RIPS

BSI - 12. Deutscher IT-Sicherheitskongress

10.-12. Mai.2011, Bonn

Automatisierte Schwachstellenerkennung in PHP-Software mittels statischer Quellcode-Analyse

Johannes Dahse, Ruhr-Universität Bochum

RUB

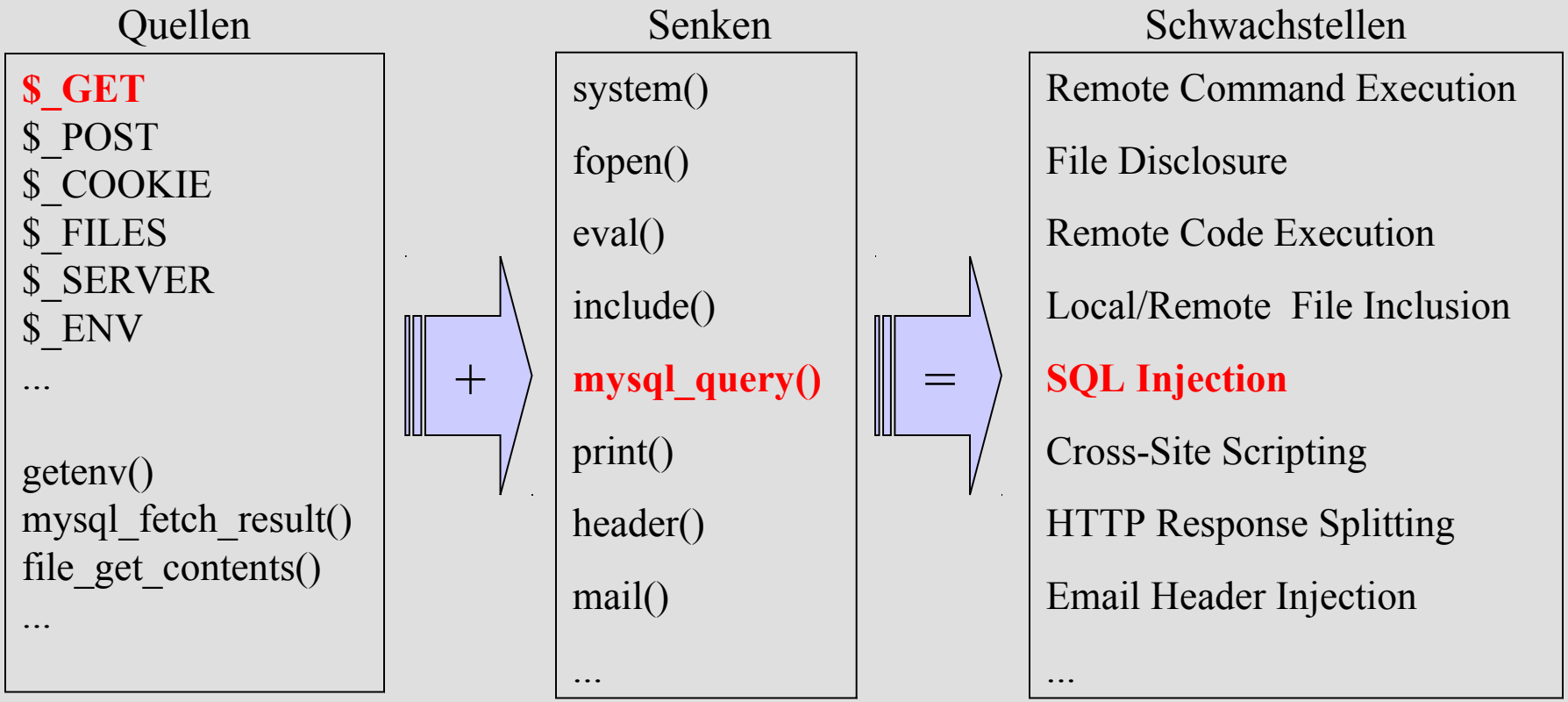
ETIT

hgi

1.1 Motivation

- Schwachstellen 2.0 im Web 2.0
- Defacements, Malware-Verbreitung, Daten
- PHP ist die verbreitetste Skriptsprache im Web
- PHP ist die anfälligste Skriptsprache bzgl. Sicherheit
- Schwachstellen-Suche kann automatisiert werden (Zeit/Geld)
- Entwickler, Heimanwender, Pentester
- Kaum oder unzureichende Open Source Tools
- Eigenes Audit-Interface gewünscht

1.2 Konzept der Injection Flaws



1.3 Konzept der Taint-Analyse

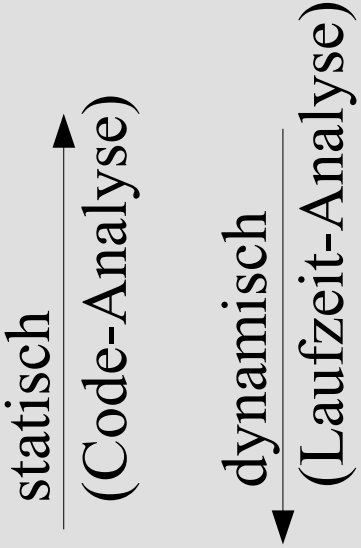
```
<?php

$id = $_GET['id'];

$query = "SELECT data FROM users WHERE id = $id";

mysql_query($query);

?>
```



2. RIPS

- Backend in PHP (~5000Z), Frontend in HTML/JS (~1000Z)
- Benötigt lediglich lokalen Webserver+Browser, < 300kb
- Detektiert Schwachstellen automatisiert mittels statischer Quellcode-Analyse, unterstützt alle Injection Flaw-Typen
- Bietet eigene IDE für manuelle Schwachstellen-Analyse
- Erstellt Exploits
- Erstellt mod_security/Snort rules (*hot patching*)
- Open Source, frei verfügbar

2.1 Analyse-Schritte

- Lexikalische Analyse
- Semantische Analyse
- Modell-Erstellung
- Kontrollfluss-Analyse
- Taint-Analyse
- Intraprozedurale Analyse
- Interprozedurale Analyse

Details: siehe Tagungsband

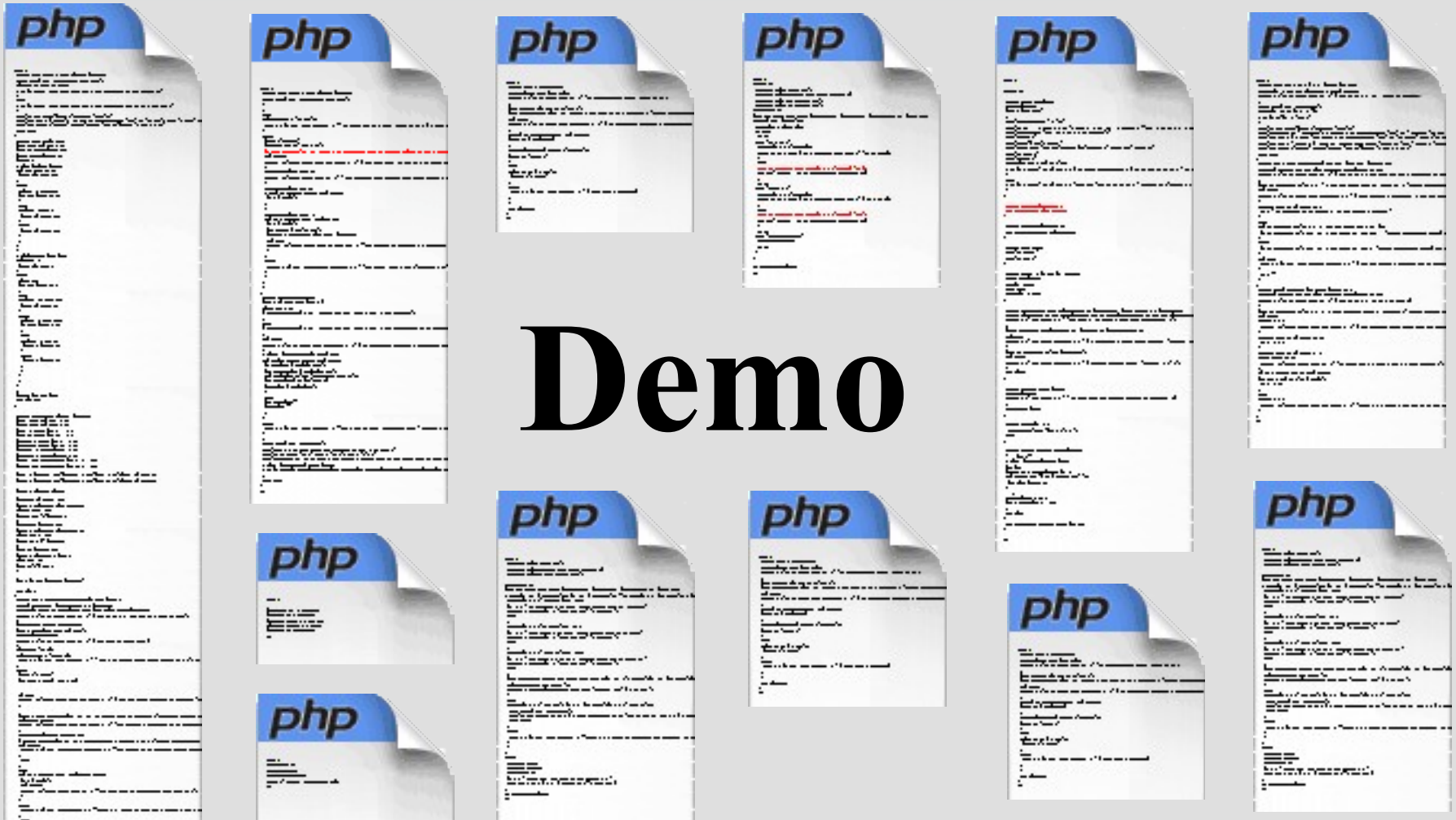
```
<?php
include('config.php');

function secure($str)
{
	return escapeshellarg($str);
}

function crypto($msg, $key)
{
	$msg = secure($msg);
	$enc = exec("./crypt.pl $msg $key");
	return $enc;
}

if(isset($_GET['msg']))
	$encrypted = crypto($_GET['msg'], KEY);
else
	echo 'Please enter a message!';

?>
```



path / file: D:/cipher3/timeclock/work.php subdirs

verbosity level: 1. user tainted only vuln type: - SQL Injection scan

code style: twilight bottom-up regex: search

windows: files | user input | stats | functions

File: D:\cipher3\timeclock\work.php

SQL Injection

26: mysql_query return mysql_query($command); // html_and_database_functions.php

25: function system ($command){

Call triggers vulnerability in *system_()* (click name to jump to declaration)

162: system $result = system ("SELECT id,start FROM timeclock WHERE employee='$u

113: $userid = (int)$_SESSION["Userid"]; // work_functions.php

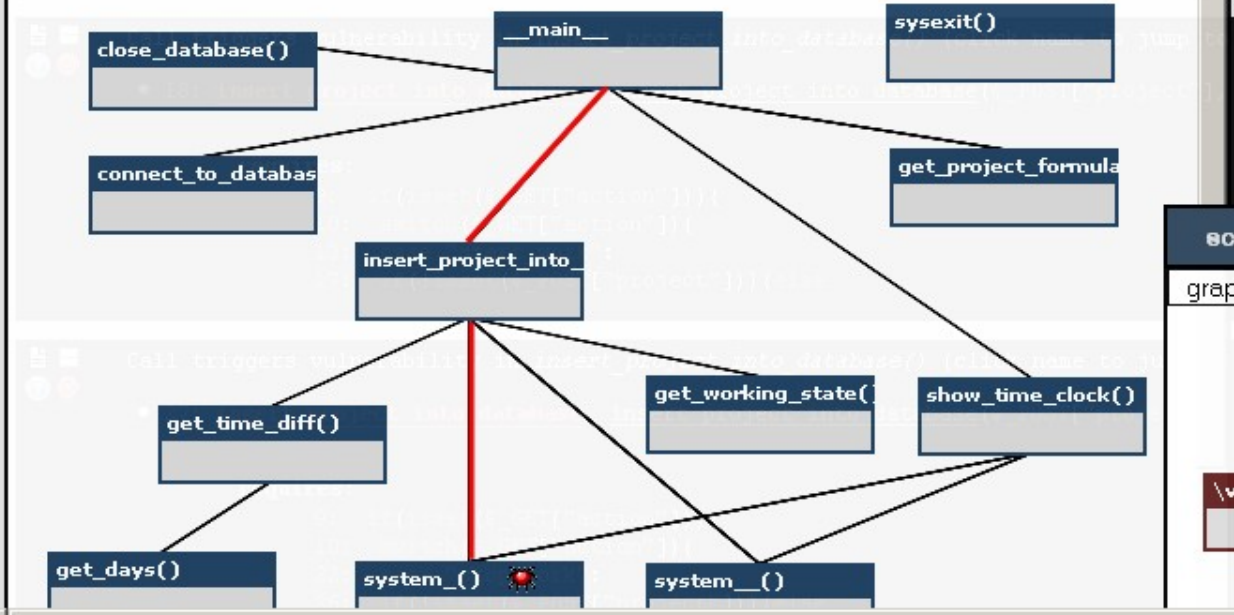
107: function insert_project_into_database($project, $state){

CodeViewer - D:\cipher3\timeclock\work.php

```
13          case "start_work":
14              if(!isset($_POST["project"])){
15                  get_project_formular("./work
16              }
17              else{
18                  insert_project_into_database
19                  header("Location: ./work.php
20              }
21              break;
22          case "stop_work":
23              if(!isset($_POST["project"])){
24                  get_project_formular("./work
25              }
26              else{
27                  insert_project_into_database
28                  header("Location: ./work.php
29              }
30              break;
31          case "show_time_clock":
32              show_time_clock();
```

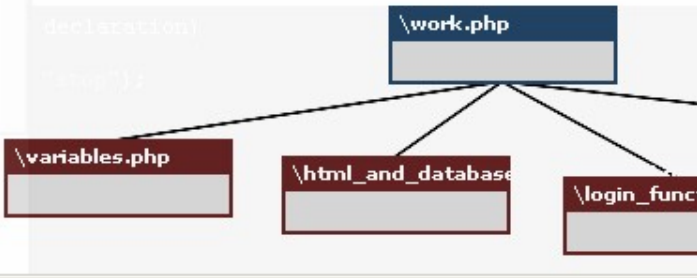
user defined functions and calls

graph | list



scanned files and includes

graph | list



3. Zusammenfassung

+ neuer Ansatz eines Quellcode-Analyse-Tools
+ Finden, Analysieren, Beseitigen und Verifizieren von Lücken
+ schnell und effizient, frei verfügbar

- die Modell-Erstellung muss optimiert werden (CFGs)
- hat einige Limitationen im OOP-Bereich
- false positives / false negatives

3. Zusammenfassung

\+ RIPS ist ein hilfreiches Tool zur PHP-Code-Analyse

\- RIPS ist (noch ;) kein ultimativer Schwachstellen-Detektor

RIPS ist Open Source (GPL) und kostenlos verfügbar:
http://sourceforge.net/projects/rips-scanner/ Beta

Download! Scan!

Feedback ist herzlich willkommen.

Fragen ?

Vielen Dank...

... für die Einladung & Aufmerksamkeit

... an Dominik Birk (RUB) für die Betreuung