To Be Software Developers' Friends: Tool Development for Cryptographic Coding

Daphne Yao 姚丹凤
Professor
Virginia Tech

http://yaogroup.cs.vt.edu/

Inscrypt Keynote 2020

Acknowledgements

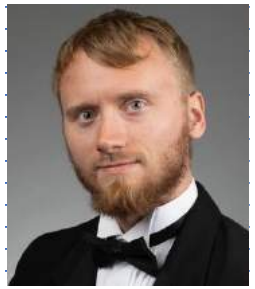
SaTC-1929701

Sazzadur Rahaman
Assistant Professor
University of Arizona

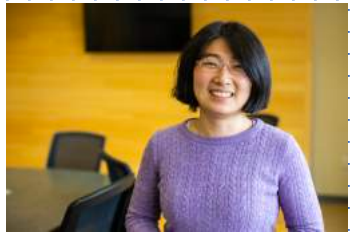
Ya Xiao, Miles Frantz
Sharmin Afrose, Na Meng
Virginia Tech

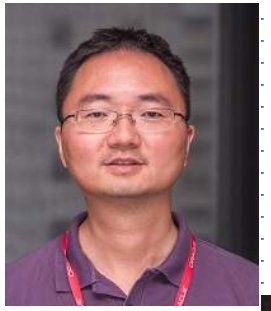
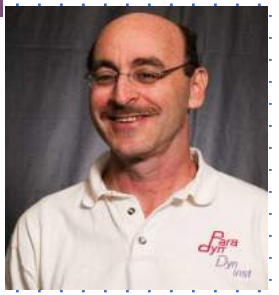
Yang Zhao
Cristina Cifuentes
Oracle Lab Australia

Barton Miller
UW-Madison

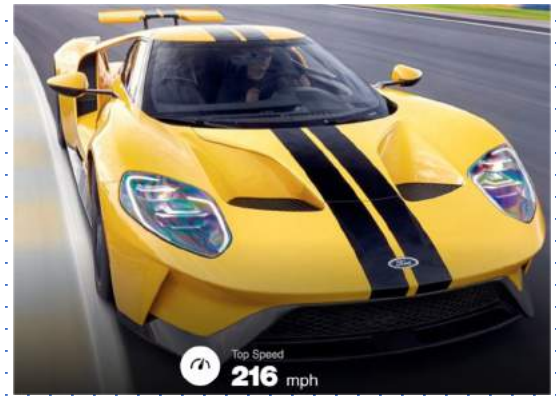







Software is everywhere



Ford GT has over 10 million lines of code

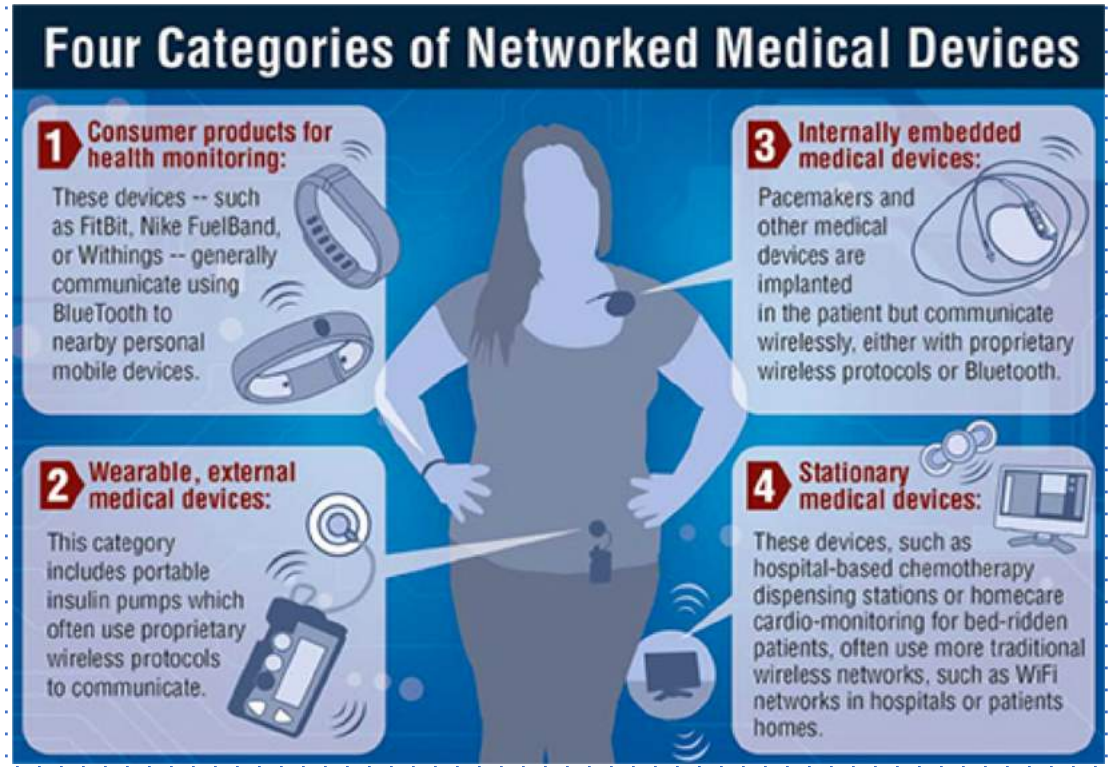
F-22 Raptor has 2 million lines of code

Boeing 787 Dreamliner has 7 million lines of code

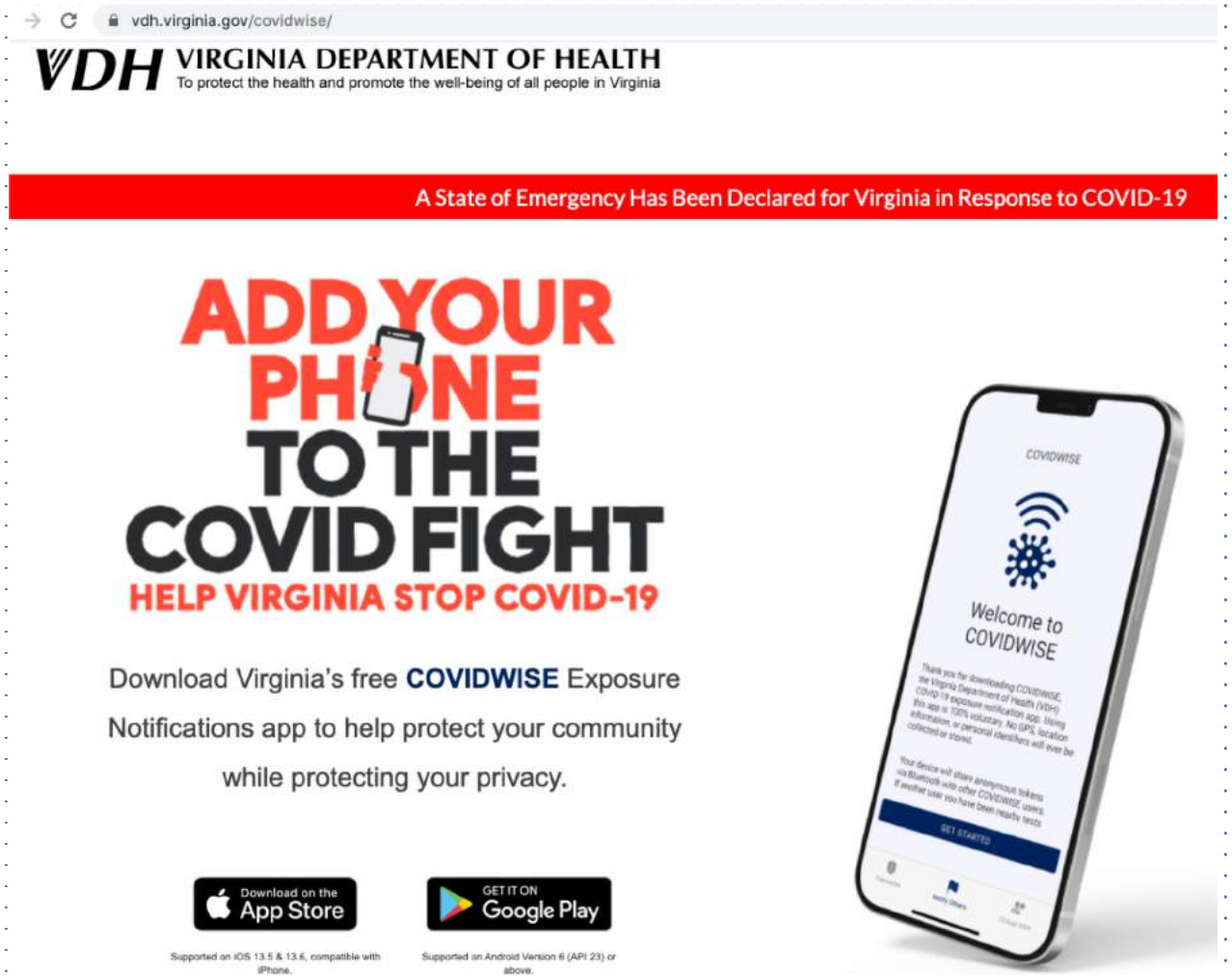
Ford pickup truck F-150 has 150 million lines of code

https://www.eitdigital.eu/news-events/blog/article/guess-what-requires-150-million-lines-of-code/

Developers' code is getting closer and closer to your body



Virginia State's contact tracing app using **Google/Apple's exposure notification library**



We need both -- developer training & using tools

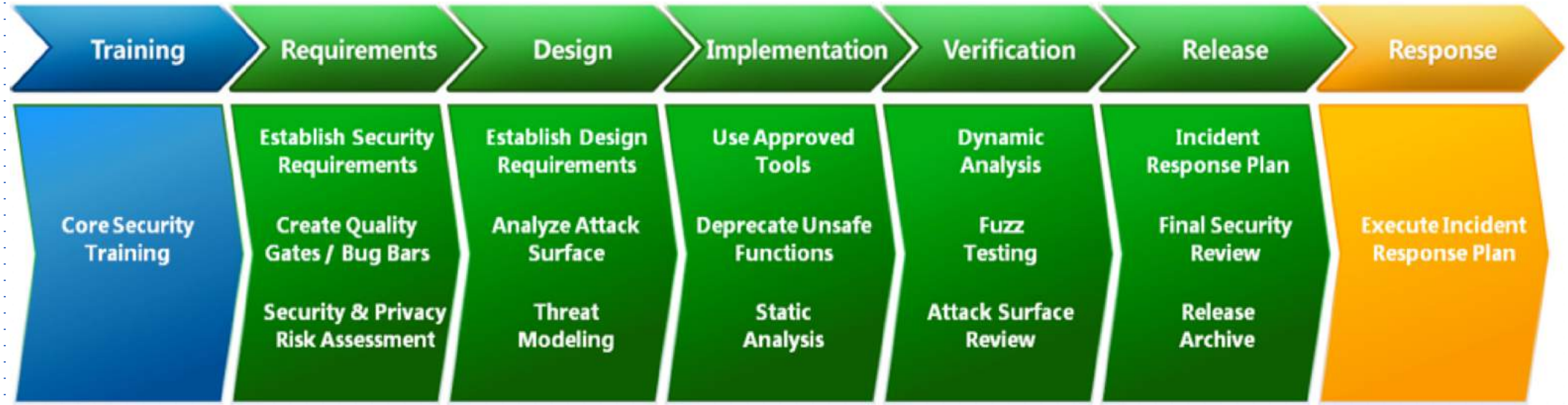
Top 10 secure coding rules

1. Validate input. Validate input from all untrusted data sources.
2. Heed compiler warnings [and other warnings].
3. Architect and design for security policies.
4. Keep it simple.
5. Default deny.
6. Adhere to the principle of least privilege.
7. Sanitize data sent to other systems.
8. Practice defense in depth.
9. Use effective quality assurance techniques.
10. Adopt a secure coding standard.

https://wiki.sei.cmu.edu/confluence/display/seccode/Top+10+Secure+Coding+Practices

Microsoft secure development lifecycle (SDL)

Developers need TOOLS and more TOOLS



https://social.technet.microsoft.com/wiki/contents/articles/7100.the-security-development-lifecycle.aspx

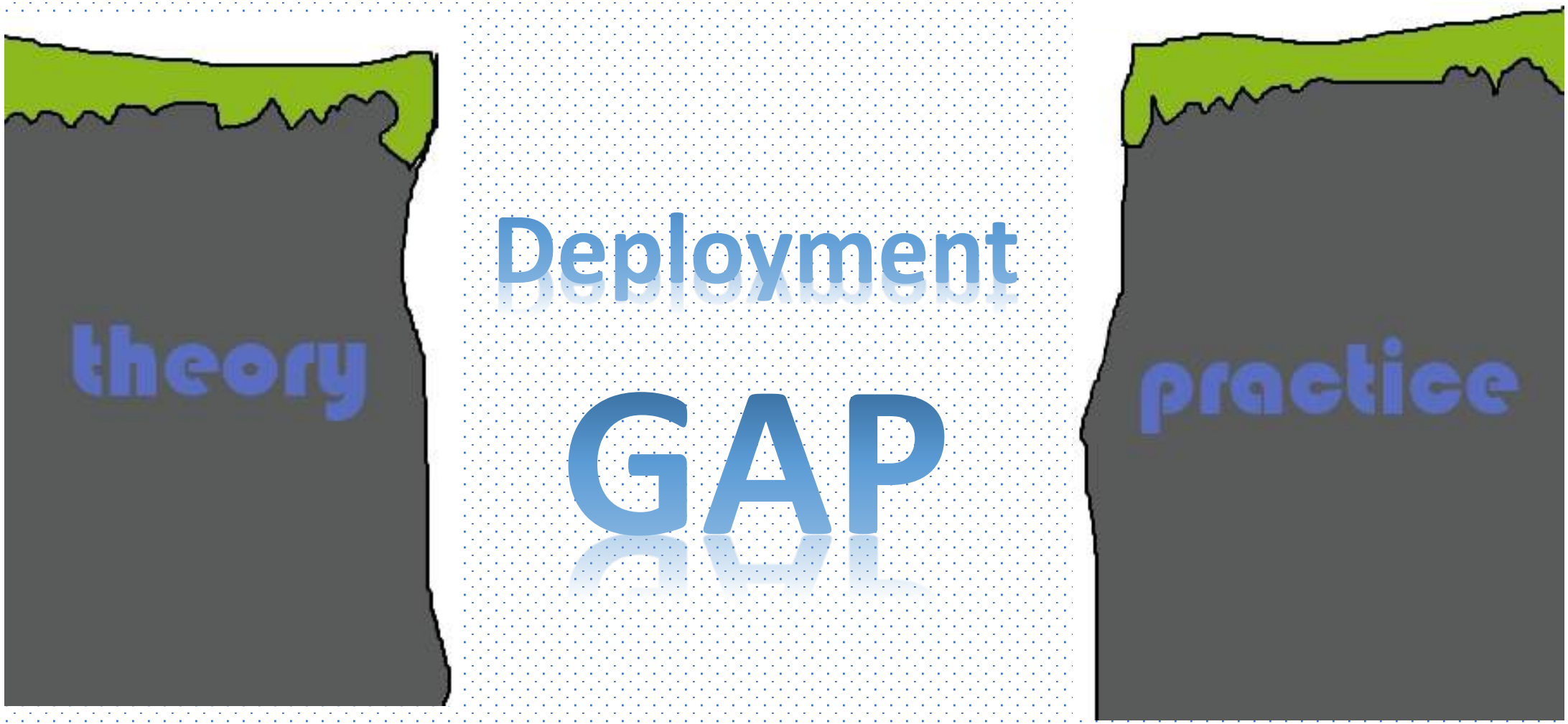
Who would not want to write secure code?

Budget

False positives

Time

Resources


theory
Deployment
GAP
practice

CSRF token in Java -- an example of the gap

Cross-Site Request Forgeries: Exploitation and Prevention

William Zeller* and Edward W. Felten*†
*Department of Computer Science
*Center for Information Technology Policy
†Woodrow Wilson School of Public and International Affairs
Princeton University
{wzeller,felten}@cs.princeton.edu

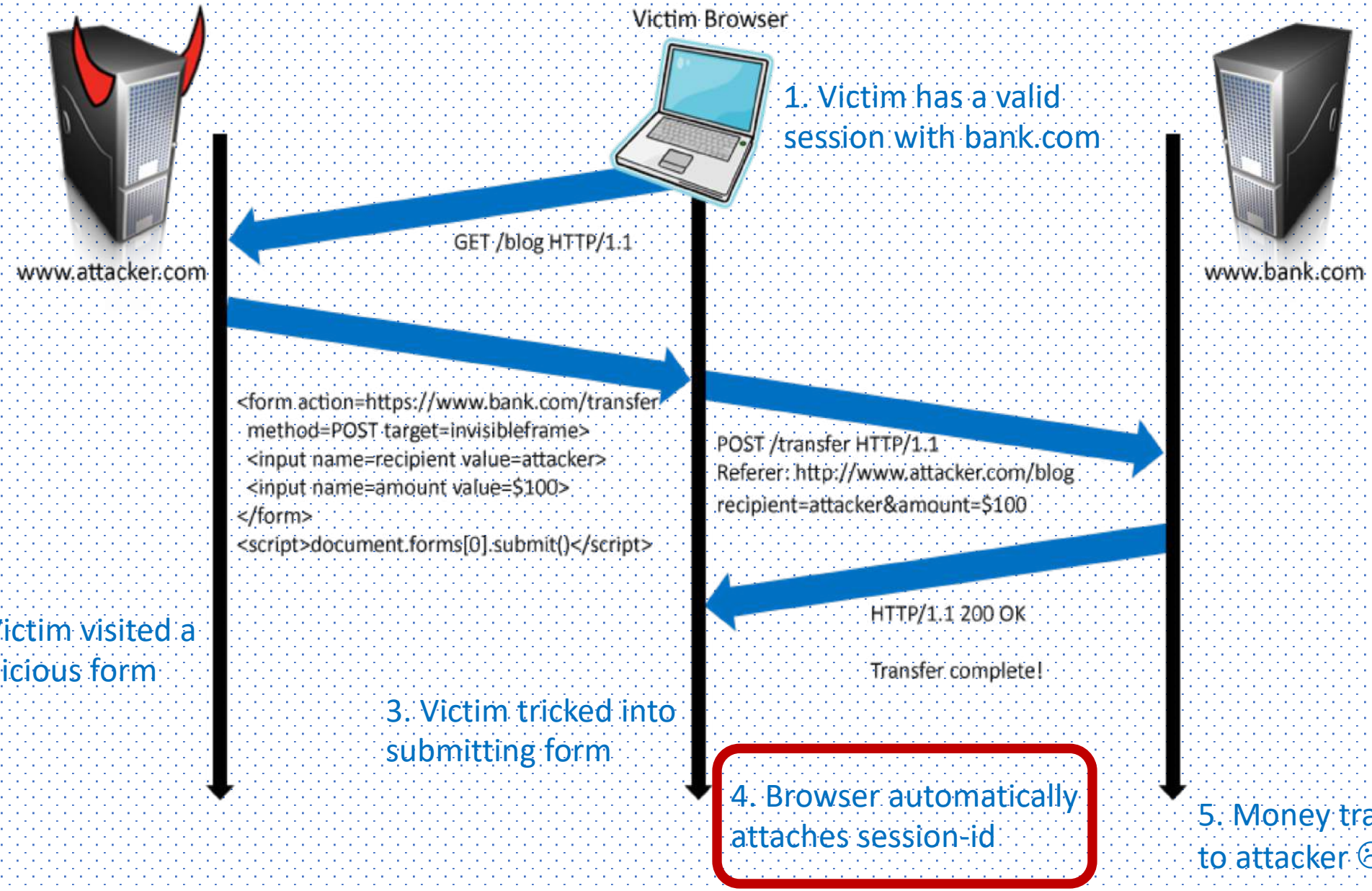
Revision 10/15/2008: Noted that the New York Times has fixed the vulnerability described below. Also clarified that our server-side CSRF protection recommendations *do*

1 Introduction

Cross-Site Request Forgery[1] (CSRF) attacks occur when a

[PDF] Robust Defenses for Cross-Site Request Forgery - Stanford Security Lab
https://seclab.stanford.edu/websec/csrf/csrf.pdf ▾
by A Barth - 2008 - Cited by 456 - Related articles
Collin Jackson. **Stanford** ... Cross-Site Request Forgery (**CSRF**) is a widely exploited web site ... the header can be used today as a reliable **CSRF** defense.

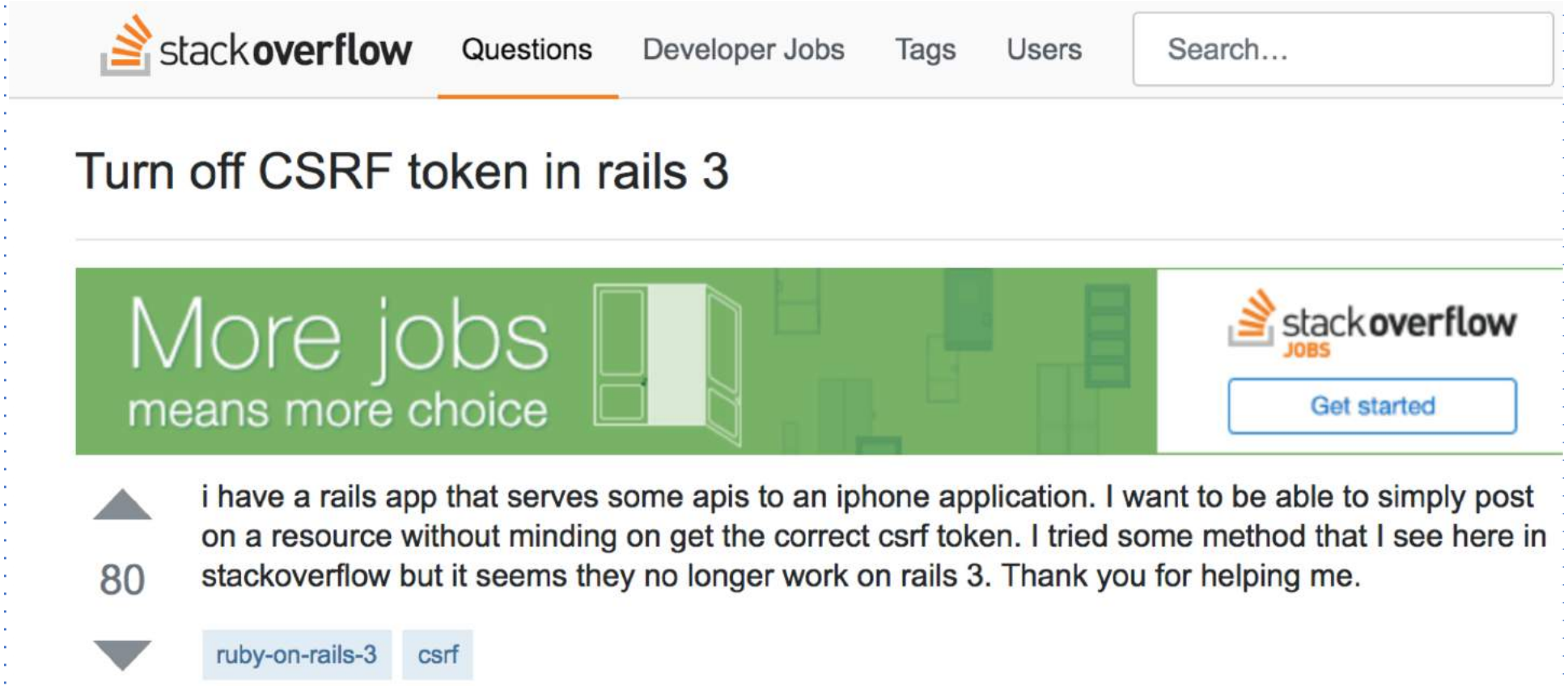
What is cross-site request forgery (CSRF) attack?



From C. Jackson

Developers need help

"Addingcsrf().disable() solved the issue!!! I have no idea why it was enabled by default" – a StackOverflow post



[Meng ICSE 2018] Available at: https://arxiv.org/pdf/1709.09970.pdf

Real quotes from StackOverflow developers forum

"Addingcsrf().disable() solved the issue!!! I have no idea why it was enabled by default"

"adding -Dtrust_all_cert=true to VM arguments"

"I want my client to accept any certificate (because I'm only ever pointing to one server)"

```
1  // Create a trust manager that does not validate certificate chains
2  TrustManager[] trustAllCerts = new TrustManager[]{
3    new X509TrustManager() {
4      public java.security.cert.X509Certificate[]
           getAcceptedIssuers() {return null;}
5      public void checkClientTrusted(...) {}
6      public void checkServerTrusted(...) {} }};
7  // Install the all-trusting trust manager
8  try {
9    SSLContext sc = SSLContext.getInstance("SSL");
10   sc.init(null, trustAllCerts, new java.security.
         SecureRandom());
11   HttpsURLConnection.setDefaultSSLSocketFactory(sc
         .getSocketFactory());
12 } catch (Exception e) {}
```

N. Meng, S. Nagy, D. Yao, W. Zhuang, and G. Argoty. *ICSE* 2018

Influencers -- how much influence does StackOverflow have?

Insecure Posts	Total Views	No. of Posts	Min Views	Max Views	Average
Disabling CSRF Protection*	39,863	5	261	28,183	7,258
Trust All Certs	491,567	9	95	391,464	58,594
Obsolete Hash	91,492	3	1,897	86,070	30,497
Total Views	**622,922**	**17**	-	-	-

As of August 2017

Insecure StackOverflow posts seem to have a large influence on developers ☹

N. Meng, S. Nagy, D. Yao, W. Zhuang, and G. Argoty. *ICSE* 2018

* In Java Spring Security for web applications

Cyberbully on StackOverflow developers forum

User: skanga [0]

“Do NOT EVER trust all certificates. That is very dangerous.”

“"accepted answer" is wrong and INDEED it is DANGEROUS. Others who blindly copy that code should know this.”

User: MarsAtomic [6,287]

“once you have sufficient reputation you will be able to comment”

“If you don't have enough rep to comment, … then participate … until you have enough rep.”

https://stackoverflow.com/questions/10594000/when-i-try-to-convert-a-string-with-certificate-exception-is-raised

The media drives a wedge between software developers and security researchers ☹

The truth is –

Developers need help to write secure crypto code

A simple vulnerability example

Constant keys defined & used in the same method (intra-procedural)

Insecure

```
String defaultKey = " Inscrypt";
byte[] keyBytes = defaultKey.getBytes();
keyBytes = Arrays.copyOf(keyBytes,16);
SecretKeySpec keySpec = new SecretKeySpec(keyBytes, "AES");
```

Secure

```
SecureRandom random = new SecureRandom();
String defaultKey = String.valueOf(random.ints());
byte[] keyBytes = defaultKey.getBytes();
keyBytes = Arrays.copyOf(keyBytes,16);
SecretKeySpec keySpec = new SecretKeySpec(keyBytes, "AES");
```

Need to recognize more complex vulnerability patterns

Multi-class/method data-flow

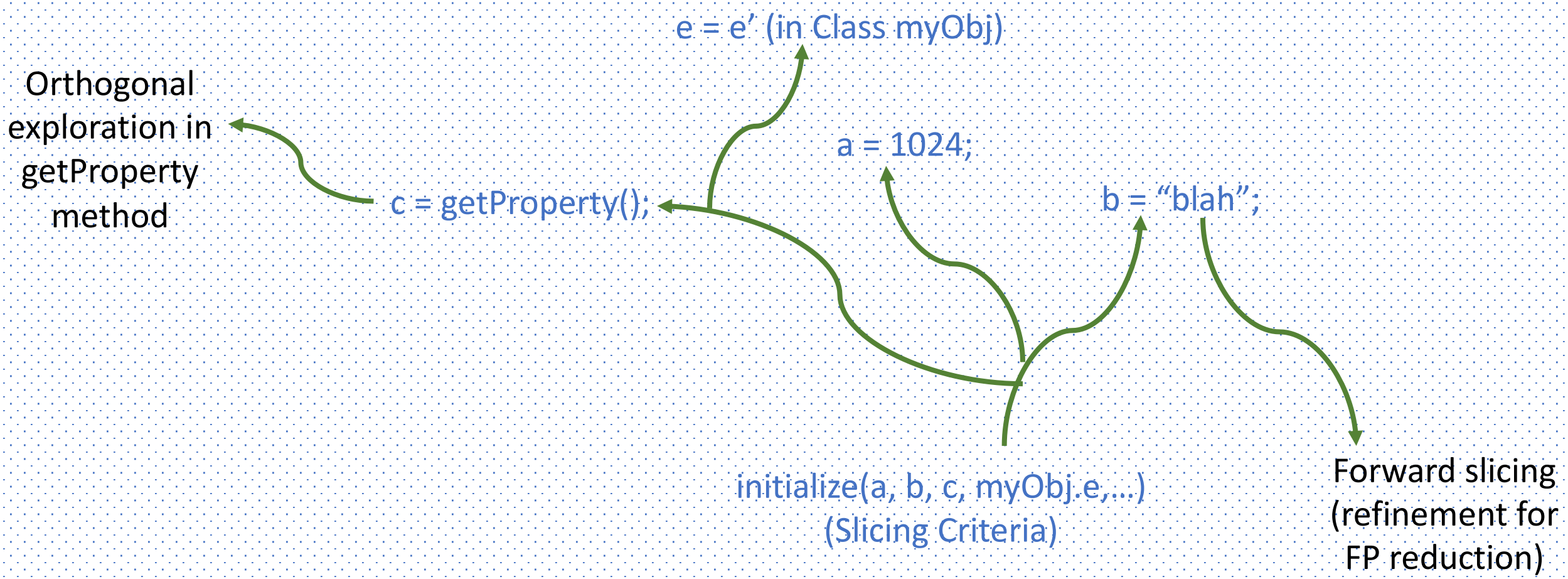
```
1  public class MultipleClass1 {
2     public void method1 (String passedAlgo) {
3        MultipleClass2 mc = new MultipleClass2 ();
4        mc.method2 (passedAlgo);
5     }
6  }
7  public class MultipleClass2 {
8        public void method2 (String cryptoAlgo) {
9        Cipher cipher = Cipher.getInstance (cryptoAlgo);
10      }
11 }
```

Lack of hostname verification (TLS)

```
1  public class SecDevTM implements X509TrustManager {
2      private X509TrustManager defaultTM;
3      ...
4      @Override
5      public void checkServerTrusted(X509Certificate[] chain, St
6      throws CertificateException {
7          try{
8              defaultTM.checkServerTrusted(chain, authType);
9          }
10         catch(CertificateException e){
11             Log.w("checkServerTrusted",e.toString());
12         }
13     }
14 }
```

Need to throw an exception

Detection approach – Mapping crypto properties → program analysis



Support flow-, Context-, Field-sensitive analysis

Crypto in Android App Libraries (on 6,181 apps)

96% of issues coming from libraries

	Rules
2	Predictable pwds for PBE
3	Predictable pwds for keystores
4	**Dummy hostname verifier**
5	**Dummy cert. verifier**
7	Use of HTTP
9	Weak PRNG
12	Static IV
16	Broken hash

[Sazzadur et al. ACM CCS 2019]

Crypto Code in Java Can Be Complex to Analyze

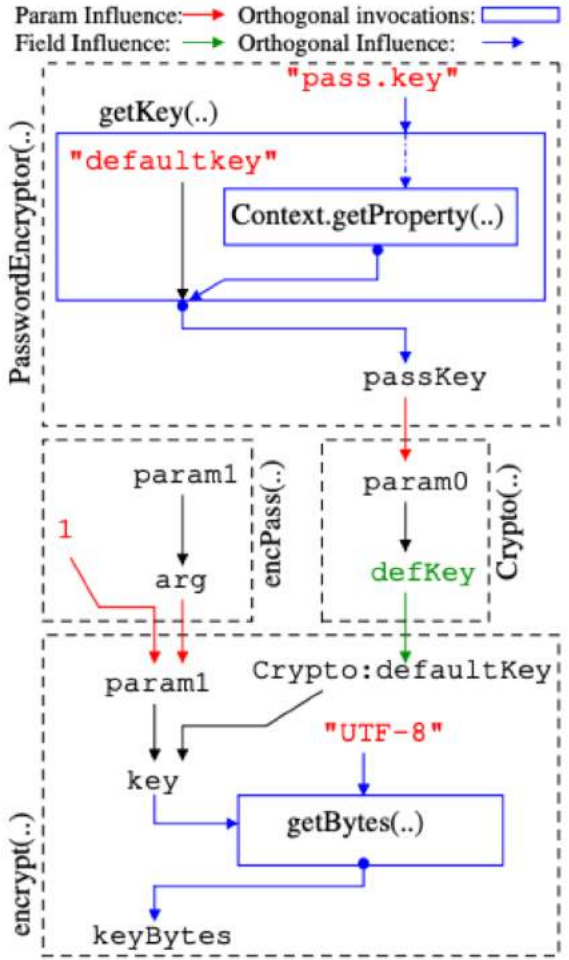
```
1 class PasswordEncryptor {
2
3  Crypto crypto;
4
5  public PasswordEncryptor(){
6   String passKey = PasswordEncryptor
                          .getKey("pass.key");
7    crypto = new Crypto(passKey); (p)
8  }
9
10 byte[] encPass(String [] arg){
11  return crypto.encrypt(arg[0], arg[1]); (p)
12 }
13
14 static String getKey(String src){
15  String key = Context.getProperty(src);
16  if (key == null) {
17    key = "defaultkey";
18  }
19  return key;
20  }
21 }
```

```
22 class Crypto {
23
24  String ALGO = "AES";
25  String ALGO_SPEC = "AES/CBC/NoPadding";
26  String defaultKey;
27  Cipher cipher;
28
29  public Crypto(String defKey){
30   cipher = Cipher.getInstance(ALGO_SPEC);
31   defaultKey = defKey; // assigning field
32  }
33
34  byte[] encrypt(String txt,String key){
35   if (key == null){
36    key = defaultKey; (f)
37   }
38   byte[] keyBytes = key.getBytes("UTF-8");
39   byte[] txtBytes = txt.getBytes();
40   SecretKeySpec keySpc =

   new SecretKeySpec(keyBytes, ALGO);
41   cipher.init(Cipher.ENCRYPT_MODE,keySpc);

42   return cipher.doFinal(txtBytes);}}
```

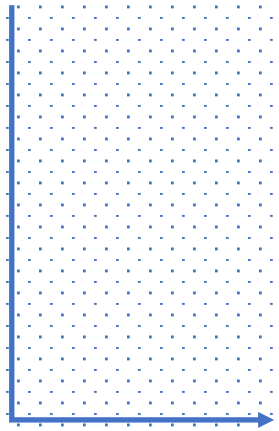
(a)



(b)

[Sazzadur et al. ACM CCS 2019]

Too Many Security Irrelevant Constants



False positives (i.e., false alarms) ☹

False alarms are counter-productive

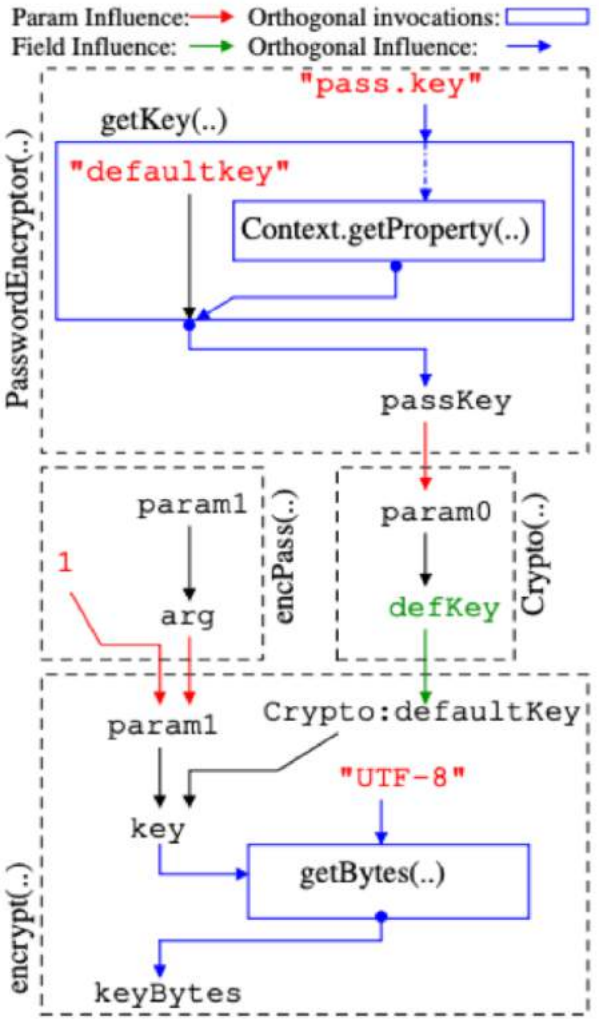
https://tastyislandhawaii.com/2018/01/16/hawaiis-humorous-reactions-to-the-false-ballistic-missile-alert/

Irrelevant vs. irrelevant constants

“UTF-8”: irrelevant (for encoding)

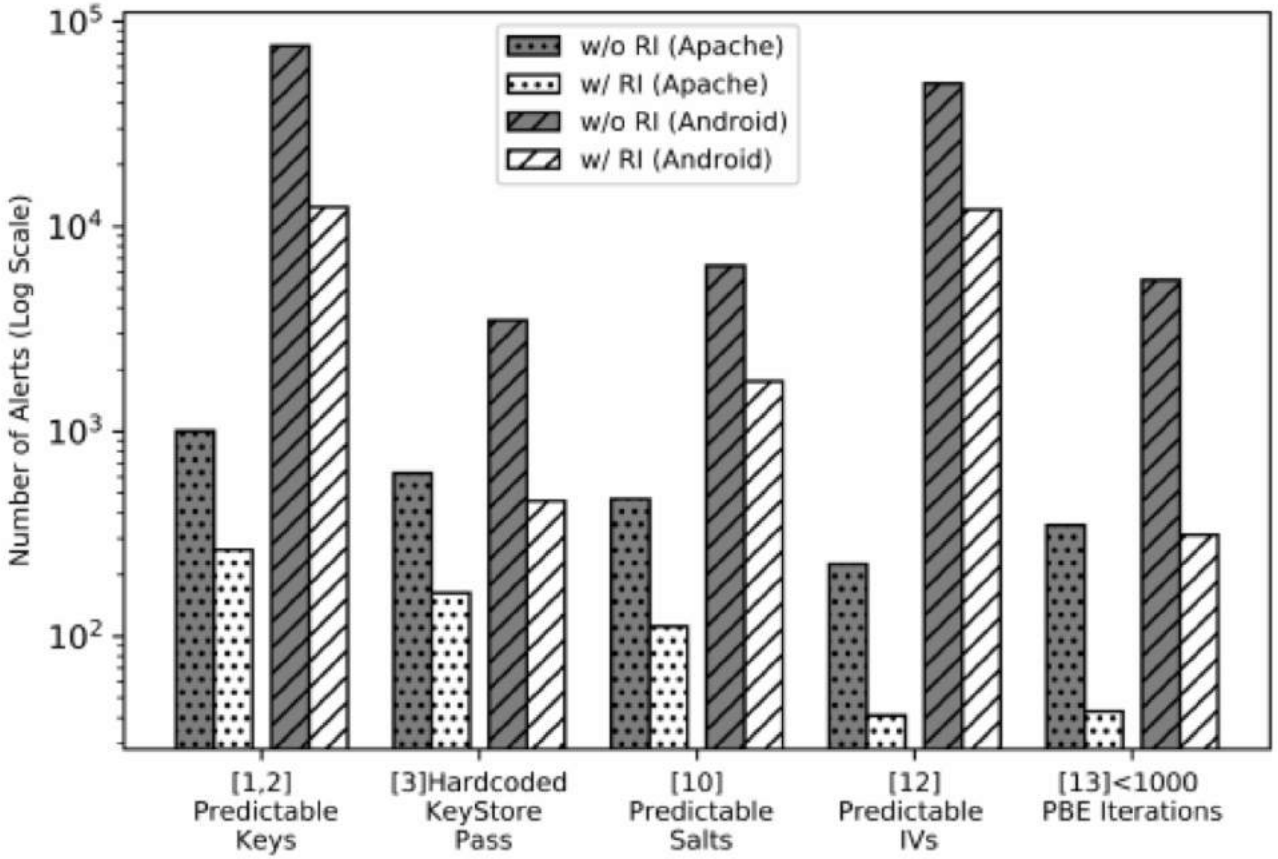
“pass.key”: irrelevant (a file name)

“defaultkey”: what we need (a hardcoded pwd)



Reduction of FPs by Refinements -- Off-the-shelf Program Slicing Would Fail

Reduce false alerts by 76% to 80%



Reduction results of FPs with refinement insights in 46 Apache projects

[Sazzadur et al. ACM CCS 2019]

Deployment-quality Accuracy and Scalability

Max, min, & avg LoC:
2,571K (Hadoop), 1.1K (Commons Crypto), & 402K, respectively

Detected insecure PBE code from Apache Ranger (They fixed it)

```
1  PBEKeySpec getPBEParameterSpec(String password) throws Throwable {
2      MessageDigest md = MessageDigest.getInstance(MD_ALGO);
3      byte[] saltGen = md.digest(password.getBytes());
4      byte[] salt = new byte[SALT_SIZE];
5      System.arraycopy(saltGen, 0, salt, 0, SALT_SIZE);
6      int iteration = password.toCharArray().length + 1;
7       return new PBEKeySpec(password.toCharArray(), salt, iteration);
8   }
```

Defined earlier as MD5 , no good ☹

Salt should not depend on pwd ☹

Should be 1000 iterations ☹

Side-channel leak, as iteration/runtime reveals the pwd length ☹

What Exactly is Deployable Accuracy?

98.6% Precision

Out of 1,295 Apache alerts, 18 are false alarms

[Sazzadur et al. ACM CCS 2019]

Crypto API Benchmarks -- driving up the industry standards

SpotBugs

CogniCrypt

coverity
a higher code

- 171 man-made test units
- 40 basic cases
- 131 advanced cases
- 16 crypto rules

Benchmark based on Apache software to come!

https://github.com/CryptoGuardOSS/cryptoapi-bench

What does industrial strength code scanner look like?

Oracle's Parfait – an industrial strength static analysis tool for software security (started in 2007)

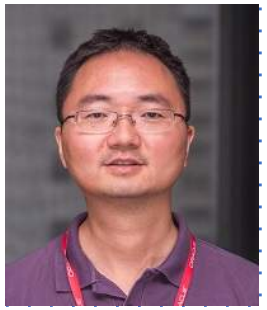
Parfait is fast -- analyzing 10.6 million of lines of code in 80 mins on a 2.9GHz AMD computer

Parfait is precise -- average false positive rate < 10%

Cristina Cifuentes and her team

https://labs.oracle.com/pls/apex/f?p=labs:49:::::P49_PROJECT_ID:13

Oracle Lab Australia implemented CryptoGuard's approach (2019) to scan production code


arXiv.org > cs > arXiv:2007.06122

Search...

Help | Advar

Computer Science > Software Engineering

[Submitted on 12 Jul 2020]

Industrial Experience of Finding Cryptographic Vulnerabilities in Large-scale Codebases

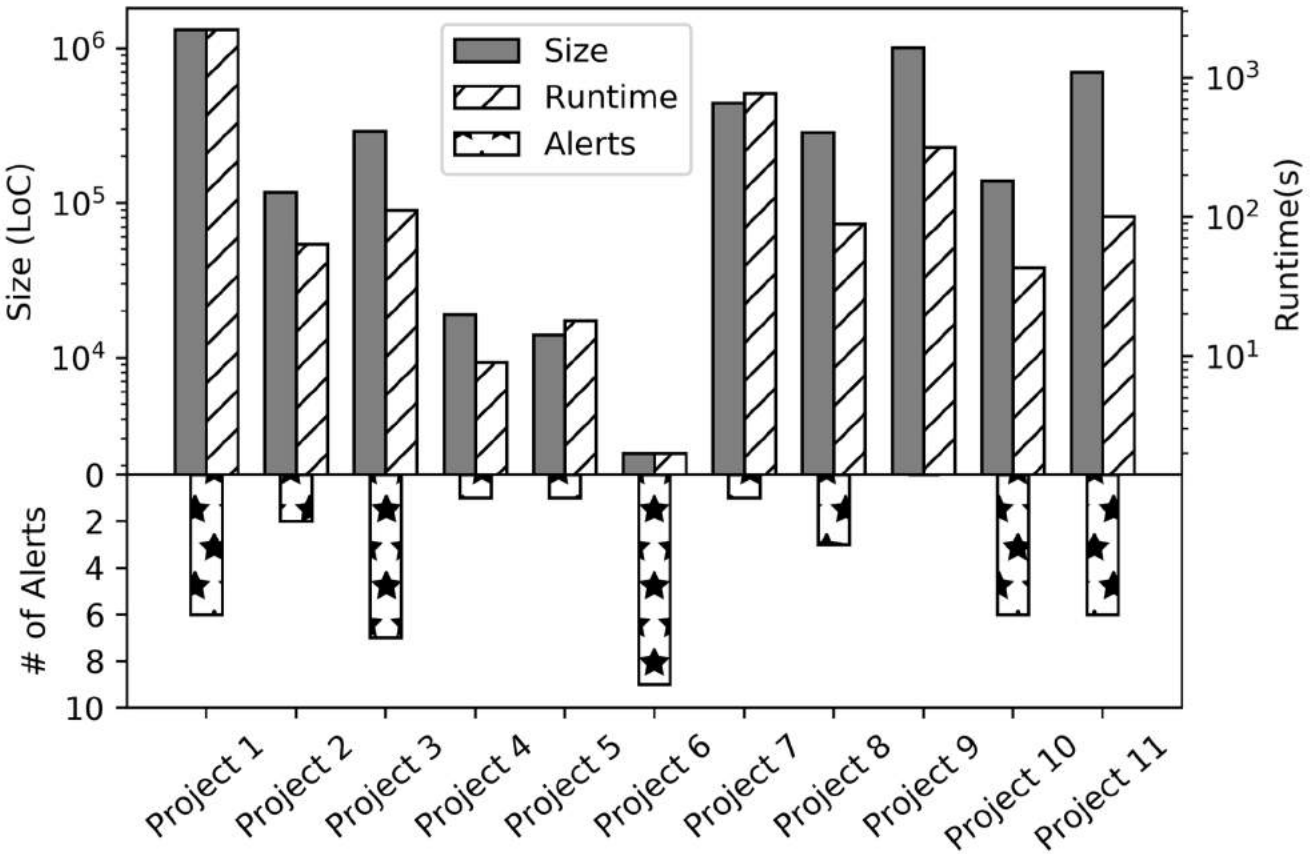
Ya Xiao, Yang Zhao, Nicholas Allen, Nathan Keynes, Danfeng (Daphne)Yao, Cristina Cifuentes

Enterprise environments need to screen large-scale (millions of lines of code) codebases for vulnerability detection, resulting in high requirements for precision and scalability of a static analysis tool. At Oracle, Parfait is one such bug checker, providing precision and scalability of results, including interprocedural analyses. CryptoGuard is a precise static analyzer for detecting cryptographic vulnerabilities in JavaTM1 code built on Soot. In this paper, we describe how to integrate CryptoGuard into Parfait, with changing intermediate representation and relying on a

https://arxiv.org/abs/2007.06122

Results of Parfait's crypto scanning 11 internal Oracle projects (Java) -- detection approach based on CryptoGuard

- Scanned 11 projects; reported 42 vulnerabilities with 0 false positive (100% precision)
- Average runtime 338.8s for 11 projects with average 395.4k LoC



Scanning on Oracle internal projects

Parfait's benchmark evaluation (on CryptoAPI-Bench)

How many actual vulnerabilities are reported? Higher the better ☺

98.4% Recall

86.6% Precision -- **100%** precision if excluding path sensitive cases

How many reported alerts are real vulnerabilities? Higher the better ☺

Type	Total Cases	Insecure Cases	Secure Cases	Reported Cases	False Positives	False Negatives	Precision	Recall
Basic Cases	27	24	3	24	0	0	100%	100%
Multiple methods	57	56	1	54	0	2	100%	96.43%
Multiple Classes	23	18	5	18	0	0	100%	100%
Field Sensitivity	19	18	1	18	0	0	100%	100%
Path Sensitivity	19	0	19	19	19	0	0 %	0 %
Heuristics	13	9	4	9	0	0	100%	100%
Total	**158**	**125**	**33**	**142**	**19**	**2**	**86.62%**	**98.40%**

Ongoing work in my group on crypto API recommendation with deep learning

Ya Xiao
(4-th year PhD student)

Bimal Viswanath
(Virginia Tech)

Xinyang Ge
(Microsoft Research)

Which API to use in Line 6?

```
1  public byte[] encryptGCM(byte[] plaintext, byte[] keyBytes,
   ↪  byte[] iv){
2      SecretKey key = new SecretKeySpec(keyBtes, "AES");
3      IvParameterSpec ivSpec = new IvParameterSpec(iv);
4      Cipher cipher = createCipher("AES/GCM/NoPadding", "BC",
       ↪  Cipher.ENCRYPT_MODE, key,ivSpec);
5      ByteArrayOutputStream byteArrayOutputStream = new
       ↪  ByteArrayOutputStream();
6      //...
7  }
```

✓ `6  CipherOutputStream cipherOutputStream = new CipherOutputStream(byteArrayOutputStream, cipher);`

✗ `6  byteArrayOutputStream.write(…)`

✗ `6  Cipher.doFinal(…)`

✗ `6  Cipher.updateAAD(…)`

Current results:
98.99% top-1 accuracy in predicting the last API in a sequence

High-frequency or obvious choices may not be correct ☹

Need more research addressing practical deployment challenges

IACR

Real World Crypto Symposium

Real World Crypto Symposium aims to bring together cryptography researchers with developers implementing cryptography in real-world systems. The conference goal is to strengthen the dialogue between these two communities. Topics covered focus on uses of cryptography in real-world environments such as the Internet, the cloud, and embedded devices.

ACSAC 2020

December 7–11, 2020 • Online

Hard Topic Theme: *Deployable and Impactful Security*

Background

Since 2013, ACSAC has had a hard topic theme that focuses the conference on tackling a hard, cutting-edge, cybersecurity pr
requiring cooperation from government, industry, and academia. This year, ACSAC especially encourages contributions in th
Deployable and Impactful Security.

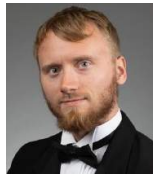
Check out our recent secure coding tutorial (IEEE SecDev 2020)

(In)secure crypto coding examples

Slides: http://yaogroup.cs.vt.edu/videos.html
Video: https://youtu.be/Z0RwBLURp9c

Secure TLS coding strategies

CryptoGuard intro/demo

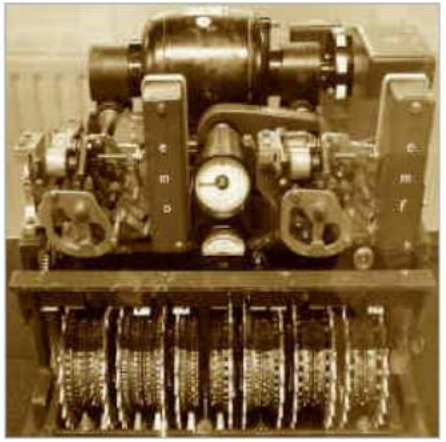


Tool eval benchmark

ACM NEWS

A Tool for Hardening Java Crypto

By R. Colin Johnson
July 23, 2020
Comments



Researchers at the Virginia Polytechnic Institute and State University (Virginia Tech) say the vulnerability checking software they developed is mature, and nearing deployment.

Credit: Wikimedia Commons

Identifying cryptographic vulnerabilities in today's million-line programs has become a critical endeavor. Because of the increasing sophistication of cybercriminals, programmers can no longer afford to test for vulnerabilities using only traditional debugging techniques, followed by releasing software, collecting bug reports and patching.

The new frontier being pursued by government, industry, and academia are automated tools that are capable of culling vulnerabilities before releasing source code into the wild. When run on existing software, such as the open-source Apache programs managing the world's servers, these tools also are finding a surprising number of vulnerabilities in software that is decades old.

Most open-source automated vulnerability checkers are still finding their way, but a team of researchers at the Virginia Polytechnic Institute and State University (Virginia Tech) claim to have vulnerability-checking software that is mature, and approaching deployment. Called CryptoGuard, the software automatically identifies cryptographic vulnerabilities in Java (and soon Python) source code. Funded by the U.S. Navy's Office of Naval Research (ONR) and the National Science Foundation (NSF), CryptoGuard is

Comm. Of ACM article on CryptoGuard: https://cacm.acm.org/news/246385-a-tool-for-hardening-java-crypto/fulltext

Related references

Papers:

- Sazzadur Rahaman, Ya Xiao, Sharmin Afrose, Fahad Shaon, Ke Tian, Miles Frantz, Murat Kantarcioglu, and Danfeng Yao. "Cryptoguard: High precision detection of cryptographic vulnerabilities in massive-sized Java projects." In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pp. 2455-2472. 2019.
- Sharmin Afrose, Sazzadur Rahaman, and Danfeng Yao. "CryptoAPI-Bench: A Comprehensive Benchmark on Java Cryptographic API Misuses." In *2019 IEEE Cybersecurity Development (SecDev)*, pp. 49-61. IEEE, 2019.
- Ya Xiao, Yang Zhao, Nicholas Allen, Nathan Keynes, and Cristina Cifuentes. "Industrial Experience of Finding Cryptographic Vulnerabilities in Large-scale Codebases." *arXiv preprint arXiv:2007.06122* (2020).

Online Resources:

- CryptoGuard. https://github.com/CryptoGuardOSS/cryptoguard
- CryptoAPI-Bench. https://github.com/CryptoGuardOSS/cryptoapi-bench
- Secure TLS/SSL code examples. https://github.com/AthenaXiao/SecureTLSCodeExample
- https://mybinder.org/v2/gh/franceme/cryptoguard/2020_SecDev_Tutorial

Questions and Comments?