SYNTHDB: Synthesizing Database via Program Analysis for Security Testing of Web Applications

An Chen[1], JiHo Lee[2], Basanta Chaulagain[1], Yonghwi Kwon[2], Kyu Hyung Lee[1]

[1]University of Georgia, [2]University of Virginia

Security Testing of Web Applications

- **Static analysis**
  - Insufficient for dynamic features introduced by PHP
  - High False Positive rate

- **Dynamic analysis**
  - Low code coverage
  - Require concrete external resource, e.g., database.
  - Obtaining a testing database is difficult in practice

Our Approach

- **SYNTHDB:**
  - A program analysis-based database generator
  - Automatic analysis
  - Initial database is not required
  - Improving security testing

Existing techniques for synthesizing database

Example application code(Schoolmate)

```
1   $q1 = query("SELECT courseid FROM registrations
                 WHERE studentid = '$_POST['student']'");
2   while($reg = fetch_array($q1)){
3       $q2 = query("SELECT dotw,coursename FROM courses
                     WHERE courseid = '$reg[0]'");
4       while($course = fetch_array($q2)){
5           switch($course[0]){
6           case 'M':
…                   ...
7                   print("Monday:".$course[1])
…                   ...
8           case ...
            }
        }
    }
```

Existing techniques for synthesizing database

Retrieve registered course by student id

Example application code(Schoolmate)

```
1  $q1 = query("SELECT courseid FROM registrations
                WHERE studentid = '$_POST['student']'");
2  while($reg = fetch_array($q1)){
3      $q2 = query("SELECT dotw,coursename FROM courses
                    WHERE courseid = '$reg[0]'");
4      while($course = fetch_array($q2)){
5          switch($course[0]){
6          case 'M':
…                  ...
7                  print("Monday:".$course[1])
…                  ...
8          case ...
           }
       }
}
```

Existing techniques for synthesizing database

If student registered any course

Example application code(Schoolmate)

```
1   $q1 = query("SELECT courseid FROM registrations
                 WHERE studentid = '$_POST['student']'");
2   while($reg = fetch_array($q1)){
3       $q2 = query("SELECT dotw,coursename FROM courses
                     WHERE courseid = '$reg[0]'");
4       while($course = fetch_array($q2)){
5           switch($course[0]){
6           case 'M':
…                   ...
7                   print("Monday:".$course[1])
…                   ...
8           case ...
            }
        }
}
```

Existing techniques for synthesizing database

Retrieve course information by course id

Example application code(Schoolmate)

```
1   $q1 = query("SELECT courseid FROM registrations
                 WHERE studentid = '$_POST['student']'");
2   while($reg = fetch_array($q1)){
3       $q2 = query("SELECT dotw,coursename FROM courses
                     WHERE courseid = '$reg[0]'");
4       while($course = fetch_array($q2)){
5           switch($course[0]){
6           case 'M':
…                   ...
7                   print("Monday:".$course[1])
…                   ...
8           case ...
            }
        }
    }
```

Existing techniques for synthesizing database

If course exists, display weekly course schedule

Example application code(Schoolmate)

```
1   $q1 = query("SELECT courseid FROM registrations
                 WHERE studentid = '$_POST['student']'");
2   while($reg = fetch_array($q1)){
3       $q2 = query("SELECT dotw,coursename FROM courses
                     WHERE courseid = '$reg[0]'");
4       while($course = fetch_array($q2)){
5           switch($course[0]){
6           case 'M':
…                   ...
7                   print("Monday:".$course[1])
…                   ...
8           case ...
            }
        }
    }
```

Existing techniques for synthesizing database

Example application code(Schoolmate)


```
1   $q1 = query("SELECT courseid FROM registrations
                 WHERE studentid = '$_POST['student']'");
2   while($reg = fetch_array($q1)){
3       $q2 = query("SELECT dotw,coursename FROM courses
                     WHERE courseid = '$reg[0]'");
4       while($course = fetch_array($q2)){
5           switch($course[0]){
6           case 'M':
…                   ...
7                   print("Monday:".$course[1])
…                   ...
8           case ...
            }
        }
    }
```

Existing techniques for synthesizing database

Example schema

Table: course

Column	Type
courseid	INT(11) AUTO INCREMENT
coursename	VARCHAR(20)
dotw	VARCHAR(5)

Table: registration

Column	Type
regid	INT(11) AUTO INCREMENT
courseid	INT(11)
studentid	INT(11)

Existing techniques for synthesizing database

Example schema

Table: course

Column	Type
courseid	INT(11) AUTO INCREMENT
coursename	VARCHAR(20)
dotw	VARCHAR(5)

Table: registration

Column	Type
regid	INT(11) AUTO INCREMENT
courseid	INT(11)
studentid	INT(11)

Example query trace

```
SELECT courseid FROM registrations WHERE studentid = 12
SELECT dotw,coursename FROM courses WHERE courseid = 718
```

Existing techniques for synthesizing database

- **Schema-based Synthesizing:**
  - Mainly examine database schema
  - e.g., DOMINO[1], Datafaker[2]

- **Query-based Synthesizing:**
  - Beyond schema analysis, it also examine collected query trace.
  - e.g., EvoSQL[3]

[1] A. Alsharif, G. M. Kapfhammer, and P. McMinn, “Domino: Fast and effective test data generation for relational database schemas,” ICST18
[2] github.com/gangly/datafaker
[3] J. Castelein, M. Aniche, M. Soltani, A. Panichella, and A. van Deursen, “Search-based test data generation for sql queries,” ICSE18

Existing techniques for synthesizing database

Example application code(Schoolmate)

```
1   $q1 = query("SELECT courseid FROM registrations
                 WHERE studentid = '$_POST['student']'");
2   while($reg = fetch_array($q1)){
3       $q2 = query("SELECT dotw,coursename FROM courses
                     WHERE courseid = '$reg[0]'");
4       while($course = fetch_array($q2)){
5           switch($course[0]){
6           case 'M':
…                   ...
7                   print("Monday:".$course[1])
…                   ...
8           case ...
            }
        }
    }
```

Existing techniques for synthesizing database

Example synthesized database(Schema-based)

Table: registration

regid	courseid	studentid
1	8(random)	93(random)

Table: course

courseid	coursename	dotw
1	Dw$c(random)	pvcr(random)

Example application code(Schoolmate)

```
1   $q1 = query("SELECT courseid FROM registrations
                 WHERE studentid = '$_POST['student']'");
2   while($reg = fetch_array($q1)){
3       $q2 = query("SELECT dotw,coursename FROM courses
                     WHERE courseid = '$reg[0]'");
4       while($course = fetch_array($q2)){
5           switch($course[0]){
6           case 'M':
…                   ...
7                   print("Monday:".$course[1])
…                   ...
8           case ...
            }
        }
    }
```

Existing techniques for synthesizing database

Example synthesized database(Schema-based)

Table: registration

regid	courseid	studentid
1	8(random)	93(random)

Table: course

courseid	coursename	dotw
1	Dw$c(random)	pvcr(random)

Example application code(Schoolmate)

Input: POST['student'] = 12

```
1   $q1 = query("SELECT courseid FROM registrations
                 WHERE studentid = '$_POST['student']'");
2   while($reg = fetch_array($q1)){
3       $q2 = query("SELECT dotw,coursename FROM courses
                     WHERE courseid = '$reg[0]'");
4       while($course = fetch_array($q2)){
5           switch($course[0]){
6           case 'M':
…                   ...
7                   print("Monday:".$course[1])
…                   ...
8           case ...
            }
        }
    }
```

Existing techniques for synthesizing database

Example synthesized database(Schema-based)

Table: registration

regid	courseid	studentid
1	8(random)	93(random)

Table: course

courseid	coursename	dotw
1	Dw$c(random)	pvcr(random)

Not Satisfied

Example application code(Schoolmate)

Input: POST['student'] = 12

```
1  $q1 = query("SELECT courseid FROM registrations
                WHERE studentid = '$_POST['student']'");
2  while($reg = fetch_array($q1)){
3      $q2 = query("SELECT dotw,coursename FROM courses
                    WHERE courseid = '$reg[0]'");
4      while($course = fetch_array($q2)){
5          switch($course[0]){
6          case 'M':
…                  ...
7                  print("Monday:".$course[1])
…                  ...
8          case ...
           }
       }
}
```

Existing techniques for synthesizing database

Example synthesized database(Schema-based)

Table: registration

regid	courseid	studentid
1	8(random)	93(random)

Not Satisfied

Table: course

courseid	coursename	dotw
1	Dw$c(random)	pvcr(random)

Example application code(Schoolmate)

Input: POST['student'] = 12

```
1  $q1 = query("SELECT courseid FROM registrations
                WHERE studentid = '$_POST['student']'");
2  while($reg = fetch_array($q1)){
3      $q2 = query("SELECT dotw,coursename FROM courses
                    WHERE courseid = '$reg[0]'");
4      while($course = fetch_array($q2)){
5          switch($course[0]){
6          case 'M':
…                  ...
7                  print("Monday:".$course[1])
…                  ...
8          case ...
           }
       }
}
```

Existing techniques for synthesizing database

Example synthesized database(Schema-based)

Table: registration

regid	courseid	studentid
1	8(random)	93(random)

Table: course

courseid	coursename	dotw
1	Dw$c(random)	pvcr(random)

Miss XSS Vulnerability

Example application code(Schoolmate)

Input: POST['student'] = 12

```
1  $q1 = query("SELECT courseid FROM registrations
                WHERE studentid = '$_POST['student']'");
2  while($reg = fetch_array($q1)){
3      $q2 = query("SELECT dotw,coursename FROM courses
                    WHERE courseid = '$reg[0]'");
4      while($course = fetch_array($q2)){
5          switch($course[0]){
6          case 'M':
…                  ...
7                  print("Monday:".$course[1])
…                  ...
8          case ...
           }
       }
   }
```

Existing techniques for synthesizing database

Example synthesized database(Query-based)

Query Trace

```
SELECT courseid FROM registrations WHERE studentid = 12
SELECT dotw,coursename FROM courses WHERE courseid = 718
```

Example application code(Schoolmate)

```
1   $q1 = query("SELECT courseid FROM registrations
                 WHERE studentid = '$_POST['student']'");
2   while($reg = fetch_array($q1)){
3       $q2 = query("SELECT dotw,coursename FROM courses
                     WHERE courseid = '$reg[0]'");
4       while($course = fetch_array($q2)){
5           switch($course[0]){
6           case 'M':
…                   ...
7                   print("Monday:".$course[1])
…                   ...
8           case ...
            }
        }
    }
```

Existing techniques for synthesizing database

Example synthesized database(Query-based)

Query Trace

```
SELECT courseid FROM registrations WHERE studentid = 12
SELECT dotw,coursename FROM courses WHERE courseid = 718
```

Table: registration

regid	courseid	studentid
1	718	12

Example application code(Schoolmate)

```
1   $q1 = query("SELECT courseid FROM registrations
                 WHERE studentid = '$_POST['student']'");
2   while($reg = fetch_array($q1)){
        $q2 = query("SELECT dotw,coursename FROM courses
                     WHERE courseid = '$reg[0]'");
4       while($course = fetch_array($q2)){
5           switch($course[0]){
6           case 'M':
…                   ...
7                   print("Monday:".$course[1])
…                   ...
8           case ...
            }
        }
    }
```

Existing techniques for synthesizing database

Example synthesized database(Query-based)

Query Trace

```
SELECT courseid FROM registrations WHERE studentid = 12
SELECT dotw,coursename FROM courses WHERE courseid = 718
```

Table: registration

regid	courseid	studentid
1	718	12

Example application code(Schoolmate)

```
1   $q1 = query("SELECT courseid FROM registrations
                 WHERE studentid = '$_POST['student']'");
2   while($reg = fetch_array($q1)){
        $q2 = query("SELECT dotw,coursename FROM courses
                     WHERE courseid = '$reg[0]'");
4       while($course = fetch_array($q2)){
5           switch($course[0]){
6           case 'M':
…                   ...
7                   print("Monday:".$course[1])
…                   ...
8           case ...
            }
        }
    }
```

Existing techniques for synthesizing database

Example synthesized database(Query-based)

Query Trace

```
SELECT courseid FROM registrations WHERE studentid = 12
SELECT dotw,coursename FROM courses WHERE courseid = 718
```

Table: registration

regid	courseid	studentid
1	718	12

Table: course

courseid	coursename	dotw
718	8!x=(random)	ab@5(random)

Example application code(Schoolmate)

```
1  $q1 = query("SELECT courseid FROM registrations
                WHERE studentid = '$_POST['student']'");
2  while($reg = fetch_array($q1)){
3      $q2 = query("SELECT dotw,coursename FROM courses
                    WHERE courseid = '$reg[0]'");
4      while($course = fetch_array($q2)){
5          switch($course[0]){
6          case 'M':
…                  ...
7                  print("Monday:".$course[1])
…                  ...
8          case ...
           }
       }
   }
```

Existing techniques for synthesizing database

Example synthesized database(Query-based)

Query Trace

```
SELECT courseid FROM registrations WHERE studentid = 12
SELECT dotw,coursename FROM courses WHERE courseid = 718
```

Table: registration

regid	courseid	studentid
1	718	12

Table: course

courseid	coursename	dotw
718	8!x=(random)	ab@5(random)

Example application code(Schoolmate)

Input: POST['student'] = 12

```
1  $q1 = query("SELECT courseid FROM registrations
                WHERE studentid = '$_POST['student']'");
2  while($reg = fetch_array($q1)){
       $q2 = query("SELECT dotw,coursename FROM courses
                    WHERE courseid = '$reg[0]'");
4      while($course = fetch_array($q2)){
5          switch($course[0]){
6          case 'M':
…                  ...
7                  print("Monday:".$course[1])
…                  ...
8          case ...
           }
       }
   }
```

Existing techniques for synthesizing database

Example synthesized database(Query-based)

Query Trace

```
SELECT courseid FROM registrations WHERE studentid = 12
SELECT dotw,coursename FROM courses WHERE courseid = 718
```

Table: registration

regid	courseid	studentid
1	718	12

Table: course

courseid	coursename	dotw
718	8!x=(random)	ab@5(random)

Example application code(Schoolmate)

Input: POST['student'] = 12

```
1  $q1 = query("SELECT courseid FROM registrations
                WHERE studentid = '$_POST['student']'");
2  while($reg = fetch_array($q1)){
3      $q2 = query("SELECT dotw,coursename FROM courses
                    WHERE courseid = '$reg[0]'");
4      while($course = fetch_array($q2)){
5          switch($course[0]){
6          case 'M':
…                  ...
7                  print("Monday:".$course[1])
…                  ...
8          case ...
           }
       }
}
```

Existing techniques for synthesizing database

Example synthesized database(Query-based)

Query Trace

```
SELECT courseid FROM registrations WHERE studentid = 12
SELECT dotw,coursename FROM courses WHERE courseid = 718
```

Satisfied

Satisfied

Table: registration

regid	courseid	studentid
1	718	12

Table: course

courseid	coursename	dotw
718	8!x=(random)	ab@5(random)

Example application code(Schoolmate)

Input: POST['student'] = 12

```
1  $q1 = query("SELECT courseid FROM registrations
                WHERE studentid = '$_POST['student']'");
2  while($reg = fetch_array($q1)){
3      $q2 = query("SELECT dotw,coursename FROM courses
                    WHERE courseid = '$reg[0]'");
4      while($course = fetch_array($q2)){
5          switch($course[0]){
6          case 'M':
…                  ...
7                  print("Monday:".$course[1])
…                  ...
8          case ...
           }
       }
   }
```

Existing techniques for synthesizing database

Example synthesized database(Query-based)

Query Trace

```
SELECT courseid FROM registrations WHERE studentid = 12
SELECT dotw,coursename FROM courses WHERE courseid = 718
```

Table: registration

regid	courseid	studentid
1	718	12

Table: course

courseid	coursename	dotw
718	8!x=(random)	ab@5(random)

Not satisfied

Example application code(Schoolmate)

Input: POST['student'] = 12

```
1  $q1 = query("SELECT courseid FROM registrations
                WHERE studentid = '$_POST['student']'");
2  while($reg = fetch_array($q1)){
3      $q2 = query("SELECT dotw,coursename FROM courses
                    WHERE courseid = '$reg[0]'");
4      while($course = fetch_array($q2)){
5          switch($course[0]){
6          case 'M':
…                  ...
7                  print("Monday:".$course[1])
…                  ...
8          case ...
           }
       }
   }
```

Existing techniques for synthesizing database

Example synthesized database(Query-based)

Query Trace

```
SELECT courseid FROM registrations WHERE studentid = 12
SELECT dotw,coursename FROM courses WHERE courseid = 718
```

Table: registration

regid	courseid	studentid
1	718	12

Table: course

courseid	coursename	dotw
718	8!x=(random)	ab@5(random)

Not satisfied

Example application code(Schoolmate)

Input: POST['student'] = 12

```
1  $q1 = query("SELECT courseid FROM registrations
                WHERE studentid = '$_POST['student']'");
2  while($reg = fetch_array($q1)){
3      $q2 = query("SELECT dotw,coursename FROM courses
                    WHERE courseid = '$reg[0]'");
4      while($course = fetch_array($q2)){
5          switch($course[0]){
6          case 'M':
…                  ...
7                  print("Monday:".$course[1])
…                  ...
8          case ...
           }
       }
   }
```

Existing techniques for synthesizing database

Example synthesized database(Query-based)

Query Trace

```
SELECT courseid FROM registrations WHERE studentid = 12
SELECT dotw,coursename FROM courses WHERE courseid = 718
```

Table: registration

regid	courseid	studentid
1	718	12

Miss XXS Vulnerability

Table: course

courseid	coursename	dotw
718	8!x=(random)	ab@5(random)

Example application code(Schoolmate)

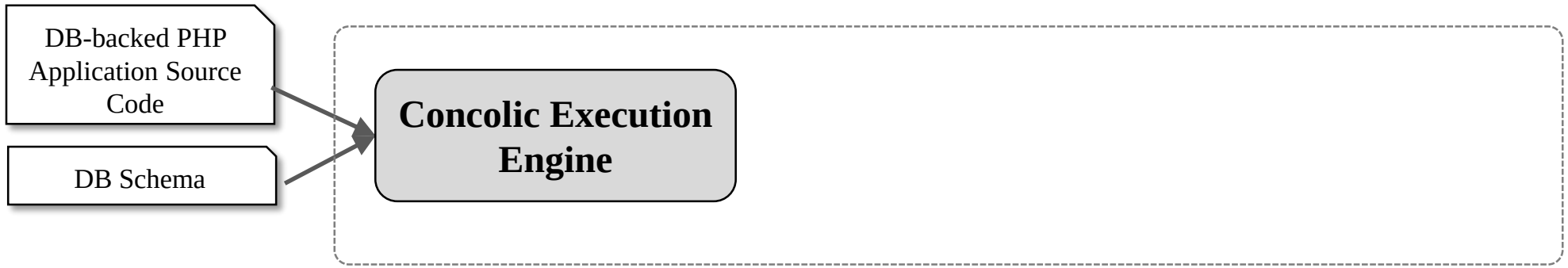
Input: POST['student'] = 12

```
1  $q1 = query("SELECT courseid FROM registrations
                WHERE studentid = '$_POST['student']'");
2  while($reg = fetch_array($q1)){
3      $q2 = query("SELECT dotw,coursename FROM courses
                    WHERE courseid = '$reg[0]'");
4      while($course = fetch_array($q2)){
5          switch($course[0]){
6          case 'M':
…                  ...
7                  print("Monday:".$course[1])
…                  ...
8          case ...
           }
       }
   }
```

System Overview

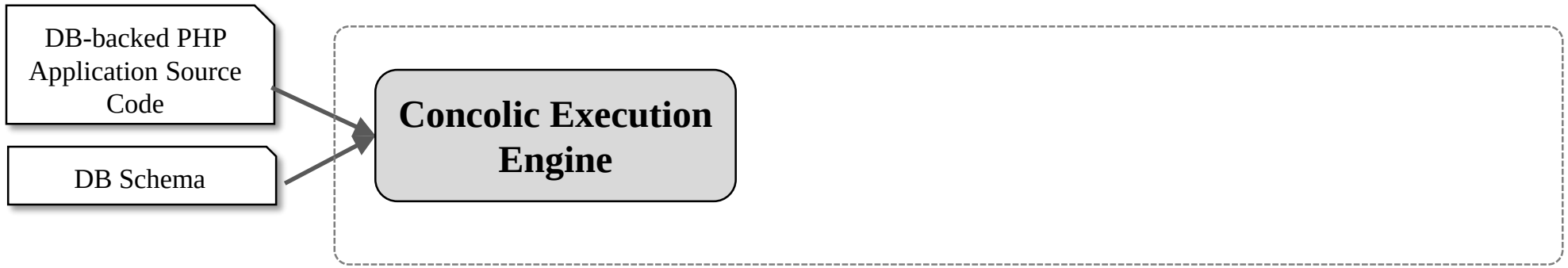
- Identify dependencies between **codebase, queries, and schema**
- Formulating database constraints
- Database instance generation based on constraints

System Overview

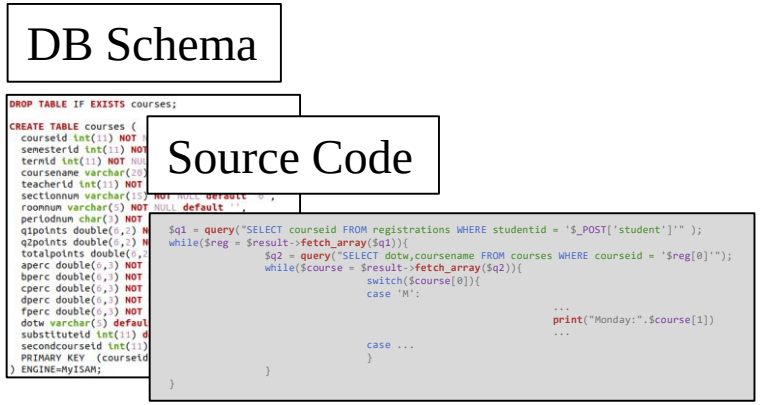


SYNTHDB: Synthesizing Database via Program Analysis

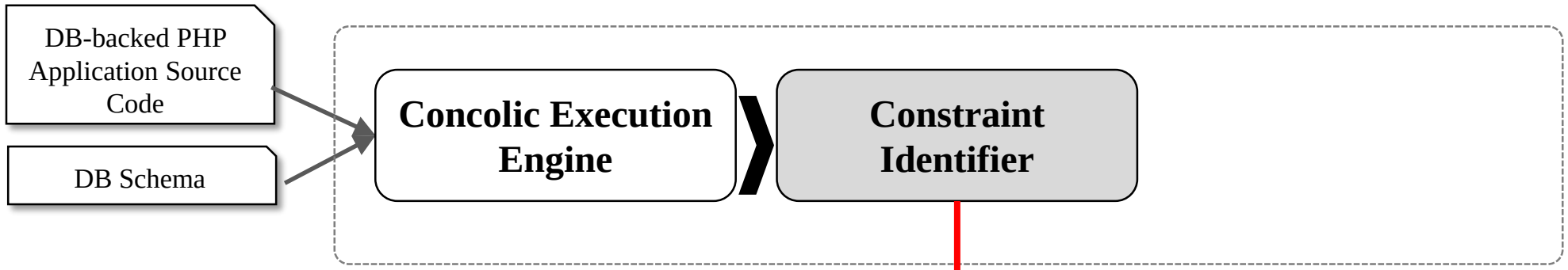
System Overview



SYNTHDB: Synthesizing Database via Program Analysis



System Overview



SYNTHDB: Synthesizing Database via Program Analysis

DB Schema

Source Code

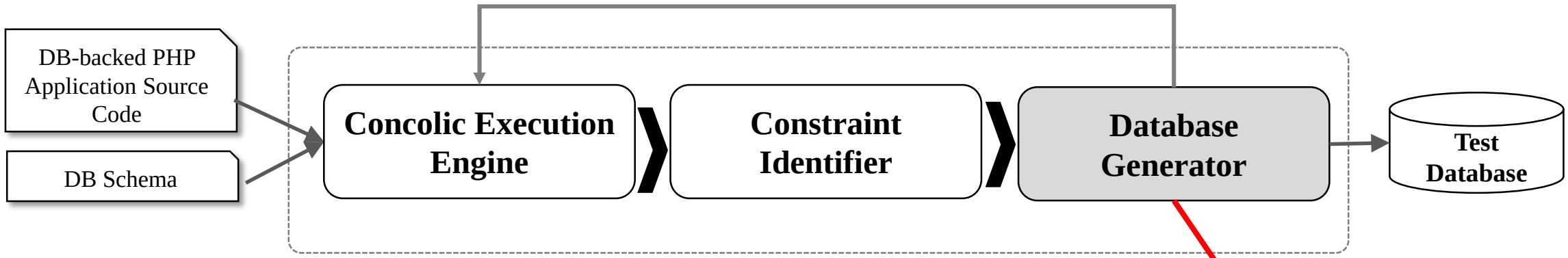
```
DROP TABLE IF EXISTS courses;
CREATE TABLE courses (
  courseid int(11) NOT
  semesterid int(11) NOT
  termid int(11) NOT NUL
  coursename varchar(20
  teacherid int(11) NOT
  sectionnum varchar(15) NOT NULL default '0',
  roomnum varchar(5) NOT NULL default '',
  periodnum char(3) NOT
  q1points double(6,2) N
  q2points double(6,2) N
  totalpoints double(6,
  aperc double(6,3) NOT
  bperc double(6,3) NOT
  cperc double(6,3) NOT
  dperc double(6,3) NOT
  fperc double(6,3) NOT
  dotw varchar(5) defaul
  substituteid int(11) d
  secondcourseid int(11)
  PRIMARY KEY  (courseid
) ENGINE=MyISAM;
```

```
$q1 = query("SELECT courseid FROM registrations WHERE studentid = '$_POST['student']'" );
while($reg = $result->fetch_array($q1)){
        $q2 = query("SELECT dotw,coursename FROM courses WHERE courseid = '$reg[0]'");
        while($course = $result->fetch_array($q2)){
                switch($course[0]){
                case 'M':
                                                ...
                                                print("Monday:".$course[1])
                                                ...
                case ...
                }
        }
}
```

Constraints

```
...
type(T.A)=int AND
var(T.A)==12 AND
type(T.B)=int AND
var(T.B)==718 AND
type(T.C)=str AND
var(T.C)=="M"
```

System Overview



SYNTHDB: Synthesizing Database via Program Analysis

DB Schema

Source Code

```
DROP TABLE IF EXISTS courses;
CREATE TABLE courses (
  courseid int(11) NOT
  semesterid int(11) NOT
  termid int(11) NOT NUL
  coursename varchar(20
  teacherid int(11) NOT
  sectionnum varchar(15) NOT NULL default '0',
  roomnum varchar(5) NOT NULL default '',
  periodnum char(1) NOT
  q1points double(6,2) N
  q2points double(6,2) N
  totalpoints double(6,2
  aperc double(6,3) NOT
  bperc double(6,3) NOT
  cperc double(6,3) NOT
  dperc double(6,3) NOT
  fperc double(6,3) NOT
  dotw varchar(5) defaul
  substituteid int(11) d
  secondcourseid int(11)
  PRIMARY KEY  (courseid
) ENGINE=MyISAM;
```

```
$q1 = query("SELECT courseid FROM registrations WHERE studentid = '$_POST['student']'" );
while($reg = $result->fetch_array($q1)){
        $q2 = query("SELECT dotw,coursename FROM courses WHERE courseid = '$reg[0]'");
        while($course = $result->fetch_array($q2)){
                switch($course[0]){
                case 'M':
                                                ...
                                                print("Monday:".$course[1])
                                                ...
                case ...
                }
        }
}
```

Constraints

```
...
type(T.A)=int AND
var(T.A)==12 AND
type(T.B)=int AND
var(T.B)==718 AND
type(T.C)=str AND
var(T.C)=="M"
```

DB Instance

Table: T

A	B	C
12	718	M

Database Constraint

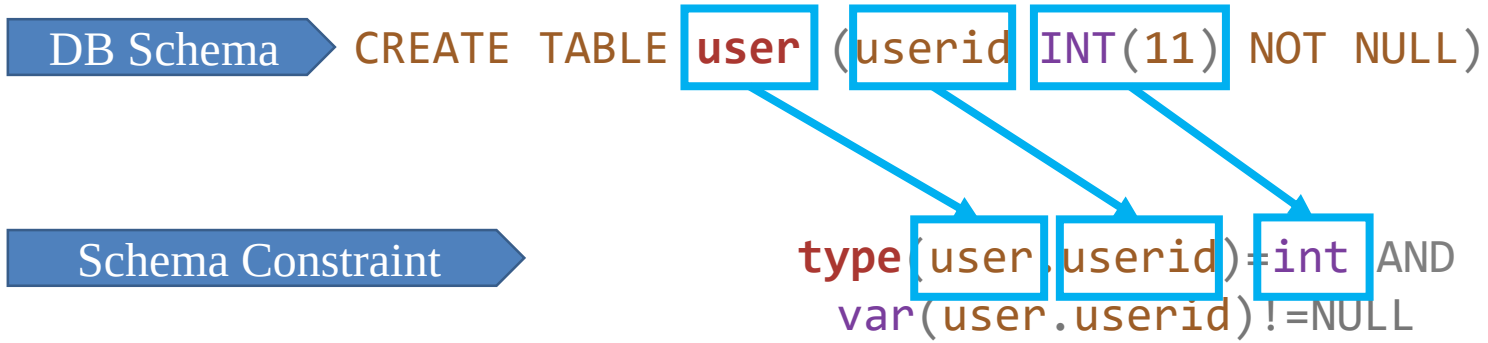
- **Schema Constraint**
  - Type definitions
  - Value specifications
  - Structural keywords
  - Ensure DB integrity

DB Schema

```
CREATE TABLE user (userid INT(11) NOT NULL)
```

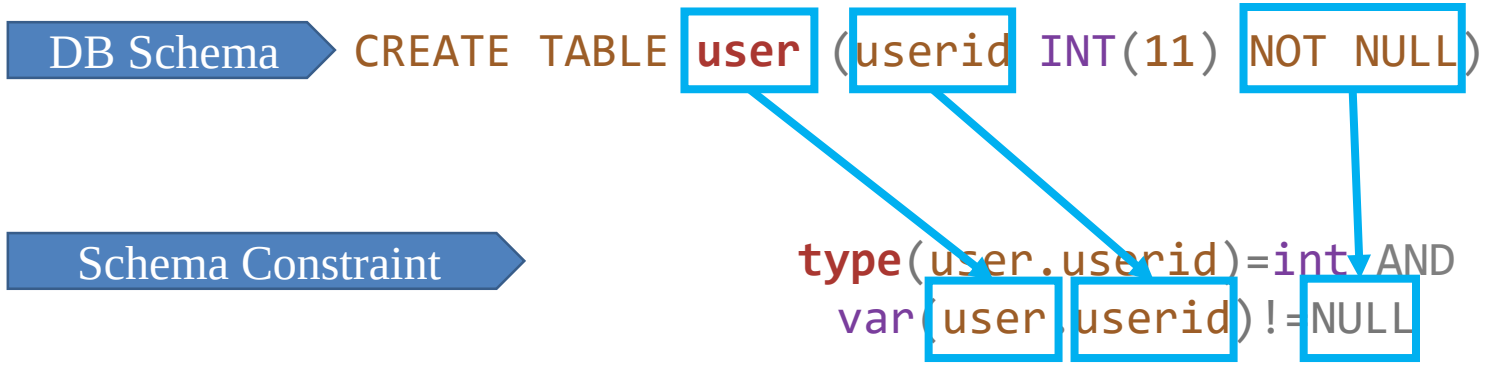
Database Constraint

- **Schema Constraint**
  - Type definitions
  - Value specifications
  - Structural keywords
  - Ensure DB integrity



Database Constraint

- **Schema Constraint**
  - Type definitions
  - Value specifications
  - Structural keywords
  - Ensure DB integrity



Database Constraint

- **Query-condition Constraint**
  - JOIN operation
  - WHERE clause

PHP Code

```
query("SELECT id, name FROM users JOIN office ON users.id=office.userid
WHERE name = 'john'");
```

Database Constraint

- **Query-condition Constraint**
  - JOIN operation
  - WHERE clause

PHP Code

```
query("SELECT id, name FROM users JOIN office ON users.id=office.userid
WHERE name = 'john'");
```

Query-condition Constraint

```
var(users.id)=var(office.userid) AND
var(users.name)=="john"
```

Database Constraint

- **Query-condition Constraint**
  - JOIN operation
  - WHERE clause

PHP Code

```
query("SELECT id, name FROM users JOIN office ON users.id=office.userid
WHERE name = 'john'");
```

Query-condition Constraint

```
var(users.id)=var(office.userid) AND
var(users.name)=="john"
```

Database Constraint

- **Pre-query constraint**
  - Computations and predicate conditions before issuing query

PHP Code

```
if (strlen($u)>=6) {
        query("INSERT INTO authors (user, active) VALUES ('$u', 1)");
}
```

Database Constraint

- **Pre-query constraint**
  - Computations and predicate conditions before issuing query

PHP Code

```
if (strlen($u)>=6) {
    query("INSERT INTO authors (user, active) VALUES ('$u', 1)");
}
```

Database Constraint

- **Pre-query constraint**
  - Computations and predicate conditions before issuing query

PHP Code

```
if (strlen($u)>=6) {
    query("INSERT INTO authors (user, active) VALUES ('$u', 1)");
}
```

Pre-Query Constraint

```
var(authors.user).length>=6 AND
var(authors.active)==1
```

Database Constraint

- **Pre-query constraint**
  - Computations and predicate conditions before issuing query

PHP Code

```
if (strlen($u)>=6) {
    query("INSERT INTO authors (user, active) VALUES ('$u', 1)");
}
```

Pre-Query Constraint

```
var(authors.user).length>=6 AND
var(authors.active)==1
```

Database Constraint

- **Post-query constraint**
  - Program code dependent on the returned data from database

PHP Code

```php
$q1 = query ("SELECT job, points FROM users");
while($users = fetch_array($q1)) {
    if ($users['job'] == "teacher") {
        ...
    }
}
```

Database Constraint

- **Post-query constraint**
  - Program code dependent on the returned data from database

PHP Code

```
$q1 = query ("SELECT job, points FROM users");
while($users = fetch_array($q1)) {
    if ($users['job'] == "teacher") {
        ...
    }
}
```

Database Constraint

- **Post-query constraint**
  - Program code dependent on the returned data from database

PHP Code

```php
$q1 = query ("SELECT job, points FROM users");
while($users = fetch_array($q1)) {
    if ($users['job'] == "teacher") {
        ...
    }
}
```

Database Constraint

- **Post-query constraint**
  - Program code dependent on the returned data from database

PHP Code

```
$q1 = query ("SELECT job, points FROM users');
while($users = fetch_array($q1)) {
    if ($users['job'] == "teacher") {
        ...
    }
}
```

Post-Query Constraint

```
var(users.job)=="teacher"
```

Database Constraint

Schema Constraint | Query-condition Constraint | Pre-Query Constraint | Post-Query Constraint | **Syn-Query Constraint**

- **Synchronized-query Constraint**
  - Analyzing a set of SQL queries in the same block

PHP Code

```php
if($_POST['claim']==true) {
  $res1 = query("INSERT INTO point (userid,...) VALUES ($id,...)");

  $res2 = query("INSERT INTO claimed (userid,...) VALUES ($id,...)");
}
```

Database Constraint

Schema Constraint | Query-condition Constraint | Pre-Query Constraint | Post-Query Constraint | **Syn-Query Constraint**

- **Synchronized-query Constraint**
  - Analyzing a set of SQL queries in the same block

PHP Code

```
if($_POST['claim']==true) {
  $res1 = query("INSERT INTO point (userid,...) VALUES ($id,...)");

  $res2 = query("INSERT INTO claimed (userid,...) VALUES ($id,...)");
}
```

Database Constraint

Schema Constraint | Query-condition Constraint | Pre-Query Constraint | Post-Query Constraint | **Syn-Query Constraint**

- **Synchronized-query Constraint**
  - Analyzing a set of SQL queries in the same block

PHP Code

```
if($_POST['claim']==true) {
  $res1 = query("INSERT INTO point (userid,...) VALUES ($id,...)");

  $res2 = query("INSERT INTO claimed (userid,...) VALUES ($id,...)");
}
```

Database Constraint

Schema Constraint | Query-condition Constraint | Pre-Query Constraint | Post-Query Constraint | **Syn-Query Constraint**

- **Synchronized-query Constraint**
  - Analyzing a set of SQL queries in the same block

PHP Code

```
if($_POST['claim']==true) {
  $res1 = query("INSERT INTO point (userid,...) VALUES ($id,...)");

  $res2 = query("INSERT INTO claimed (userid,...) VALUES ($id,...)");
}
```

Synchronized-Query Constraint

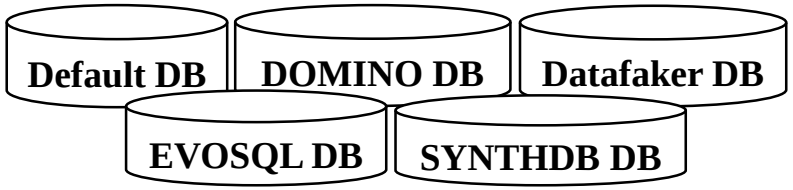
```
var(point.userid) == var(claimed.userid)
```

Evaluation

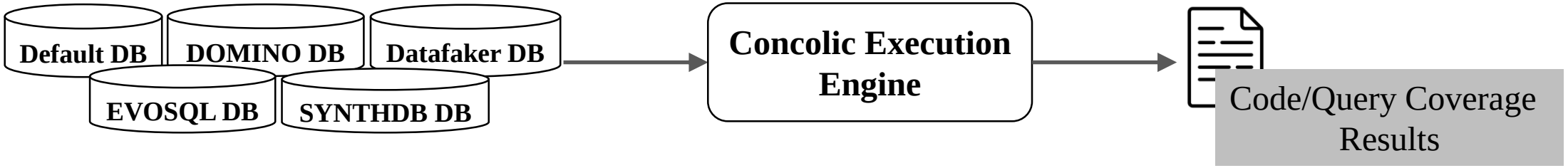
- 17 real-world PHP web applications from different categories

Name	Files	LLOC	# of Query	Category
SchoolMate (1.5.4)	63	1,587	263	School Management
Webchess	29	1,505	94	Web Game
Timeclock (1.04)	63	10,820	299	Employee Management
Mybloggie (2.1.4)	59	3,053	84	Content Management
Faqforge (1.3.2)	15	302	30	Online Forum
Wackopicko	49	720	40	Photo Management
phpBB (2.0.23/3.3.8)	74/1,091	10,798/40,612	377/1,343	Online Forum
OpenCart (3.0.3.8/4.0.0)	1,932/2,866	60,515/49,018	943/999	Ecommerce
WordPress (5.1.2/6.0.1)	901/1,332	84,891/110,227	315/307	Content Management
Simple Machines Forum (2.1.2)	316	45,641	1,206	Online Forum
OsCommerce (2.4.0)	422	15,809	916	Ecommerce
CEPhoenix (1.0.7)	1,361	23,938	686	Ecommerce
ZenCart (1.5.7)	1,829	74,960	1,920	Ecommerce
Drupal (9.0.0)	8,854	237,001	322	Content Management
In total	21,256	771,406	10,144	

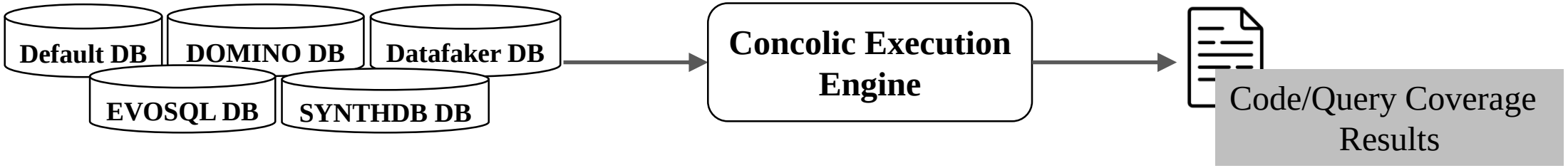
Evaluation: Coverage

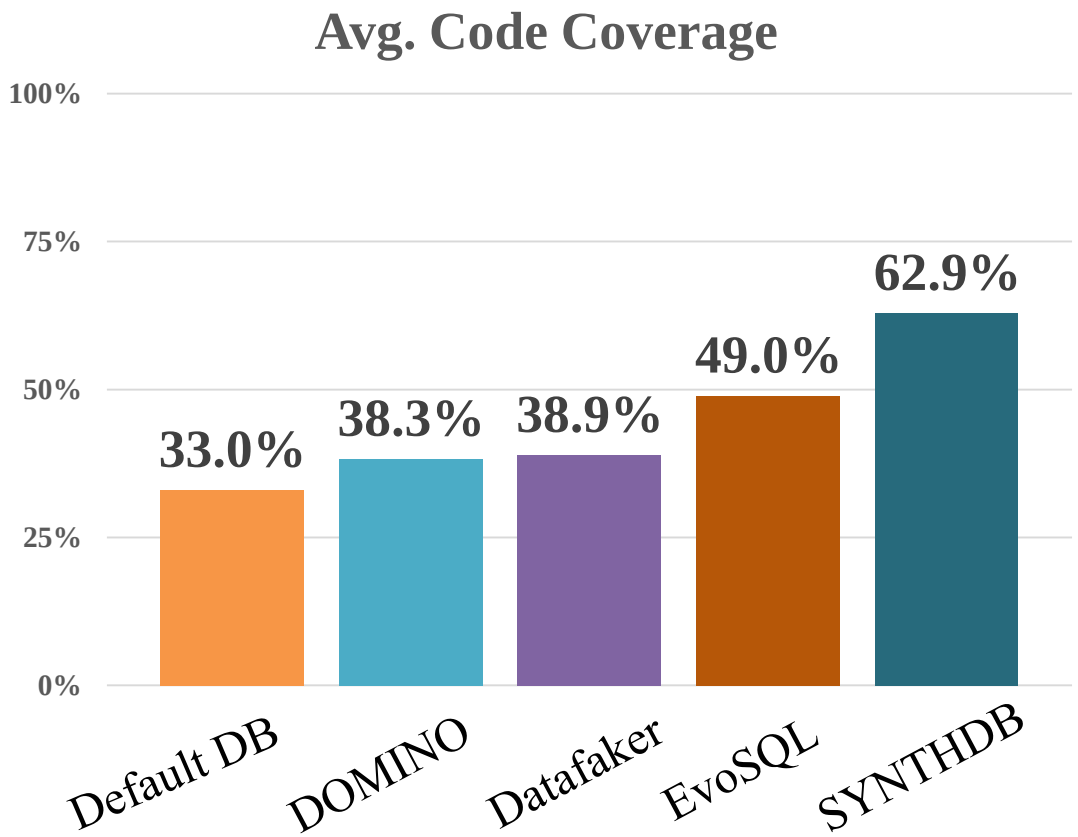


Evaluation: Coverage

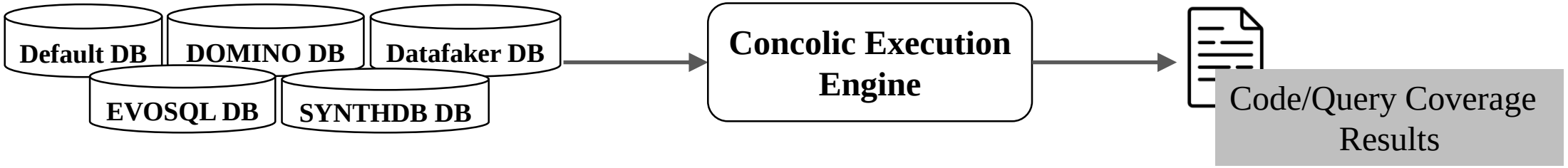


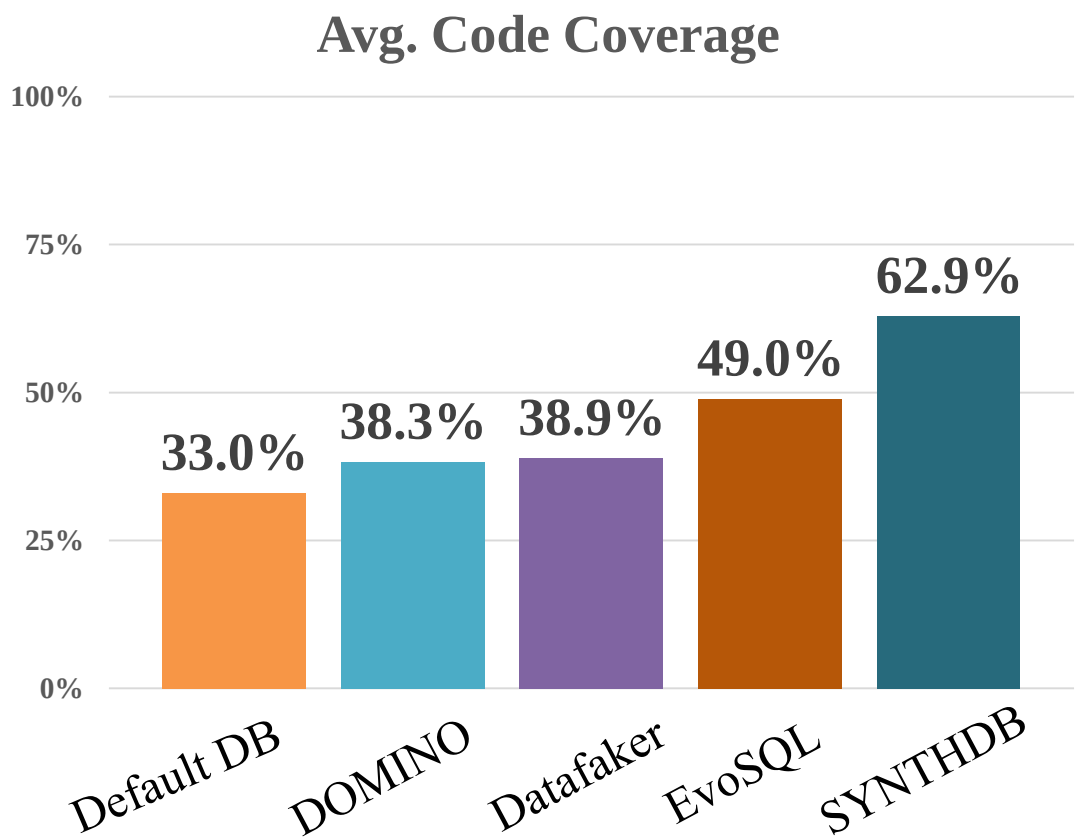
Evaluation: Coverage

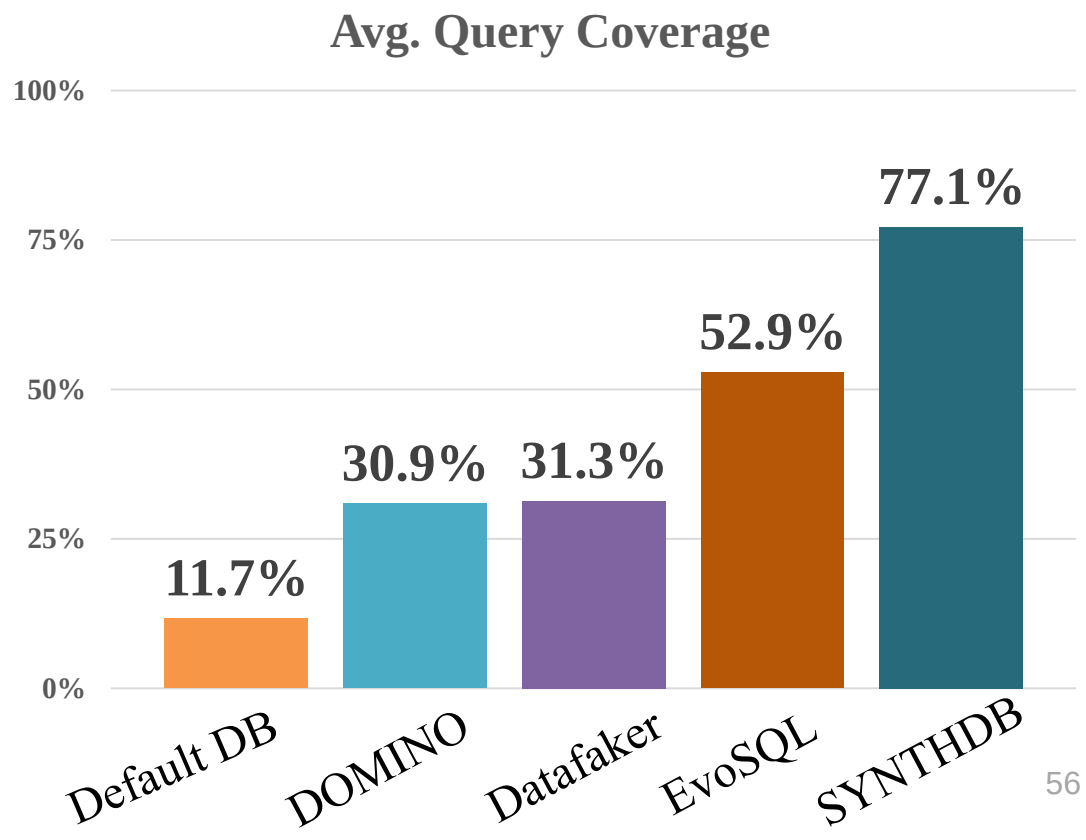




Evaluation: Coverage







Evaluation: Reachability

Known vulnerabilities reachability test

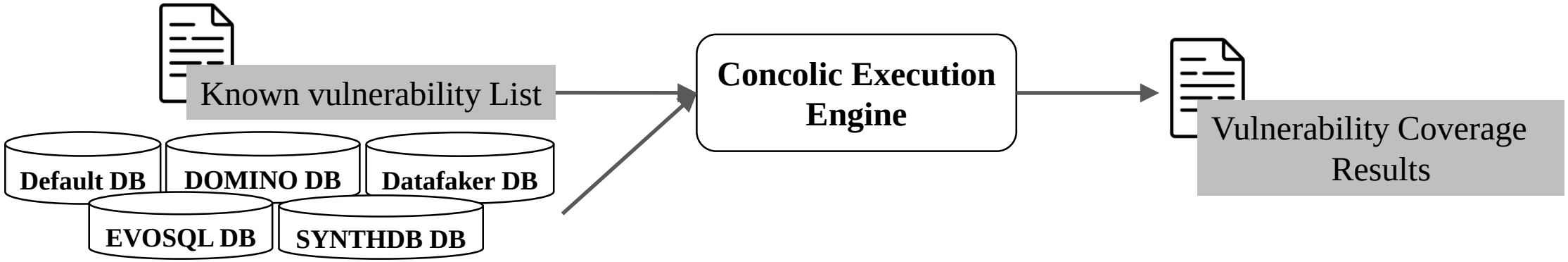
- Collected 189 known vulnerabilities from 11 PHP application

Known vulnerability List

Evaluation: Reachability

Known vulnerabilities reachability test

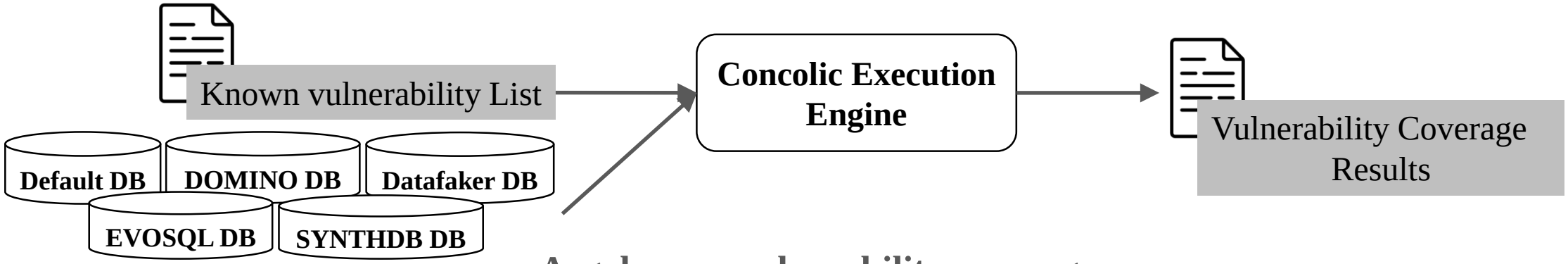
- Collected 189 known vulnerabilities from 11 PHP application

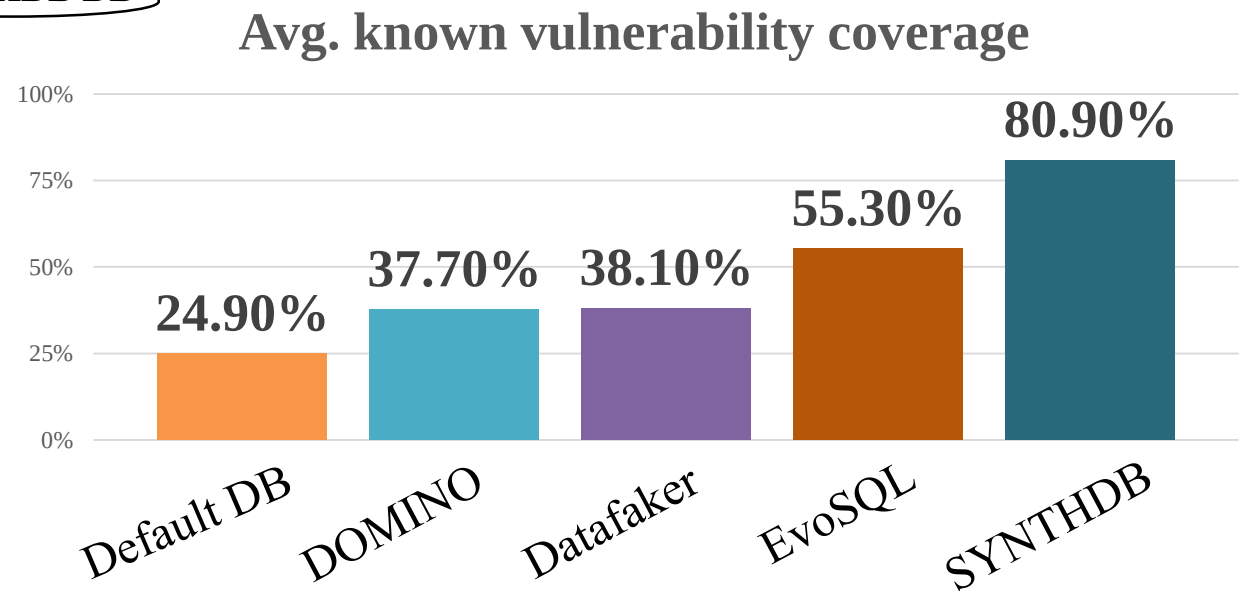


Evaluation: Reachability

Known vulnerabilities reachability test

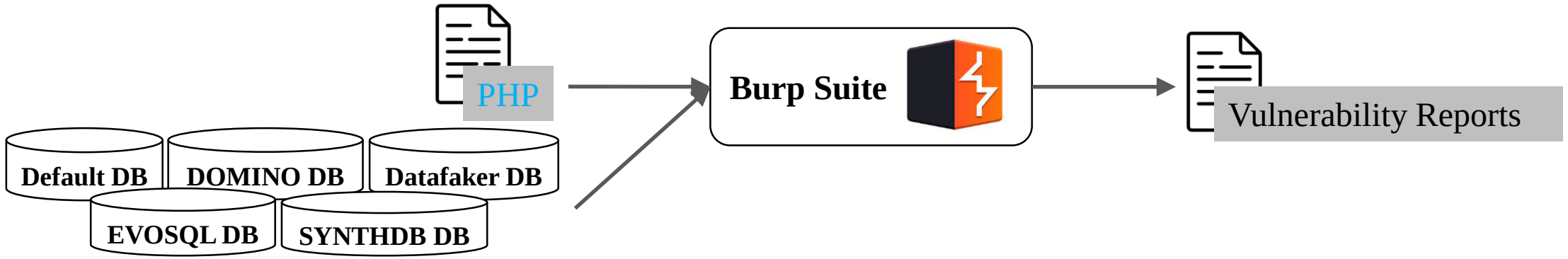
- Collected 189 known vulnerabilities from 11 PHP application





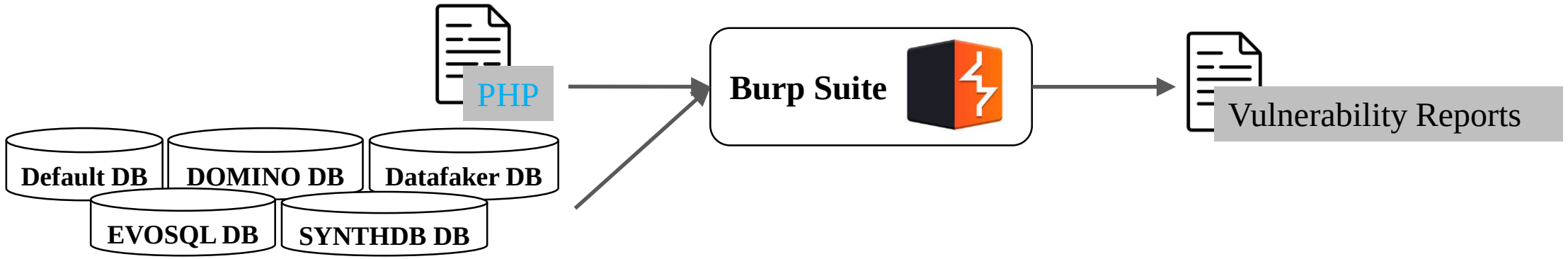
Evaluation: Security

Vulnerabilities scanning with Burp Suite



Evaluation: Security

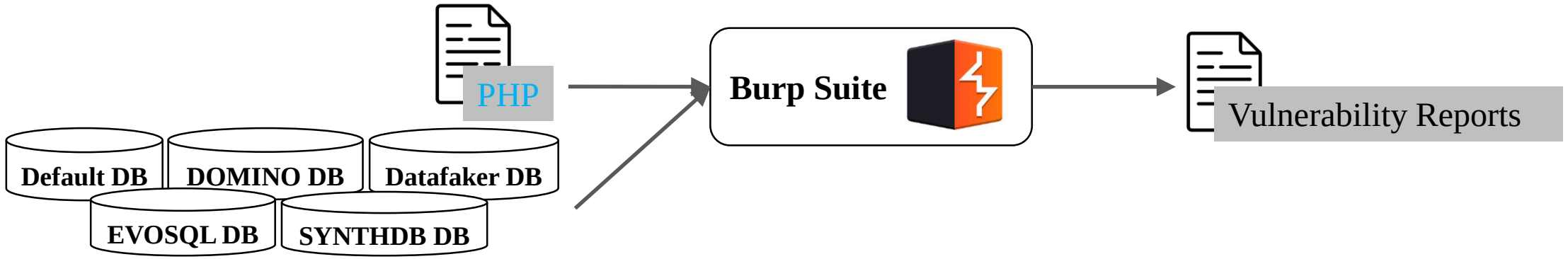
Vulnerabilities scanning with Burp Suite



Name	# Known Vulnerabilities	Default DB	DOMINO	Datafaker	EVOSQL	SYNTHDB
SchoolMate (1.5.4)	80	7	31	31	62	80 (+13 new)
Webchess	18	3	5	5	12	18 (+4 new)
Timeclock (1.04)	8	2	2	2	3	7 (+7 new)
Mybloggie (2.1.4)	13	3	4	4	8	9 (+8 new)
OsCommerce (2.4.0)	0	0	0	0	0	0 (+1 new)

Evaluation: Security

Vulnerabilities scanning with Burp Suite



Name	# New Vulnerabilities	SQLI	XSS	
SchoolMate (1.5.4)	13	0	13	Reported
Webchess	4	2	2	CVE-2023-22959
Timeclock (1.04)	7	2	5	Reported
Mybloggie (2.1.4)	8	7	1	Reported
OsCommerce (2.4.0)	1	1	0	Reported

Takeaways

SYNTHDB:

- Identify dependencies between **codebase, queries, and schema**
- Introduce 5 types of database constraints
- Cover more code and reach more known vulnerabilities
- Aid Burp Suite discovers 33 new vulnerabilities

Thank You

Please contact an.chen25@uga.edu for any questions