Fix Me Up:
Repairing Access-Control Bugs in Web Applications

Sooel Son
Kathryn S. McKinley     Vitaly Shmatikov
UT Austin and Microsoft Research
... a developer exposes a reference to an internal implementation object... Without an access control check, attackers can access unauthorized data.
Access–control bugs (2)

(From WhiteHat Website Security Report 2012)

Figure 3. Top Ten Vulnerability Classes (2011)
Percentage likelihood that at least one serious* vulnerability will appear in a website
Forced-browsing attack


http://host/delete.php?id=victim_id
Make sure that every entry is locked with the proper access-control logic.
About FixMeUp

Static program transformation tool for finding and fixing access-control bugs in PHP applications

Given an example of correct access control ... 

1. Finds calling contexts that do not implement the correct access-control logic
2. Produces candidate repaired code that prevents forced-browsing attacks
FixMeUp workflow

- Example of access-control logic
- Access-control template (ACT)
- Apply ACT to fix the bugs
- Validate the repairs
1. Example of access-control logic

```php
lockSession();
if(!empty($_SESSION['name']) && !empty($_SESSION['pass'])) {
    ...
    $logined = @mysql_affected_rows();
}
if($logined !== 1 && !empty($_COOKIE[COOKIE_USER]) && !empty($_COOKIE[COOKIE_PASS])) {
    ...
    $logined = @mysql_affected_rows();
}
if($logined !== 1) {
    unlockSessionAndDestroyAllCokies();
    sleep(5);
    header('Location: ' . QUERY_STRING_BLANK . 'login');
    die();
}
```
2. Access-control template

- Compute an ACT
  - Stat 1
  - Stat 2
  - Stat 3
  - Invoke B
    - Stat 4
    - Stat 5
    - Stat 6
    - Invoke C
      - Stat 7
      - Stat 8
      - Access-control check
      - Exit
  - Access-control check
  - Exit
3. Apply ACT to fix the bugs

- Finds vulnerable contexts that do not implement the same logic as the ACT
3. Apply ACT to fix the bugs (2)

- Replicates ACT into the vulnerable context while reusing already existing statements.
4. Validate repairs

- Recompute ACT - should be the same as before!

No match? Issue a warning
Evaluation

- 10 open-source interactive PHP server apps
- Generated 38 repairs
  - 31 correct
  - 7 in addition to already existing access-control logic
- 28 partial repairs
  - Reusing existing statements is important!
- 1 warning
- 1 unwanted side effect
Evaluation (2)

```php
include('class/common.php') ; // [FixMeUp repair]
$GR_newone = new COMMON( ) ; // [FixMeUp repair]
if (( $ SESSION [ 'no' ] != 1)) { // [FixMeUp repair]
    $GR_newone ->error( 'Require admin privilege' , 1 , 'CLOSE' ) ; // [FixMeUp repair]
}
```

```php
include('class/common.php') ; // existing statement
$GR = new COMMON( ) ; // existing statement
if (( $ SESSION [ 'no' ] )) { // [FixMeUp repair]
    $GR->error( 'Require login procedure' ) ; // [FixMeUp repair]
}
```

```php
......
//@SSO( 'member')
@fwrite($tmpfs, $saveResult);
```
Warning: after applying the ACT, repaired code does not implement the same logic as the ACT

Program Entry

```php
include 'conf.php';
session_start(); // existing statement

if ($confirm=="" ) {
}
else if( $confirm== "yes") {
  dbConnect(); // existing statement
  if ( !verifyuser ( ) ) // [FixMeUp repair]
  {
      header('Location: ./login.php');//[FixMeUp repair]
      exit; // [FixMeUp repair]
  }

  $sql = "DELETE FROM blogdata WHERE postid = $postid";
  $query = mysql_query( $sql ) or die( "Cannot query the database .<br>" .mysql_error() );
  ....
}
```
Limitations

- Environmental data dependencies, eval
- Unwanted side effects

Diagram:
- Entry Stat 1
- Entry Stat 2
- Entry Access-control check
- Entry exit
- Entry Sensitive Operation

?
Avoiding unwanted side effects

- Use fresh variable names
  
  $local_var_1 = session_id()

  $local_var_1_new = session_id()

- Do not replicate already existing statements

  entry session_start()

  entry include "a.php"

  entry Access-control check

  entry exit

  entry session_start()

  entry include "a.php"

  entry Sensitive Operation
Related work

- Static detection of access-control bugs
- Dynamic detection of access-control bugs
- Dynamic repair of software bugs
Conclusion

- FixMeUp computes code templates for access-control logic from examples
- Finds and repairs access-control bugs in PHP applications
  - Reuses existing statements
  - Avoids introducing unwanted dependences
- Successfully repaired 30 access-control bugs in 10 real-world PHP applications
Q & A

Thank you