MACE: Detecting Privilege Escalation Vulnerabilities in Web Applications

Maliheh Monshizadeh, UIC
Prasad Naldurg, IBM Research India
V. N. Venkatakrishnan, UIC

mmonsh2@uic.edu, venkat@uic.edu, pnaldurg@in.ibm.com

UIC

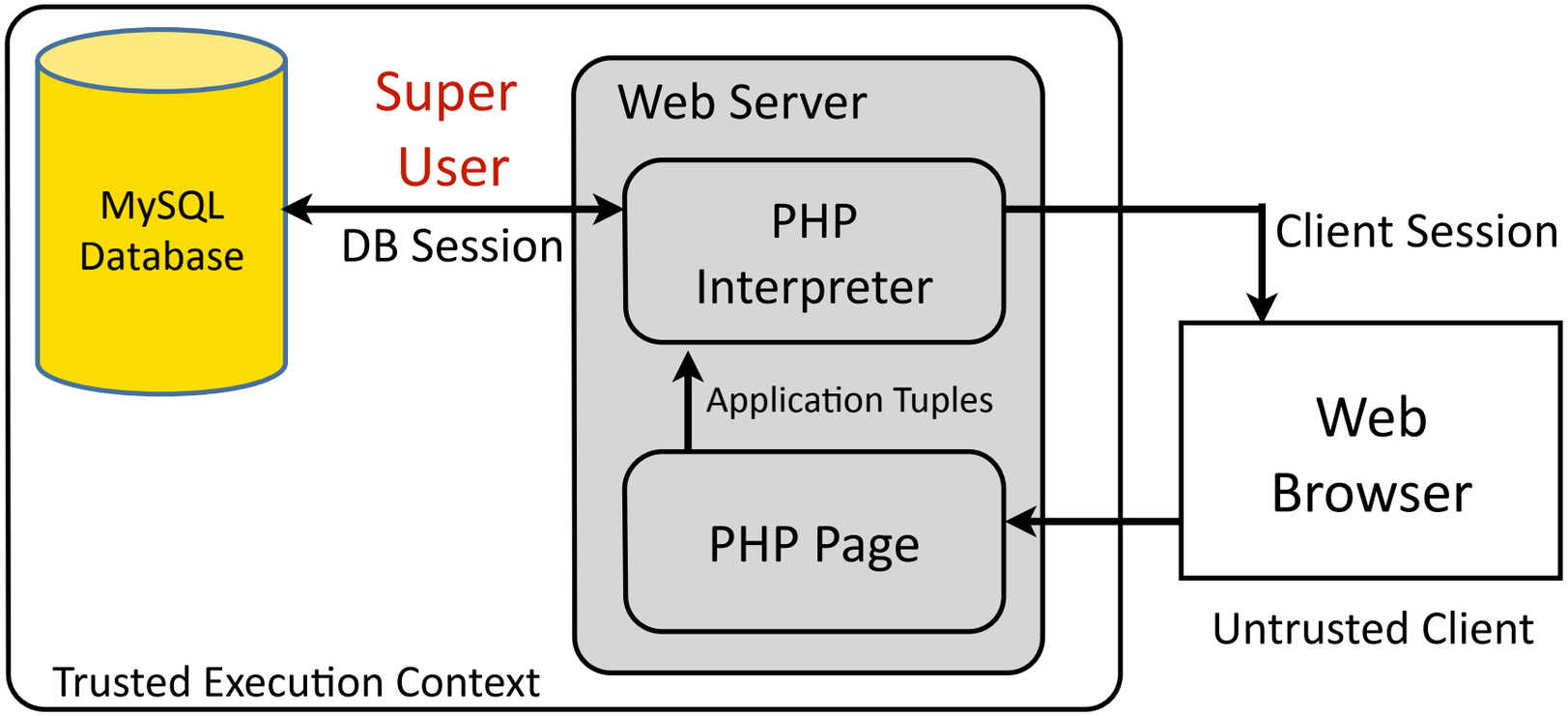
Introduction

UIC

Typical Web Application Architecture

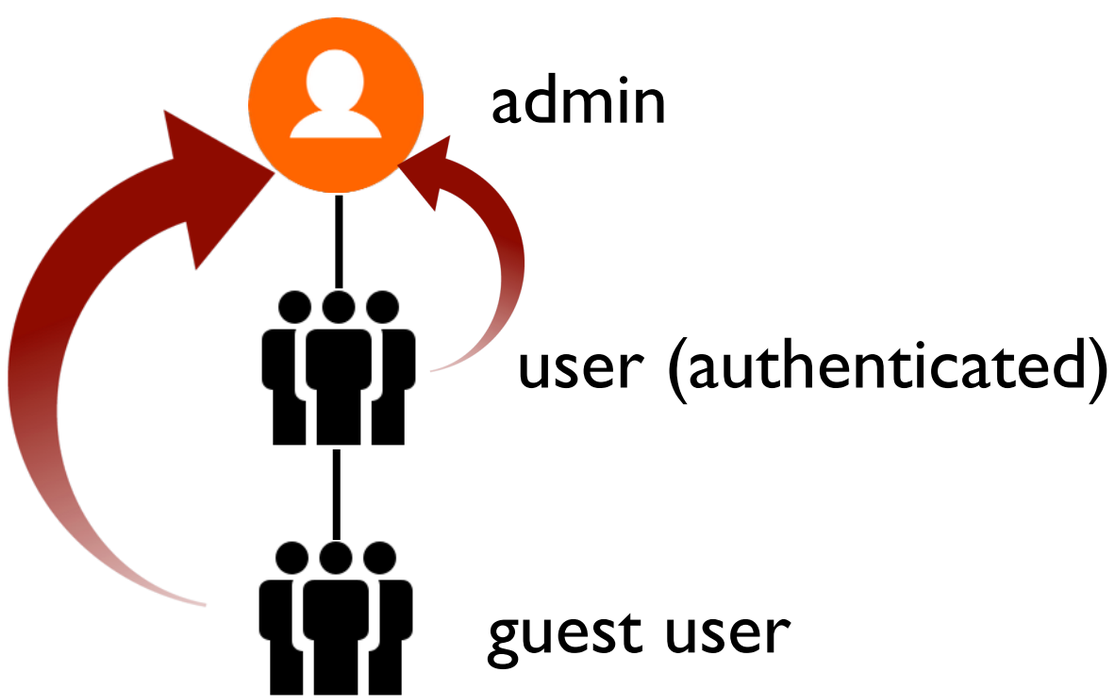
- Applications run at the privilege of the **administrator** of database
  - With highest privileges
- Typically, **no** privilege minimization

➡ Susceptible to **privilege escalation**

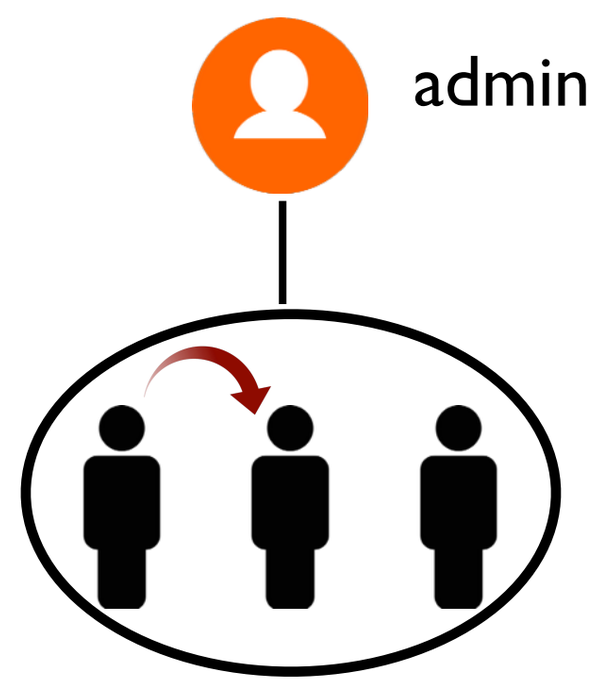


Privilege Escalation

- Vertical Privilege Escalation
  - e.g.:
    - XSS
    - Authentication Bypass



- Horizontal Privilege Escalation
- Example: Banking Website
- Can be caused by:
  - XSS, Session Fixation
  - IDs/Passwords Prediction



UIC

Vertical Priv. Escalation - Example

Untrusted Input

```
verifyUser();
if($_GET['action'] == 'deleteAll')
    query("DELETE FROM tbl_articles");
```

Any logged-in user can delete **all** articles

```
function verifyUser(){
    if(!isset($_SESSION['userID']))
        header('Location: /login.php');
    else $userID = $_SESSION['userID'];
    return;
}
```

Should check for admin credentials

UIC

Horizontal Priv. Escalation - Example

```
verifyUser();
if($_SESSION[$userID]['canWrite'] && $action == 'delete')
    query("DELETE FROM tbl_articles WHERE article_ID =
        '" + sanit($_GET['article_ID']) + "'");
```

Untrusted Input

- Guessing a valid article_ID
  - Hidden field in HTML code
  - Auto-increment

Any logged-in user can delete other users' articles

UIC

Problem & Challenges

- Missing/Improper authorization checks
  - A serious security threat for web applications
  - Facilitates an attacker to perform sensitive operations
- **Problem:** How to identify missing authorization checks?
  - **Challenge 1:** No authorization policy specification available
    - The only documentation is source code
  - **Challenge 2:** Manual analysis difficult or tedious
    - Applications are large, many thousand LOC

➡ **Goal:** must infer authorization policy from code

UIC

Inferring Access Control Rules

- How to infer policy from application, if policy implementation is not complete?
  - Intuition: For the **same** resources, in **different** access locations
    - **Same** access control rules
- Which **conditions should hold** for the user to be able to access a resource?
  - Code includes:
    - Path to the query
    - Query

UIC

Approach

- Identify **Authorization Context**
  - **Conditions** along the path to the query: Authorization **checks**
  - **Conditions** in the query: **Access parameters**
- Resource Access Analysis
  - Building an independent access model for **each resource**
  - Compare authorization contexts within each group
  - Detect **inconsistent** accesses

UIC

Why MACE?

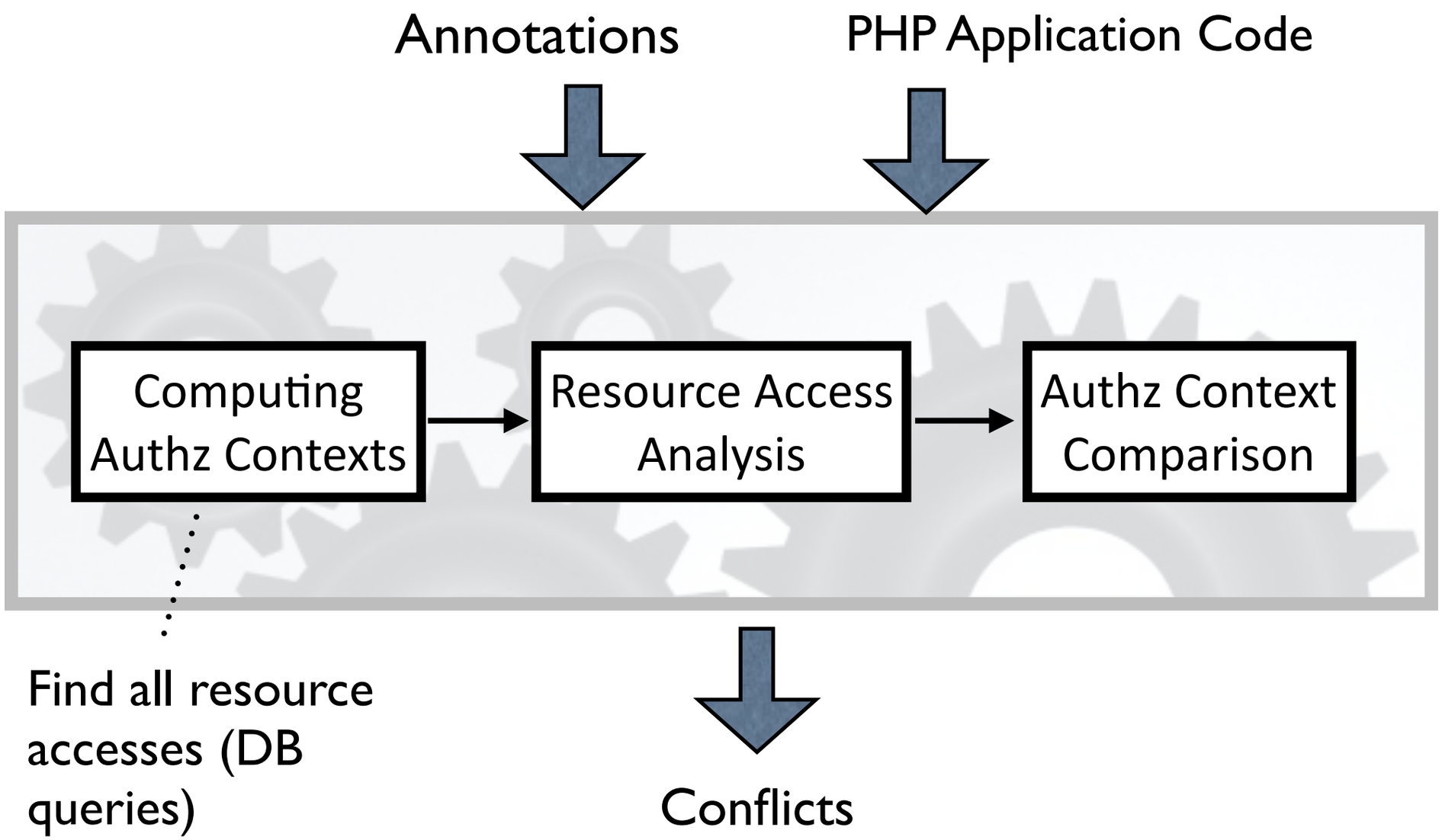
- Focuses on authorization vulnerabilities
  - A more precise and flexible authorization model for resources
- Detects both
  - Vertical privilege escalation
  - Horizontal privilege escalation
    - **First tool** for automatic detection of **Horizontal Priv. Escalation**

UIC

MACE

UIC

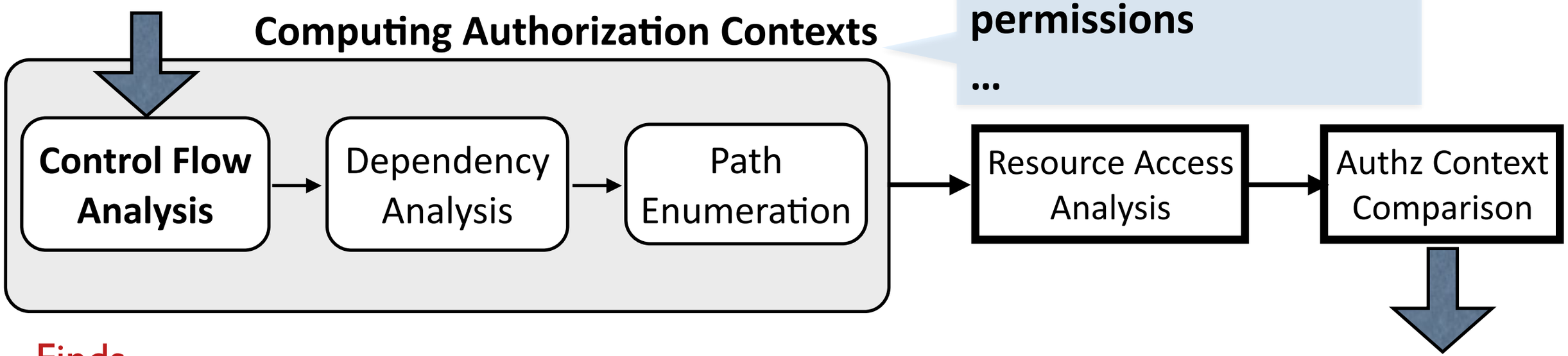
MACE



- Find all resource accesses (DB queries)
- Compute Authz Contexts for them

Computing Authorization Contexts

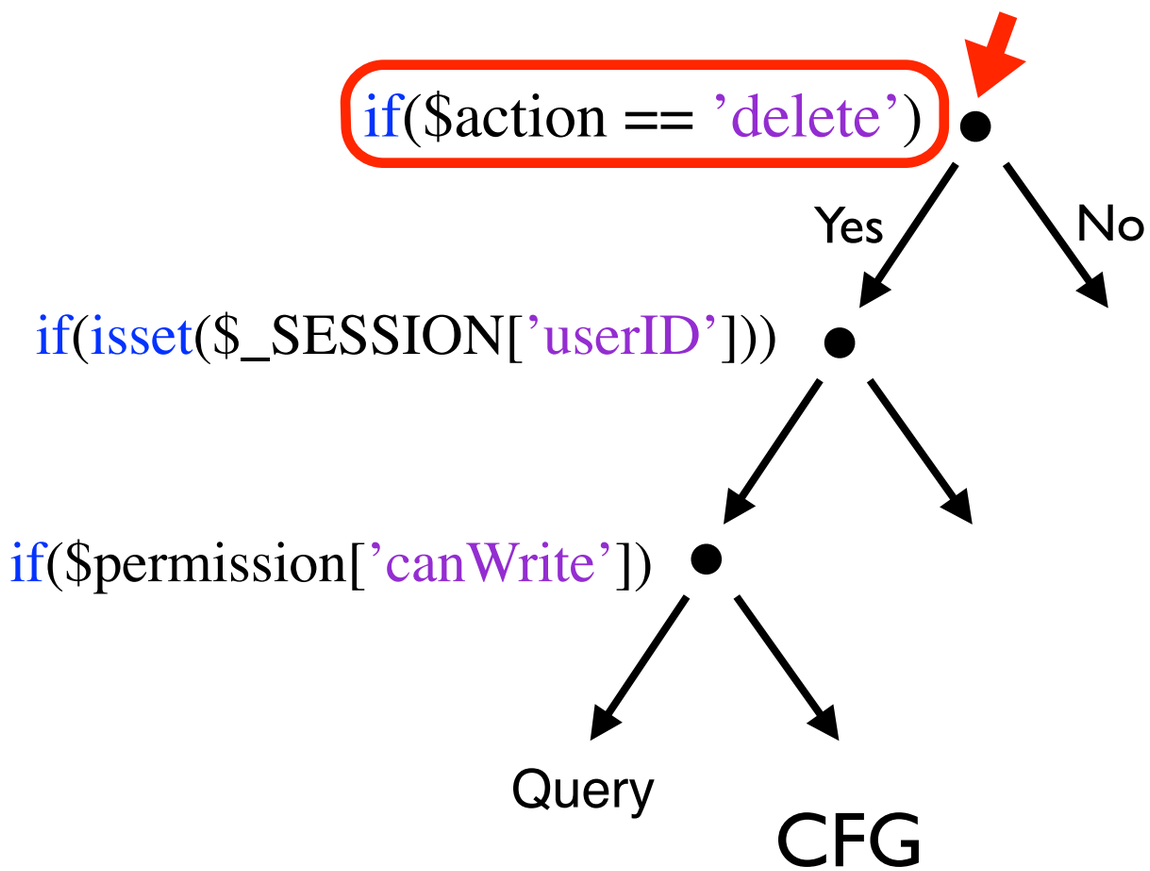
- Application Code
- Annotations



- Finds Authorization Contexts

Control Flow Analysis

- What authorization **conditions** hold at the query location?
  - **Traverse** Control Flow Graph to gather the conditions related to authorization
- Intra-procedural/Inter-procedural
- Context-sensitive, flow-sensitive



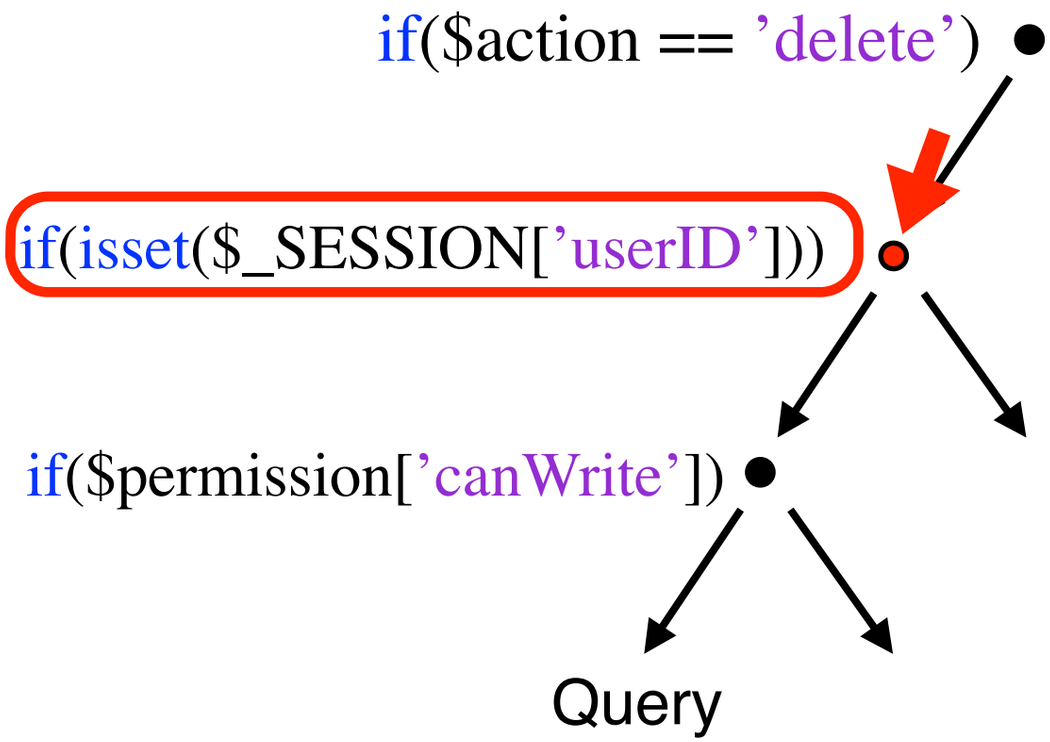
Authorization Context

Annotation

$_SESSION['userID']
$userid['level']
$permission['canWrite']

UIC

Control Flow Analysis



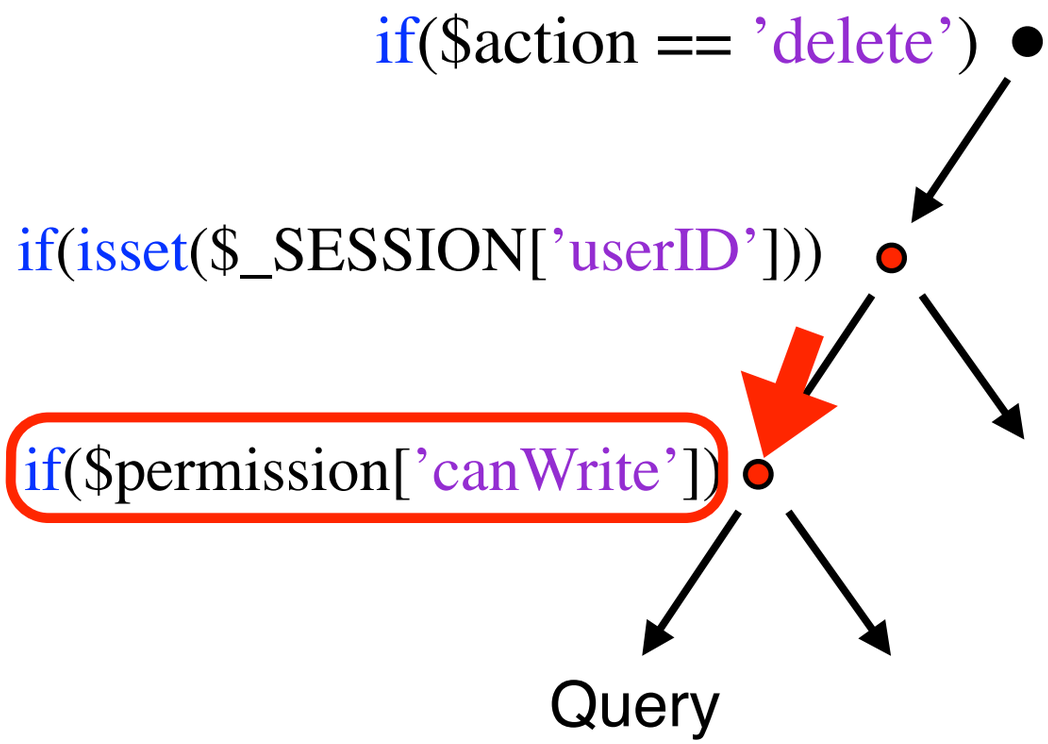
Authorization Context

isset($_SESSION['userID'])

Annotation

$_SESSION['userID']
$userid['level']
$permission['canWrite']

Control Flow Analysis



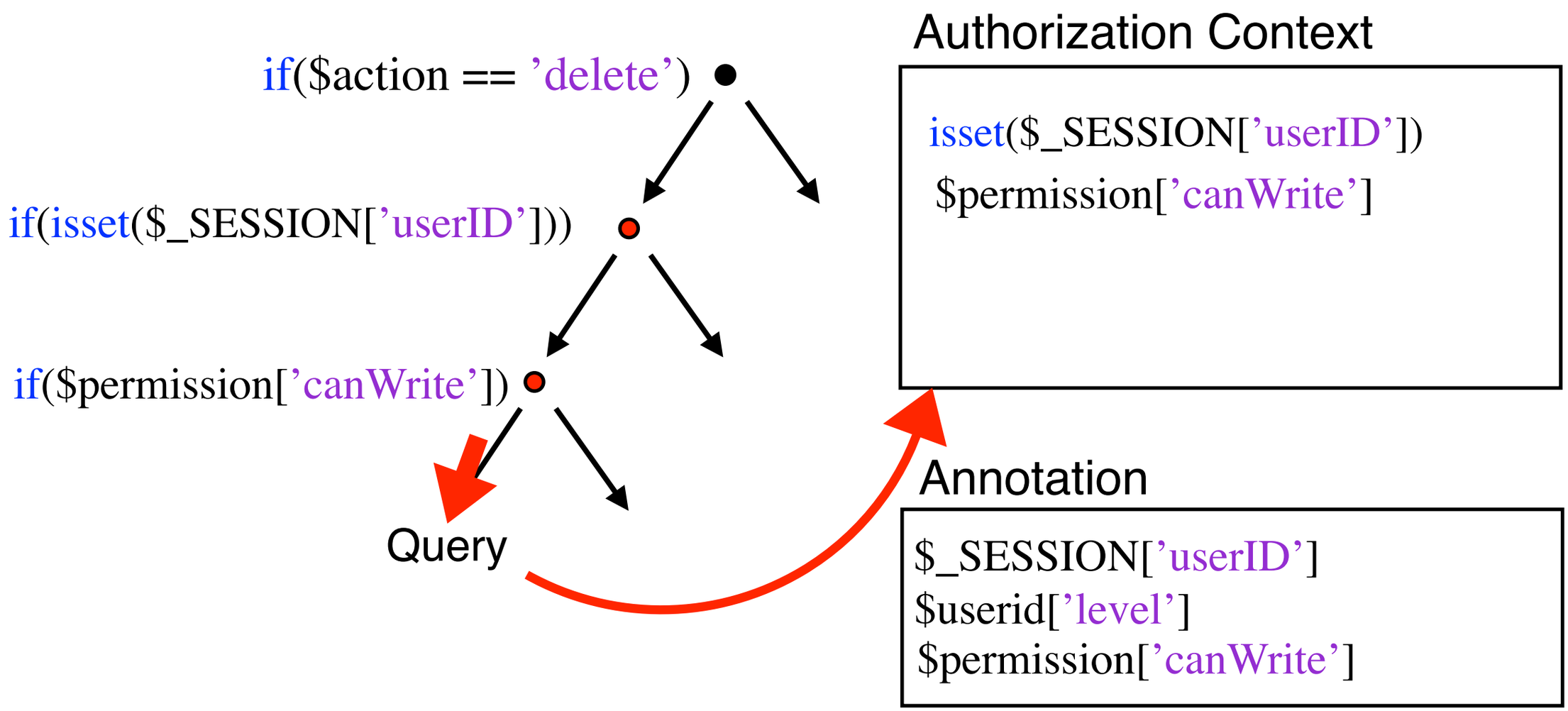
Authorization Context

```
isset($_SESSION['userID'])
$permission['canWrite']
```

Annotation

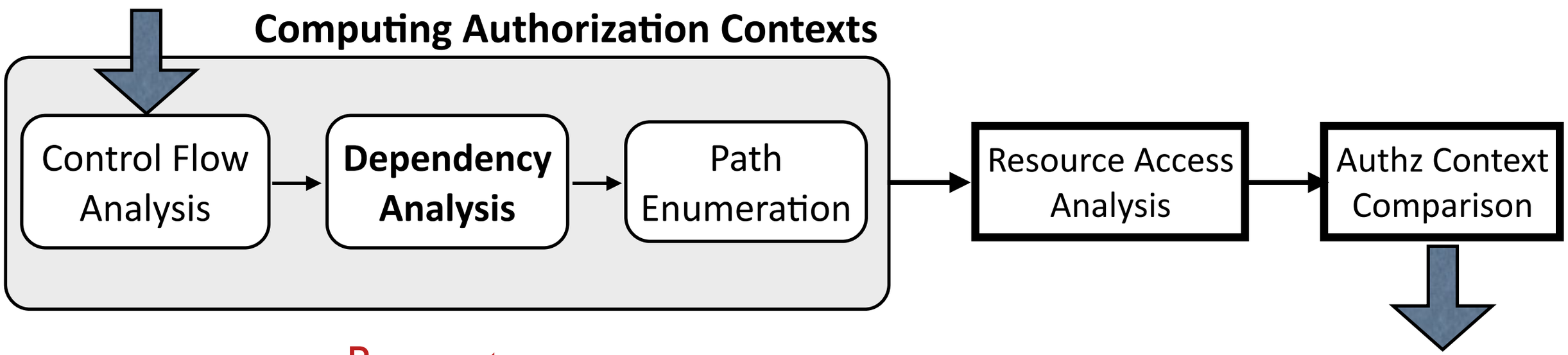
```
$_SESSION['userID']
$userid['level']
$permission['canWrite']
```

Control Flow Analysis



Computing Authorization Contexts

- Application Code
- Annotations



- Propagates Authorization Context (Values)
- Builds Symbolic Queries

Data Dependency Analysis

- Propagates Authorization Values for
  - If-statements
  - Symbolic Queries
- Intra-procedural/Inter-procedural

```
$userID = $_SESSION['userID']
query("INSERT INTO tbl_comments VALUES (
    $userID , …)");
```

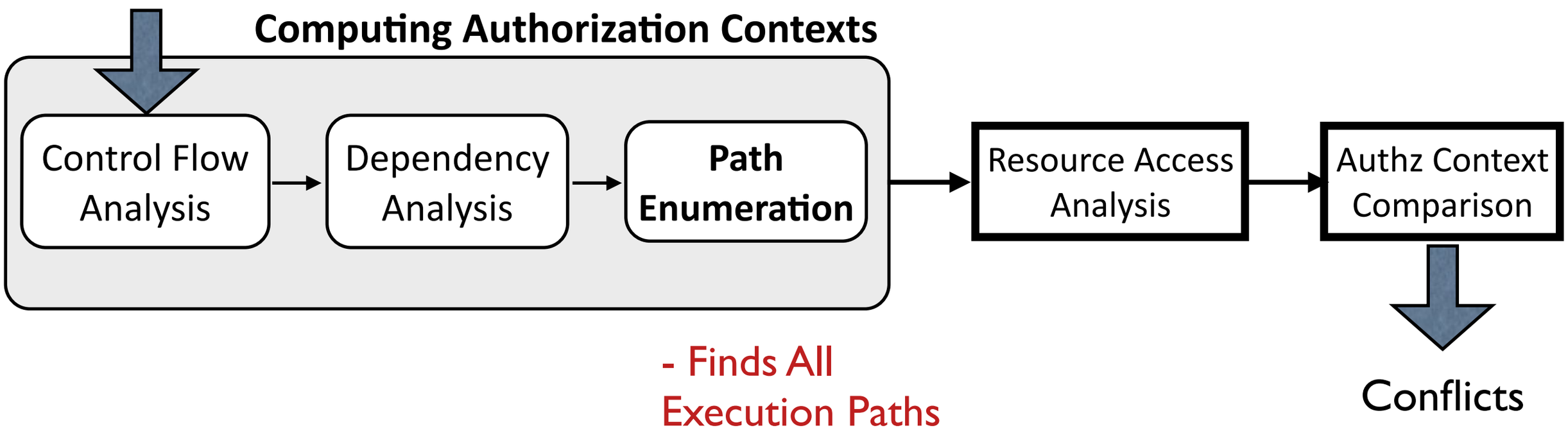
Annotation

```
$_SESSION['userID']
$userid['level']
$permission['canWrite']
$userID
```

UIC

Computing Authorization Contexts

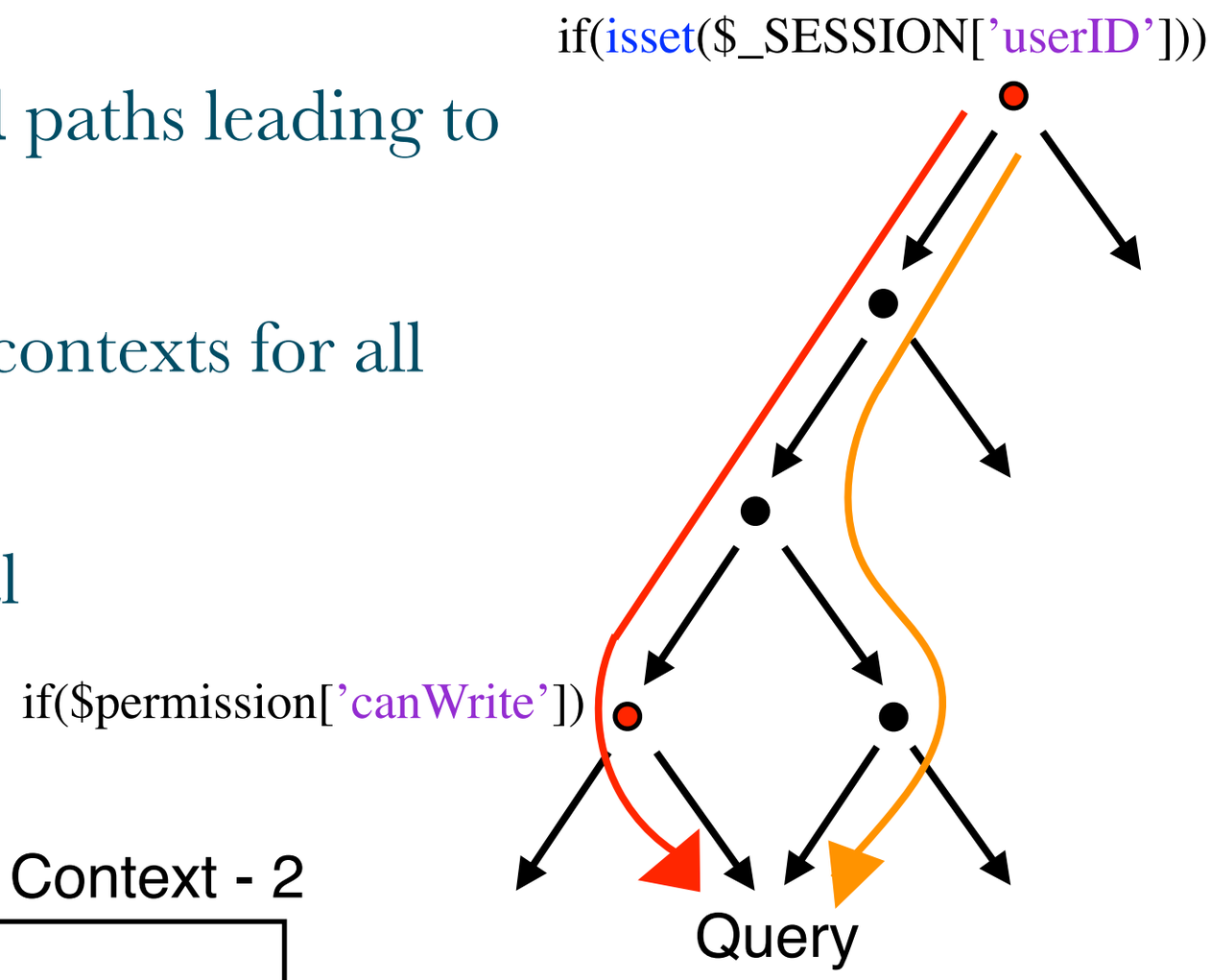
- Application Code
- Annotations



UIC

Path Enumeration

- Authorization Contexts are **path-sensitive**
- Path Enumeration enumerates **all** paths leading to queries
- Therefore we have authorization contexts for all <query, path> pairs
- Intra-procedural/Inter-procedural



Authorization Context -1

```
isset($_SESSION['userID'])
$permission['canWrite']
```

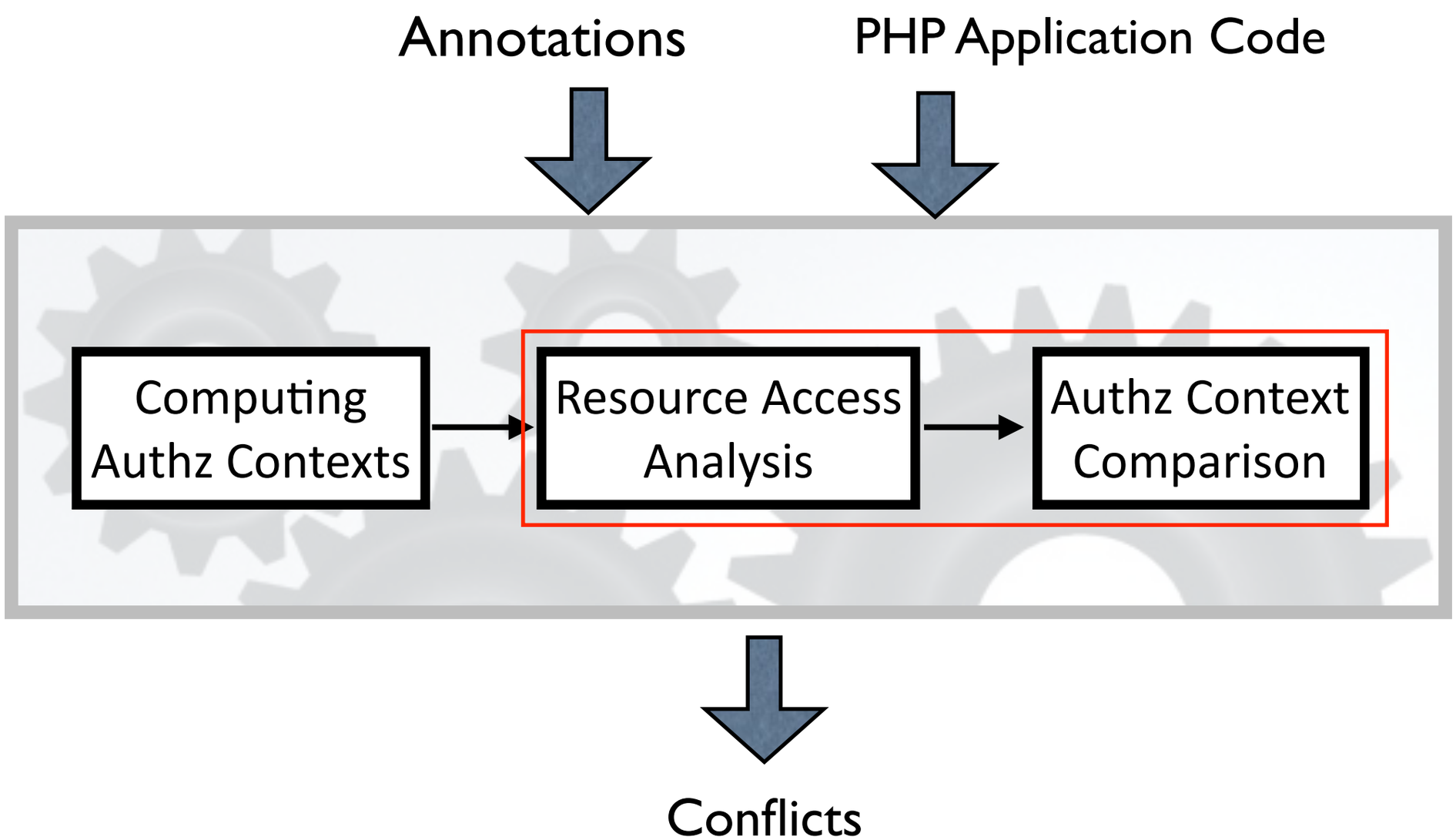
≠

Authorization Context - 2

```
isset($_SESSION['userID'])
```

UIC

MACE


Annotations
PHP Application Code
Computing Authz Contexts
Resource Access Analysis
Authz Context Comparison
Conflicts

UIC

Resource Access Analysis

- For each resource (DB table)
  - Gather resource access information
    - Authorization Context (Query-Path)
    - Access Parameters (Query WHERE Clause)
  - Compare authorization information
    - Insert-Insert, Insert-Update, Insert-Delete

```
query("INSERT INTO tbl_articles VALUES (
    sanit($_GET['article']),
    $_SESSION['userID'], …");
```

Authorization Context

```
isset($_SESSION['userID'])
$permission['canWrite']
. . . . . .
authorID = $_SESSION['userID']
```

≠

```
query("DELETE FROM tbl_articles WHERE
article_ID ='" + sanit($_GET['article_ID'])
+ "'");
```

Authorization Context

```
isset($_SESSION['userID'])
$permission['canWrite']
```

UIC

Detecting Conflicts

- What is the **rationale for comparison**?
  - INSERT queries: typically the **creator** of the data **owns** the data
- Different type of conflicts show different vulnerabilities
  - Authorization Context Conflicts (conflicts in the paths)

    ➡ Vertical privilege escalation (VPE)
  - Access Parameter Conflicts (conflicts in the WHERE clause)

    ➡ Horizontal privilege escalation (HPE)

UIC

Evaluation

UIC

Summary of Vulnerabilities

- 7 popular small to large PHP applications (1K to 90K LOC)
- **Overall 71 vulnerabilities found**
  - **66 previously unknown** vulnerabilities detected
- **5 out of 7** applications found to be vulnerable
- **Zero false positives**

Application	Description	VPE	HPE	# of Vulns
phpns	News System	✔	✔	7
DCPPortal	CMS	✔	✔	46
DNScript	Domain Trading	-	-	0
MyBloggie	Blog	✔	✔	6
MiniBloggie	Blog	✔	-	1
SCARF	Conference Management	✔	✔	11
WeBid	Online Auction	-	-	0

- **Our results demonstrate that the approach used by MACE is effective, scalable**

UIC

Identified Vulnerabilities

- DCPPortal
  - CMS
  - VPE: Use of cookies for authorization


```
if (UserValid($_COOKIE["dcp5_member_id"])) {
…
}
```

cookies are used to validate the users

Cookies are untrusted sources

  - HPE: Users can delete any agenda items from the shared calendar

```
if ((isset($_REQUEST["action"])) && ($_REQUEST["action"]=="delete")) {
    $sql = "DELETE FROM $t_agenda WHERE id = '".$_REQUEST["agid"]."'";
```

untrusted source

UIC

Precision

- False Positives?
  - **Multiple** INSERT Queries with **inconsistent** Authorization Contexts
    - Which one is correct?
- False Negatives
  - SELECT Queries
    - Disclosure policy is not available
  - INSERT queries with empty authorization contexts
    - Subsequent DELETE & UPDATE queries undetected

UIC

Performance & Scalability

- Application size: 1K to 90K
- Analysis Time
  - 95%: Dependency Analysis/Path Enumeration
  - Resource Access Analysis

	SLOC	# DB Tables	Analysis Time (s)
phpns	4.5K	13	8220
DCPPortal	**89K**	34	982
DNScript	1.5K	7	**35093**
MyBloggie	6K	5	373
MiniBloggie	1.5K	**2**	**35**
SCARF	**1K**	7	54
WeBid	27K	**47**	1492

UIC

Related Work

- Prevention of Authorization Vulnerabilities
  - New Development Frameworks: Capsules.
  - Run-time Frameworks: **Nemesis**[5], **Diesel**[9], **Swaddler**[4], **CLAMP**[10].
  - MACE is focused on static detection of authorization vulnerabilities
- Finding Authorization Bugs in Legacy Web Applications
  - e.g.: **RoleCast**[11]
  - MACE uses a fine-grained access model
- General Vulnerability Analysis in Applications
  - Using Program Invariants, Behavioral Patterns
  - e.g.: **Waler**[7], **AutoISES**[14], **NoTamper**[1]
  - MACE focuses on authorization vulnerabilities —> better precision

UIC

Conclusion

- **Missing/improper** authorization a serious problem for Web applications
- MACE:
  - Best-effort tool for **automatic** detection of HPE, VPEs
  - **71 total, 66 unknown** vulnerabilities were found
  - Reduces **weeks of testing** effort
- Lack of documentation about access control policies makes it challenging to detect vulnerabilities
- MACE demonstrates that it is possible to infer policies from code and use them for vulnerability detection

UIC

Questions?

UIC

References - I

1. BISHT, P., HINRICHS, T., SKRUPSKY, N., BOBROWICZ, R., and VENKATAKRISHNAN, V. N. **Notamper: Automatic blackbox detection of parameter tampering opportunities in web applications.** In Proceedings of the 17th ACM Conference on Computer and Communications Security (New York, NY, USA, 2010), CCS '10, ACM.

2. BISHT, P., HINRICHS, T., SKRUPSKY, N., and VENKATAKRISHNAN, V. N. **Waptec: Whitebox analysis of web applications for parameter tampering exploit construction.** In Proceedings of the 18th ACM Conference on Computer and Communications Security (New York, NY, USA, 2011), CCS '11, ACM, pp. 575–586.

3. BISHT, P., SISTLA, A. P., and VENKATAKRISHNAN, V. N. **Taps: Automatically preparing safe sql queries.** In Proceedings of the 17th ACM Conference on Computer and Communications Security (New York, NY, USA, 2010), CCS '10, ACM, pp. 645–647.

4. COVA,M.,BALZAROTTI,D.,FELMETSGER,V., and VIGNA,G. **Swaddler: An approach for the anomaly-based detection of state violations in web applications.** In Proceedings of the 10th International Conference on Recent Advances in Intrusion Detection (Berlin, Heidelberg, 2007), RAID'07, Springer-Verlag, pp. 63–86.

5. DALTON,M.,KOZYRAKIS,C., and ZELDOVICH,N. **Nemesis: Preventing authentication & access control vulnerabilities in web applications.** In USENIX Security Symposium (2009), USENIX Association, pp. 267–282.

UIC

References - II

6. ENGLER, D., CHEN, D.Y., HALLEM, S., CHOU, A., and CHELF, B. **Bugs as deviant behavior: A general approach to inferring errors in systems code.** In Proceedings of the Eighteenth ACM Symposium on Operating Systems Principles (New York, NY, USA, 2001), SOSP '01, ACM, pp. 57–72.

7. FELMETSGER,V., CAVEDON, L., KRUEGEL,C., and VIGNA, G. **Toward automated detection of logic vulnerabilities in web applications.** In Proceedings of the 19th USENIX Conference on Security (Berkeley, CA, USA, 2010), USENIX Security'10, USENIX Association, pp. 10–10.

8. JOVANOVIC,N.,KRUEGEL,C.,and KIRDA,E. **Pixy: A static analysis tool for detecting web application vulnerabilities** (short paper). In IN 2006 IEEE SYMPOSIUM ON SECURITY AND PRIVACY (2006), pp. 258–263.

9. FELT, A.P., FINIFTER, M., WEINBERGER, J., and WAGNER, D. **Diesel: Applying privilege separation to database access.** In Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security (New York, NY, USA, 2011), ASIACCS '11, ACM.

10. PARNO, B., MCCUNE, J.M., WENDLANDT, D., ANDERSEN, D.G., and PERRIG, A. **Clamp: Practical prevention of large-scale data leaks.** In Proceedings of the 2009 30th IEEE Symposium on Security and Privacy (Washington, DC, USA, 2009), SP '09, IEEE Computer Society, pp. 154–169.

UIC

References - III

11. SON, S., MCKINLEY, K.S., and SHMATIKOV, V. **Rolecast: finding missing security checks when you do not know what checks are.** In Proceedings of the 2011 ACM international conference on Object oriented programming systems languages and applications (New York, NY, USA, 2011), OOPSLA '11, ACM, pp. 1069–1084.

12. SRIVASTAVA,V., BOND, M.D., MCKINLEY, K.S., and SHMATIKOV, V. **A Security Policy Oracle: Detecting Security Holes using Multiple API Implementations.** In PLDI'11: Proceedings of the 2011 ACM SIGPLAN Conference on Programming Language Design and Implementation (San Jose, CA, USA, 2011).

13. SUN, F., XU, L., and SU, Z. **Static detection of access control vulnerabilities in web applications.** In Proceedings of the 20th USENIX conference on Security (Berkeley, CA, USA, 2011), SEC'11, USENIX Association, pp. 11–11.

14. TAN, L., ZHANG, X., MA, X., XIONG, W., and ZHOU, Y. **Autoises: Automatically inferring security specifications and detecting violations.** In Proceedings of the 17th Conference on Security Symposium (Berkeley, CA, USA, 2008), SS'08, USENIX Association, pp. 379–394.

UIC

Annotation

- How to identify the authorization variables?
  - MACE Extracts of Globals/SuperGlobals
  - Developers Confirm
  - Examples:
    - $_SESSION['user_id']
    - $userid['level']
    - $permission['can_write']

UIC

Annotation Example - MyBloggie

- $HTTP_SERVER_VARS[REMOTE_ADDR]
- $_SERVER[REMOTE_ADDR]
- $_SERVER[HTTP_HOST]
- $_SERVER[SERVER_NAME]
- $_FILES[userfile][name]
- $_FILES[userfile][type]
- $_FILES[userfile][size]
- $_FILES[userfile][tmp_name]
- **$_SESSION[username]**
- **$_SESSION[user_id]**
- **$_SESSION[passwd]**

UIC

Precision - False Positives?

- **Multiple** INSERT Queries with **inconsistent** Authorization Contexts
- Which one is correct?
  - Logged-in users can insert
  - Guest users can insert

```
query("INSERT INTO tbl_comments VALUES (
    $_SESSION['userID'],
    …)");
```

Authorization Context

```
isset($_SESSION['userID'])
```

≠

```
query("INSERT INTO tbl_comments VALUES (
    $_SERVER['REMOTE_ADDR'], …)");
```

Authorization Context

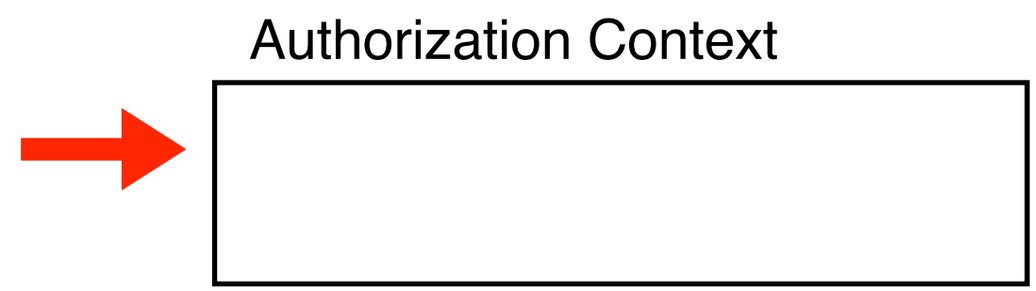
- Mitigation technique: Comparison with the strongest context

UIC

Precision - False Negatives

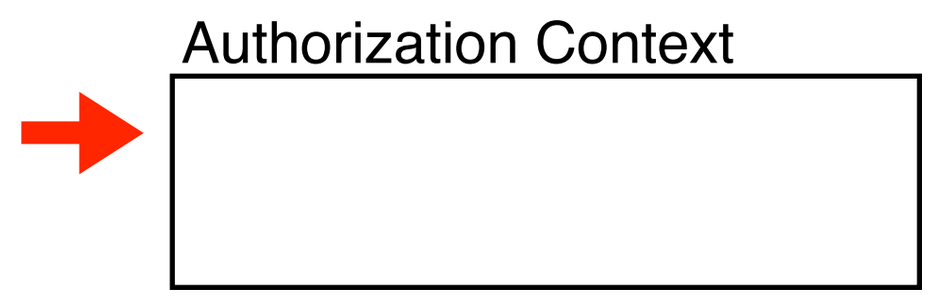
- SELECT Queries
- Missing Authorization Information in INSERTs

```
query("INSERT INTO tbl_articles VALUES (
    sanit($_GET['article_code']),
    $_GET['userID'],
    sanit($_GET['article_msg']))");
```



=

```
query("DELETE FROM tbl_articles WHERE
article_ID ='" + sanit($_GET['article_ID']) + "'");
```



- MACE reports the insert queries which have empty contexts

UIC

Detected Violations

Application	Number of Violations		
	insert-insert	insert-update	insert-delete
phpns	0	5	2
DCPPortal	0	21	25
DNScript	0	0	0
MyBloggie	0	3	3
MiniBloggie	0	0	1
SCARF	1	8	3
WeBid	0	0	0

UIC