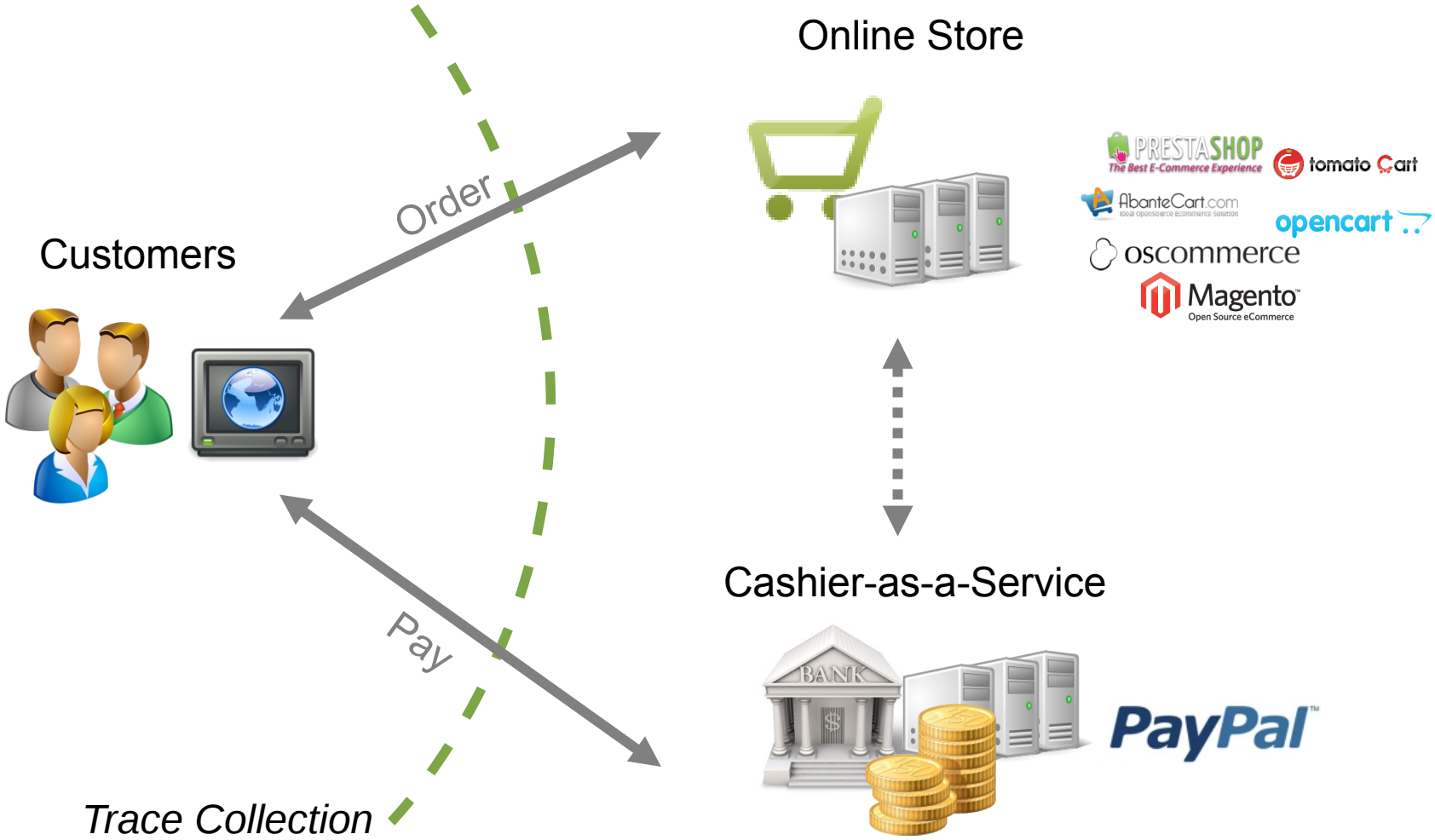
74.125.230.240 > 192.168.1.89
192.168.1.89 > 74.125.230.240
74.125.230.240 > 192.168.1.89

Oracle

Verdict:
Flaw found in test 1 and 2

Evaluation

Case Study: Shopping Cart Web Applications



Experiments and Results

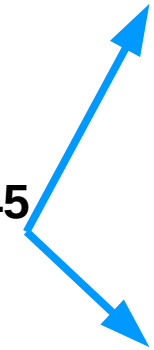
- Target: 7 popular eCommerce Web Applications
  - Deployed by >13M online stores
- Testbed: created 12 Paypal sandbox configurations
- In total **3,145 test cases**

Experiments and Results

- Target: 7 popular eCommerce Web Applications
  - Deployed by >13M online stores

- Testbed: created 12 Paypal sandbox configurations

- In total **3,145 test cases**



1,253 “misuse” detected

1,892 were executed

Experiments and Results

- Target: 7 popular eCommerce Web Applications
  - Deployed by >13M online stores

- Testbed: created 12 Paypal sandbox configurations

- In total **3,145 test cases**
  - 1,253 “misuse” detected
  - **1,892 were executed**
    - 983 not violations
    - **909 violation**

Experiments and Results

- Target: 7 popular eCommerce Web Applications
  - Deployed by >13M online stores

- Testbed: created 12 Paypal sandbox configurations

- In total **3,145 test cases**
  - 1,253 “misuse” detected
  - **1,892 were executed**
    - 983 not violations
    - **909 violation**
      - 849 caused by bugs in the GUI
      - **90 caused by logic flaws**

Vulnerabilities

- 10 previously-unknown vulnerabilities
  - Allowing to shop for free or pay less

Application	Shop for free	Pay less	Session Fixation	
AbanteCart	x			Notified Devel.
Magento				
OpenCart		x x		Notified Devel.
osCommerce	x	x		CVE-2012-2991
PrestaShop				
TomatoCart	x	x x	x	CVE-2012-4934
CS-Cart	x			CVE-2013-0118

Conclusion

Conclusion

- Proposed a black-box technique to detect logic flaws in web applications
- Combined passive model inference and attacker pattern-based test case generation
- Developed a prototype
  - assessed against 7 popular eCommerce web applications
- Discovered 10 previously-unknown logic flaws
  - allow an attacker to shop for free or pay less

References

[BalzarottiCCS07] D. Balzarotti, M. Cova, V. Felmetsger, G. Vigna,
Multi-Module Vulnerability Analysis of Web-based Applications. CCS 2007

[FelmetsgerUSENIX10] V. Felmetsger, L. Cavedon, C. Kruegel, G. Vigna
Toward Automated Detection of Logic Vulnerabilities in Web Applications. USENIX 2010

[LoweCSF97] G. Lowe
A Hierarchy of Authentication Specifications. TACAS96

[ArmandoCSF07] A. Armando, R. Carbone, and L. Compagna
LTL Model Checking for Security Protocols. CSF ’07.

[DoupèDIMVA10] A. Doupè, M. Cova, and G. Vigna
Why Johnny Can't Pentest: An Analysis of Black-Box Web Vulnerability Scanners. DIMVA2010

[WangS&P11] R. Wang, S. Chen, X. Wang, S. Qadeer
How to Shop for Free Online - Security Analysis of Cashier-as-a-Service Based Web Stores. IEEE S&P 2011

[WangS&P12] R. Wang, S. Chen, X. Wang
Signing Me onto Your Accounts through Facebook and Google: a Traffic-Guided Security Study of Commercially Deployed Single-Sign-On Web Services. IEEE S&P 2012

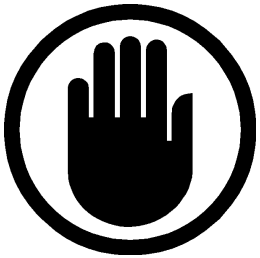
EURECOM
Sophia Antipolis

Thank you

Contact Information:

Giancarlo Pellegrino
gpellegrino@deeds.informatik.tu-darmstadt.de

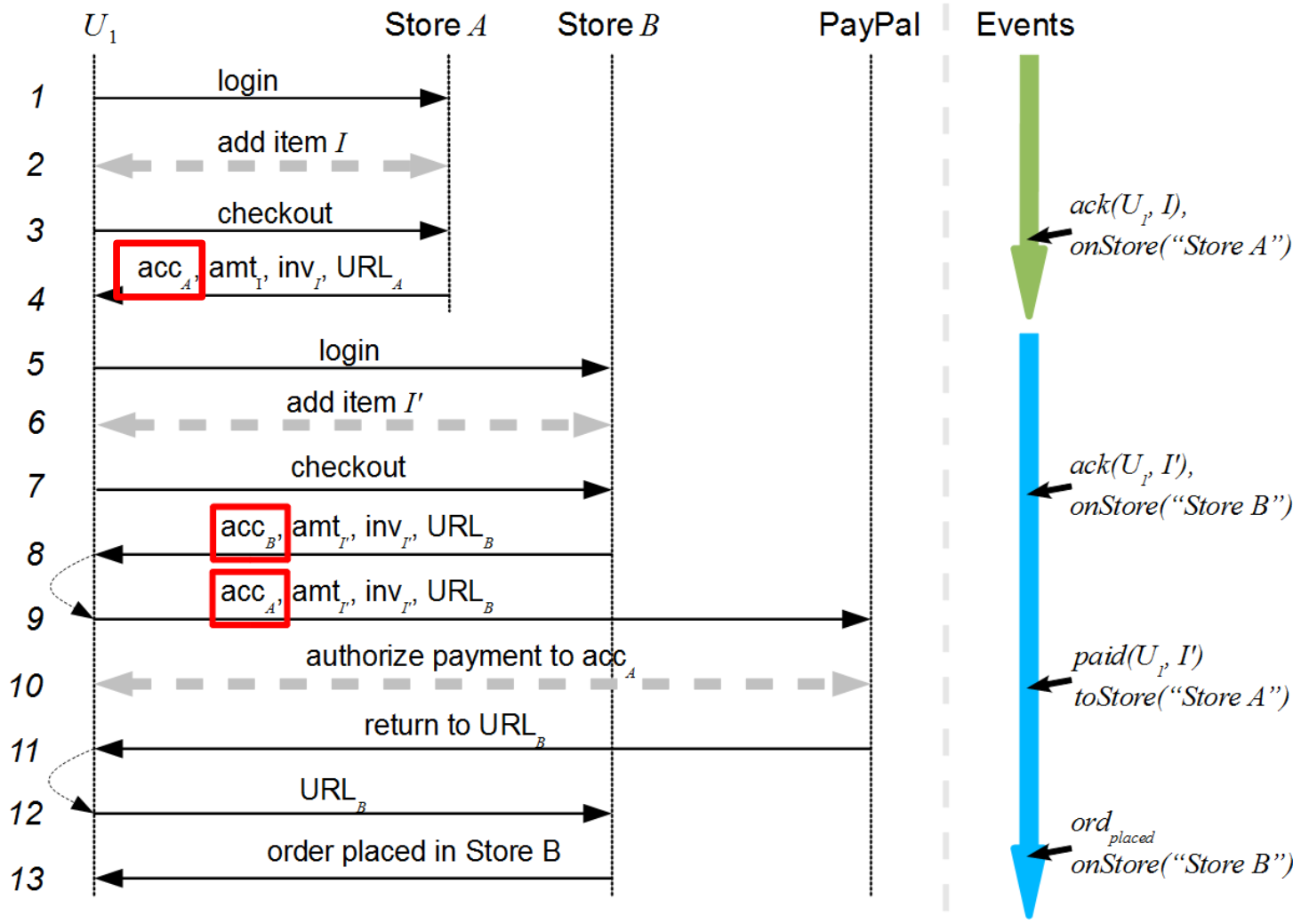
Backup slides



Results

Applications	CaaS	# Test Cases	# TC Exec.	Property Violations	
				due to Bugs	due to Vulns
AbanteCart	Std	233	74	16	1
Magento	Exp	343	240	65	-
	Std	386	210	126	-
OpenCart	Exp	173	140	46	12
	Std	135	71	30	-
osCommerce	Exp	165	117	22	20
	Std	225	128	34	1
PrestaShop	Exp	137	85	-	-
TomatoCart	Exp	302	238	65	25
	Std	224	115	24	-
CS-Cart	Exp	600	347	313	-
	Std	222	127	108	1
	Total	***3145***	***1892***	***849***	***60***

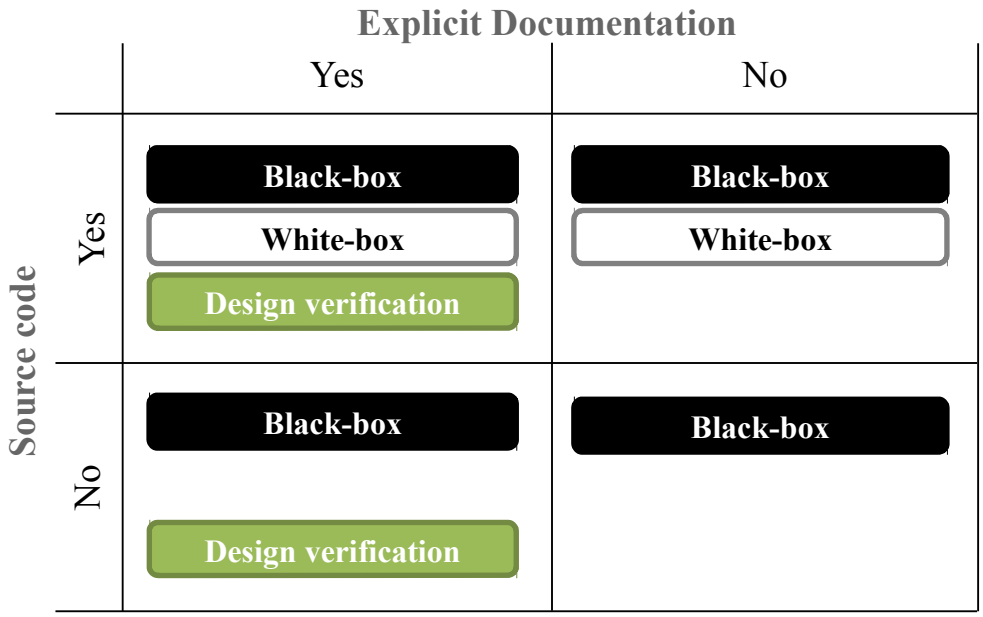
osCommerce and AbanteCart: Shopping for Free



OWASP Testing Guide v3: Manual Testing

- Understand the web application
  - Intended workflow and data flow
- Design tests to violate workflow and data flow
  - E.g., reorder steps, replay tokens, …
- Run tests and observe the results

Problem



- White-box testing [BalzarottiCCS07, FelmetsgerUSENIX10, ...]
  - Source code of WA may not be available → White-box not applicable!!
- Design verification [LoweCSF97, ArmandoCSF07, ...]
  - Specification of WA may not be available → DV not applicable!
- Black-box testing, e.g., web scanners [DoupèDIMVA10, WangS&P11, WangS&P12]
  - Cannot automatically detect logic flaws

➔ ***Testing for logic flaws is done manually***

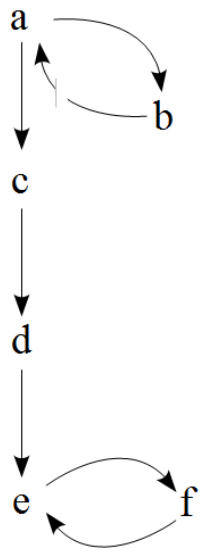
Workflow Patterns

Traces:

$$\pi_1 = \langle a, b, a, c, d, e, f, e \rangle$$

$$\pi_2 = \langle a, c, d, e, f, e \rangle$$

Model:



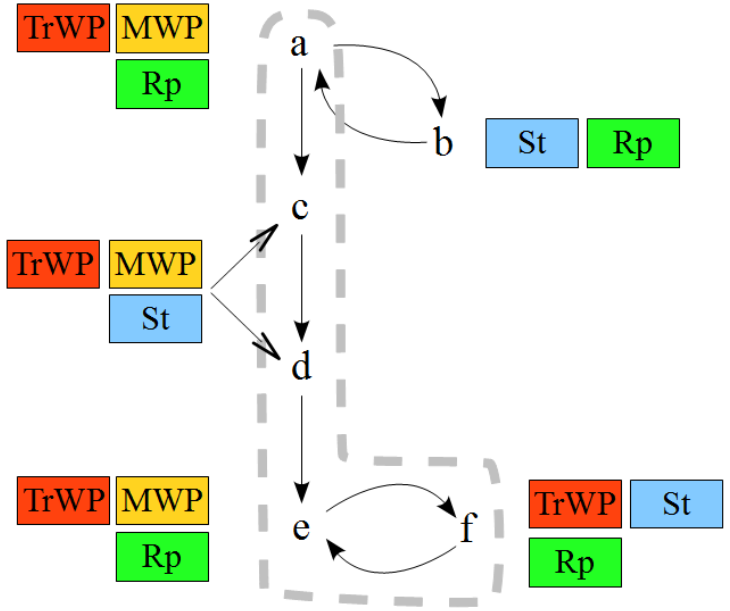
Workflow Patterns

Traces:

$$\pi_1 = \langle a, b, a, c, d, e, f, e \rangle$$

$$\pi_2 = \langle a, c, d, e, f, e \rangle$$

Model:



TrWP : Trace Waypoints

St : Singleton Nodes

: Multi-step Operations

Rp : Repeatable Operations

MWP : Model Waypoints