Do security tools work?

- Adwait Nadkarni

Secure Platforms Lab

William & Mary
CHARTERED 1693


NEWS
FIRST DEBATE
PLAN YOUR VOTE
CORONAVIRUS
U.S. NEWS
OPINION
BUSINESS
WORLD
PODCASTS
TECH & MEDIA
'I'm in your baby's room': Nest cam hacks show risk of internet-connected devices
The breaches also point to a new hacking strategy that can compromise secure systems through the use of old passwords.
INTERNET OF SH*T —
When coffee makers are demanc
ransom, you know IoT is screwed
Watch along as hacked machine grinds, beeps, and spews water.
NEWS
c|net
COVID-19
BEST PRODUCTS
REVIEWS
NEWS
HOW TO
FINANCE
HEALTH
SMART HOME
CARS
Hacked Nest Cam convinces family that US is being attacked by North Korea
Nest says its systems weren't breached.
Richard Nieva, Laura Hautala Jan. 22, 2019 4:27 p.m. PT
5
Security Risks Can Be
y 3, 2019
Security flaws found in popular smart home devices
Science Nov 6, 2019 12:32 PM EDT

Home automation

Home automation

trigger

1 When Alice leaves, turn the Camera ON (*for **security***)

action

trigger

2 When Alice returns, turn the Camera Off (*for **privacy***)

action

Home Automation ***routines***/ ***IoT apps***/*programs*

Security Implications of Home Automation

Bob wants to steal from Alice
without being monitored

Security Implications of Home Automation

Bob wants to steal from Alice *without being monitored*

Bob tries to directly attack the security camera, *and fails*

Security Implications of Home Automation

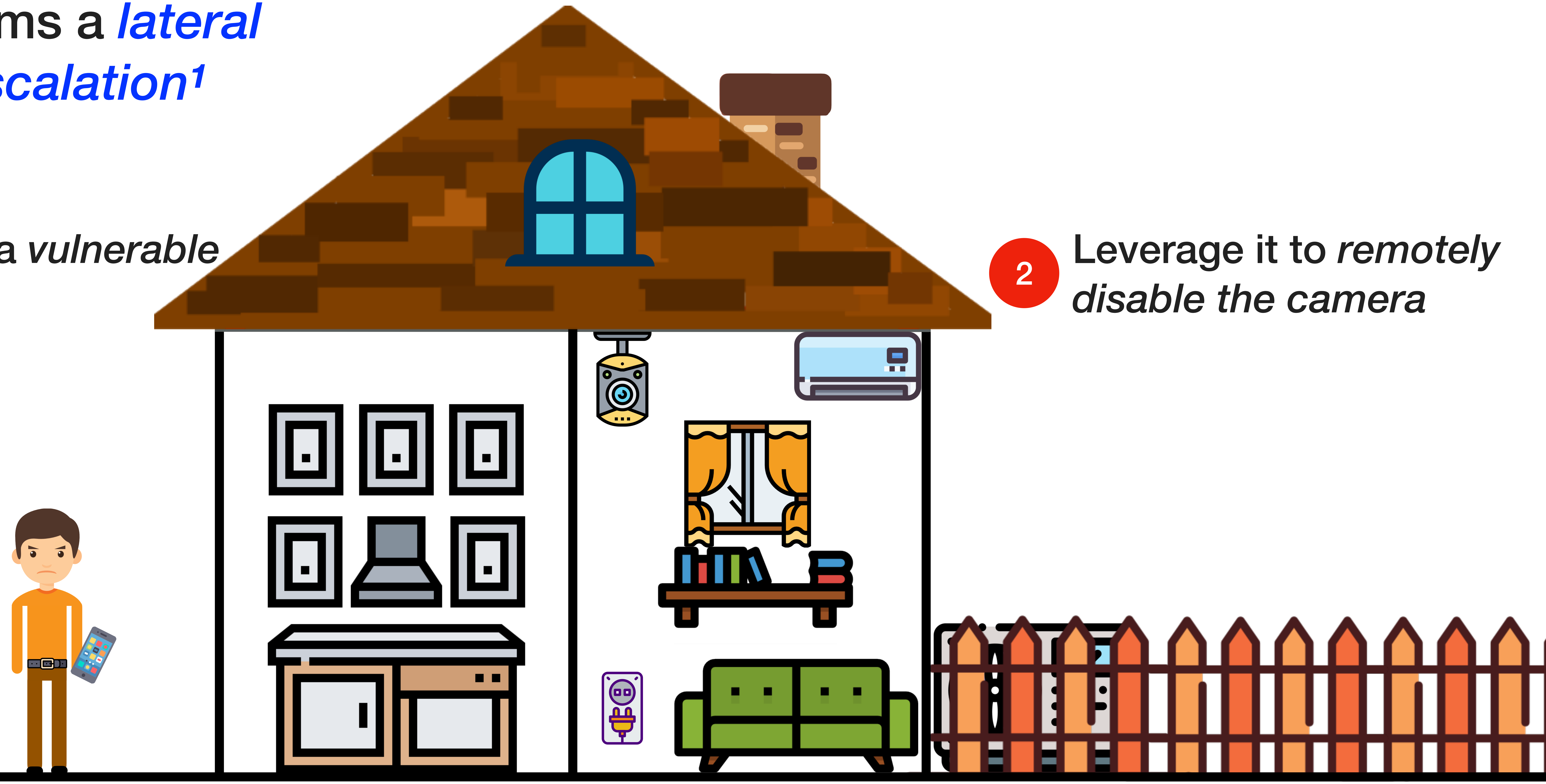
Bob wants to steal from Alice *without being monitored*

Bob tries to directly attack the security camera, *and fails*

However, **Bob** can *indirectly* attack the camera

Security Implications of Home Automation

Bob performs a *lateral privilege escalation*[1]



1. Compromise a *vulnerable* component
2. Leverage it to *remotely disable the camera*

[1] Kafle, Kaushal, Kevin Moran, Sunil Manandhar, Adwait Nadkarni, and Denys Poshyvanyk. ***A Study of Data Store-based Home Automation.*** In *Proceedings of the Ninth ACM Conference on Data and Application Security and Privacy (CODASPY)*, *Best Paper Award*.

Security Implications of Home Automation

Bob performs a *lateral privilege escalation*[1]

1 Compromise a *vulnerable* component

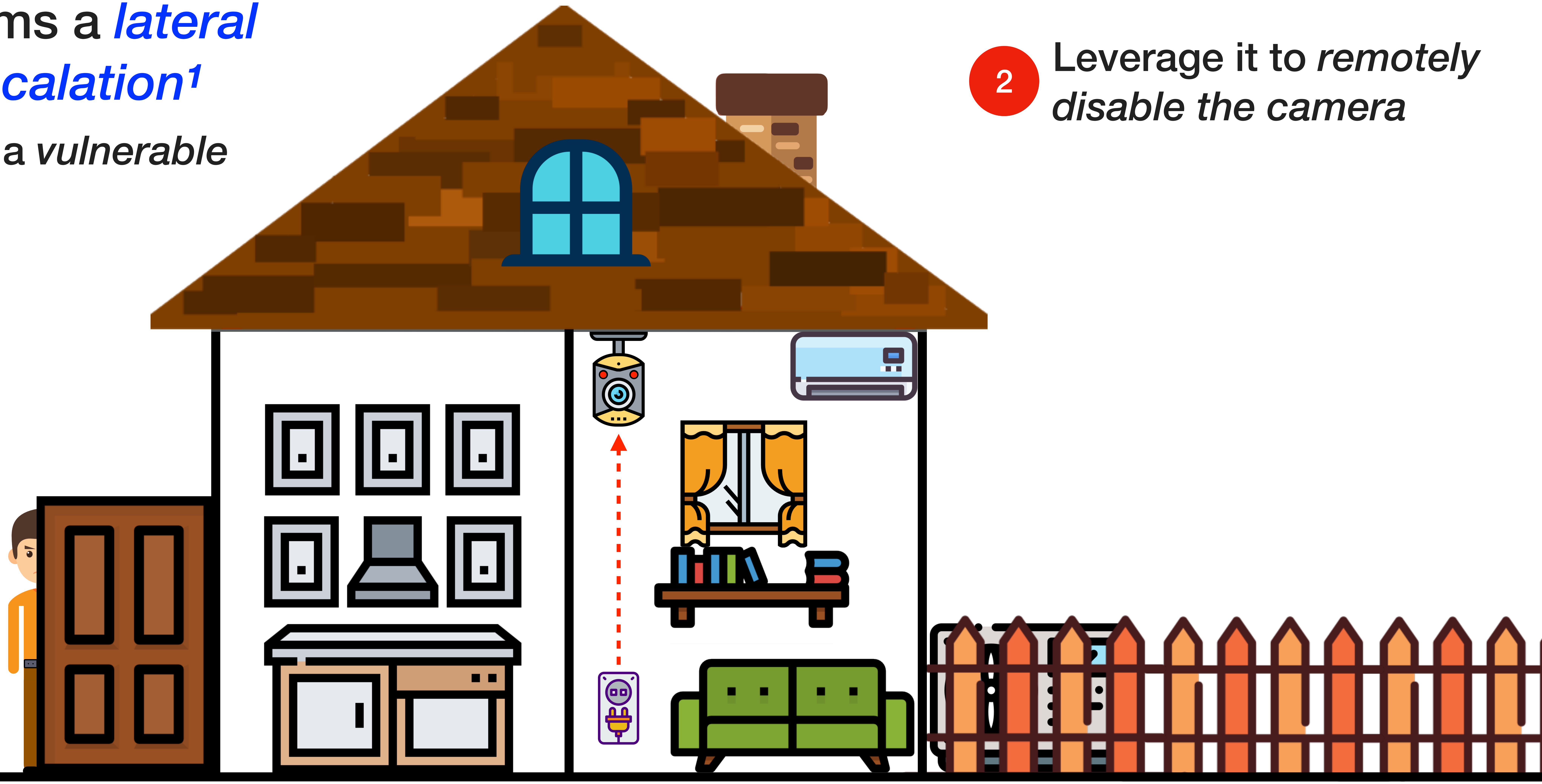
2 Leverage it to *remotely disable the camera*

[1] Kafle, Kaushal, Kevin Moran, Sunil Manandhar, Adwait Nadkarni, and Denys Poshyvanyk. ***A Study of Data Store-based Home Automation.*** In *Proceedings of the Ninth ACM Conference on Data and Application Security and Privacy (CODASPY)*, ***Best Paper Award.***

Security Implications of Home Automation *and* mobile-IoT apps

Bob performs a ***lateral privilege escalation***[1]

1. Compromise a *vulnerable* component
2. Leverage it to *remotely disable the camera*



[1] Kafle, Kaushal, Kevin Moran, Sunil Manandhar, Adwait Nadkarni, and Denys Poshyvanyk. ***A Study of Data Store-based Home Automation.*** In *Proceedings of the Ninth ACM Conference on Data and Application Security and Privacy* (CODASPY), *Best Paper Award*.

Security Implications of Home Automation *and* mobile-IoT apps

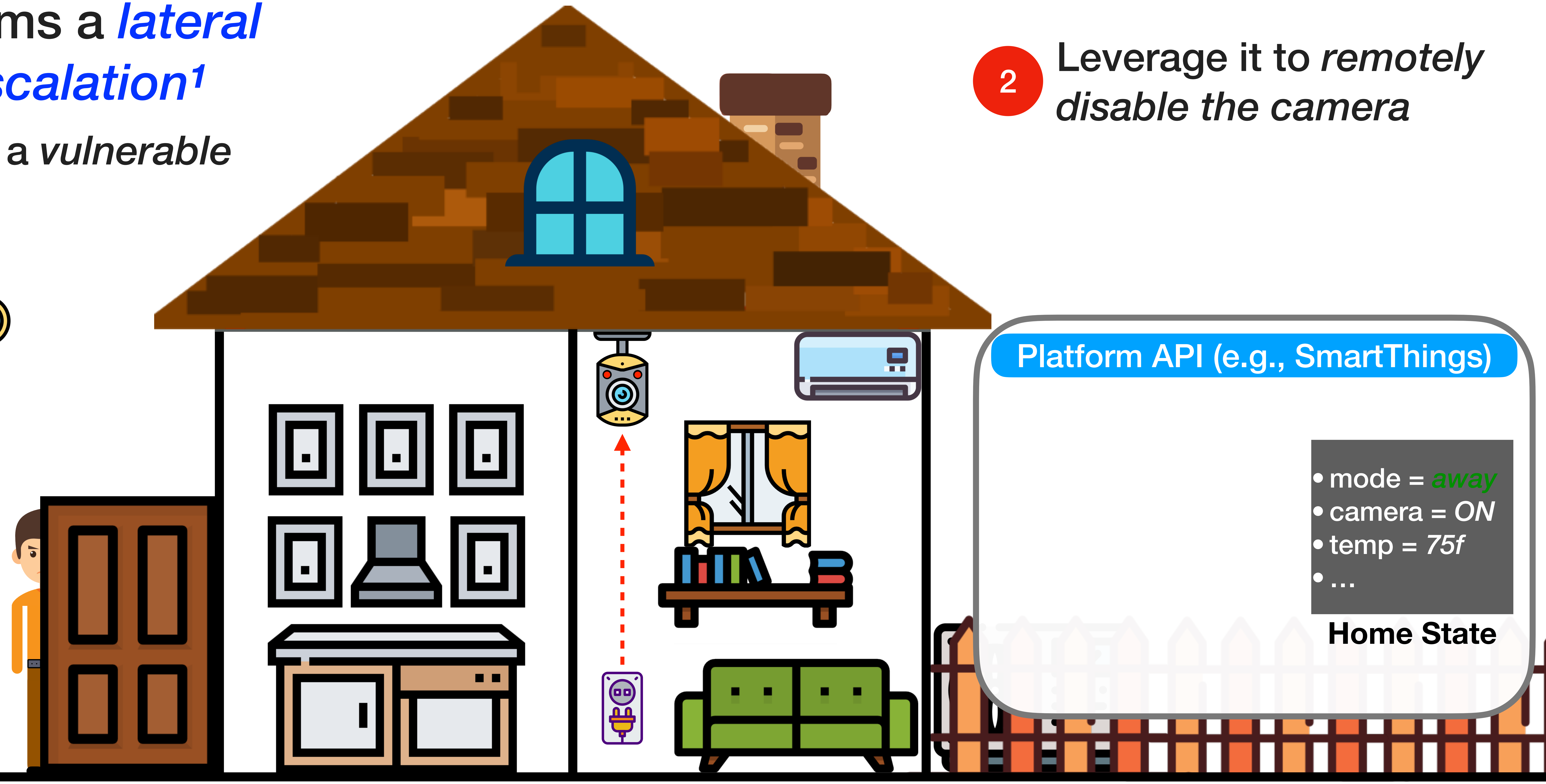
Bob performs a *lateral privilege escalation*[1]

1 Compromise a *vulnerable* component

TP Link Kasa app + MiTM Attack

Auth Token

2 Leverage it to *remotely disable the camera*

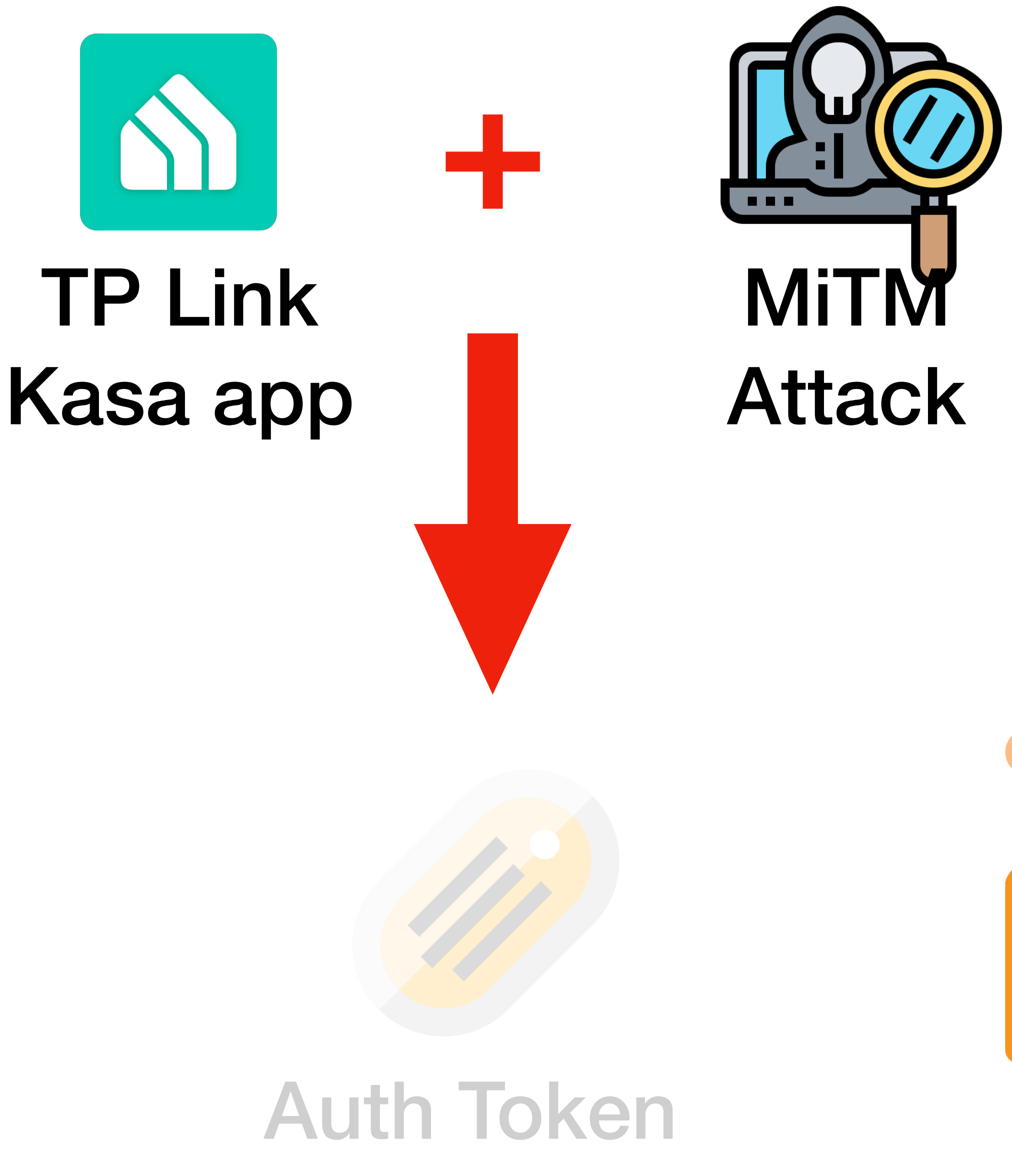


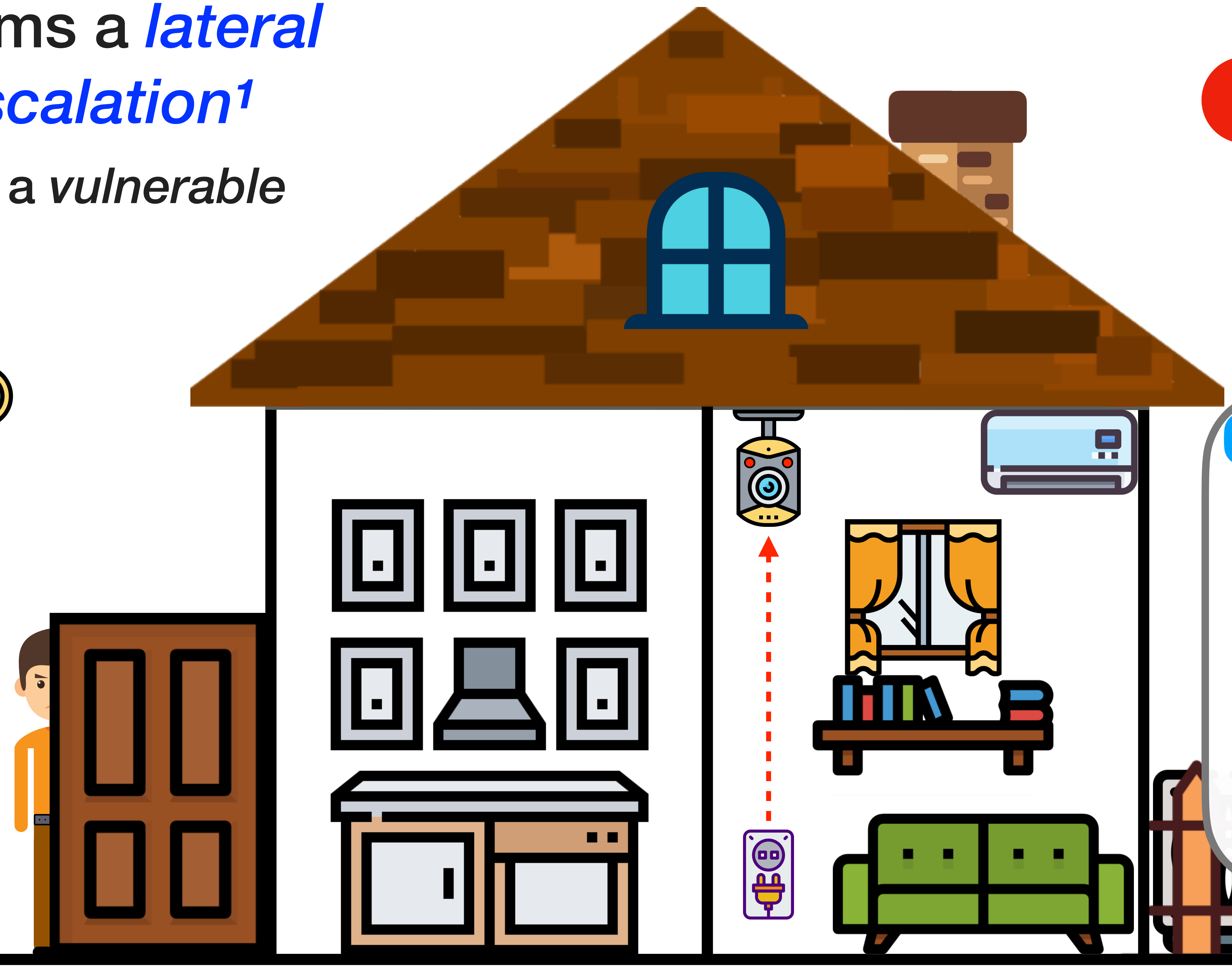
[1] Kafle, Kaushal, Kevin Moran, Sunil Manandhar, Adwait Nadkarni, and Denys Poshyvanyk. ***A Study of Data Store-based Home Automation.*** In *Proceedings of the Ninth ACM Conference on Data and Application Security and Privacy* (CODASPY), *Best Paper Award*.

Security Implications of Home Automation *and* mobile-IoT apps

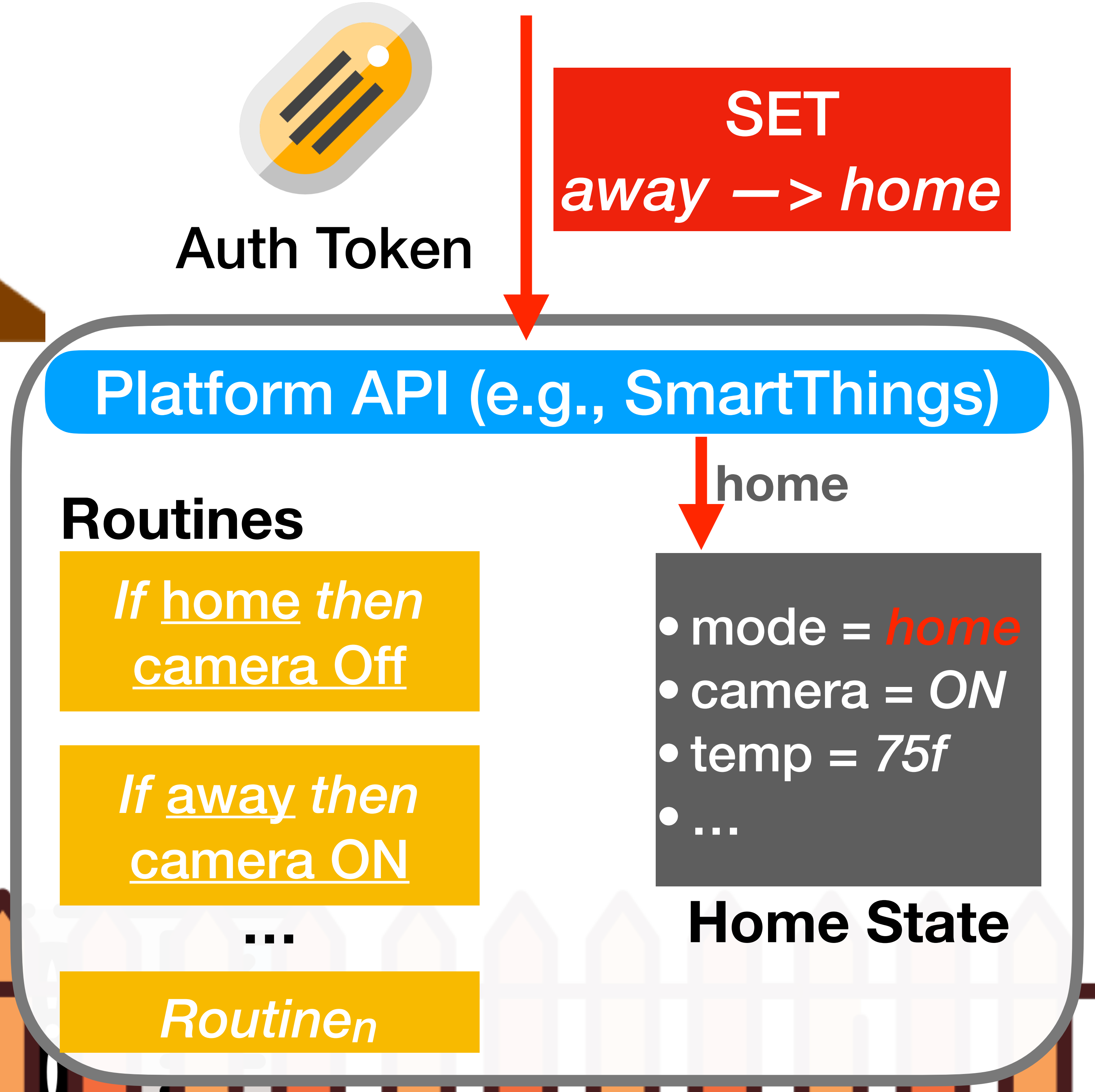
Bob performs a *lateral privilege escalation*[1]

1. Compromise a *vulnerable* component





2. Leverage it to *remotely disable the camera*



[1] Kafle, Kaushal, Kevin Moran, Sunil Manandhar, Adwait Nadkarni, and Denys Poshyvanyk. ***A Study of Data Store-based Home Automation.*** In *Proceedings of the Ninth ACM Conference on Data and Application Security and Privacy (CODASPY)*, *Best Paper Award*.

Security Implications of Home Automation *and* mobile-IoT apps

Bob performs a *lateral privilege escalation*[1]

1 Compromise a *vulnerable* component

2 Leverage it to *remotely disable the camera*

[1] Kafle, Kaushal, Kevin Moran, Sunil Manandhar, Adwait Nadkarni, and Denys Poshyvanyk. ***A Study of Data Store-based Home Automation.*** In *Proceedings of the Ninth ACM Conference on Data and Application Security and Privacy (CODASPY)*, *Best Paper Award*.

Security Implications of Home Automation *and* mobile-IoT apps

Bob performs a *lateral privilege escalation*[1]

1 Compromise a *vulnerable* component

2 Leverage it to *remotely disable the camera*

[1] Kafle, Kaushal, Kevin Moran, Sunil Manandhar, Adwait Nadkarni, and Denys Poshyvanyk. ***A Study of Data Store-based Home Automation.*** In *Proceedings of the Ninth ACM Conference on Data and Application Security and Privacy (CODASPY)*, *Best Paper Award*.

Security Implications of Home Automation *and* mobile-IoT apps

[1] Kafle, Kaushal, Kevin Moran, Sunil Manandhar, Adwait Nadkarni, and Denys Poshyvanyk. *A Study of Data Store-based Home Automation.* In *Proceedings of the Ninth ACM Conference on Data and Application Security and Privacy (CODASPY)*, *Best Paper Award.*

Security Implications of Home Automation *and* mobile-IoT apps

[1] Kafle, Kaushal, Kevin Moran, Sunil Manandhar, Adwait Nadkarni, and Denys Poshyvanyk. *A Study of Data Store-based Home Automation.* In *Proceedings of the Ninth ACM Conference on Data and Application Security and Privacy (CODASPY)*, *Best Paper Award*.

Several popular security tools, often integrated into CI/CD pipelines (e.g., GitHub Code Scan)

CryptoGuard

CogniCrypt

ShiftLeft

SpotBugs

Coverity

QARK

sonarqube

Codiga

Checkmarx

CODACY

{/code.scan}

SYNOPSYS

VERACODE

XANITIZER
... because security matters

SECURE CODE
WARRIOR

Do Security Tools Work?

RQ$_1$ — Do security tools and techniques detect the *vulnerabilities that they claim to detect*?

[**S&P'22**, TOPS'20, USENIX'18]

RQ$_2$ — Do the tools detect vulnerabilities *as developers expect them to*?

[**S&P'24**]

Real-world implications of failures of (automated) security analysis

[**USENIX'24**, NDSS'24, CCS'22, USENIX'22]

Context: Static Analysis Security Tools (SAST) used for detecting *crypto-API misuse*

CryptoGuard, ShiftLeft, CogniCrypt, QARK, SpotBugs, Coverity, sonarqube, Xanitizer ... because security matters

Evaluating Security Tools

RQ$_1$ — Do security tools and techniques detect the *vulnerabilities that they claim to detect*?

[**S&P'22**, TOPS'20, USENIX'18]

RQ$_2$ — Do the tools detect vulnerabilities *as developers expect them to*?

[S&P'24]

Real-world implications of failures of (automated) security analysis

[**USENIX'24**, NDSS'24, CCS'22, USENIX'22]

Context: Static Analysis Security Tools (SAST) used for detecting *crypto-API misuse*

CryptoGuard, ShiftLeft, CogniCrypt, QARK, SpotBugs, Coverity, sonarqube, SANITIZER ... because security matters

[1]https://web.archive.org/web/20201128090742/https://github.com/OWASP/Benchmark/issues/92

Rigorously Evaluating Security Tools

RQ$_1$ — Do security tools and techniques detect the *vulnerabilities that they claim to detect*?

[**S&P'22**, TOPS'20, USENIX'18]

RQ$_2$ — Do the tools detect vulnerabilities *as developers expect them to*?

[S&P'24]

Real-world implications of failures of (automated) security analysis

[**USENIX'24**, CCS'22, USENIX'22]

Context: Static Analysis Security Tools (SAST) used for detecting *crypto-API misuse*

CryptoGuard ShiftLeft CogniCrypt QARK

SpotBugs Coverity sonarqube Xanitizer ... because security matters

[1]https://web.archive.org/web/20201128090742/https://github.com/OWASP/Benchmark/issues/92

Rigorously Evaluating Security Tools

RQ$_1$ — Do security tools and techniques detect the *vulnerabilities that they claim to detect*?

[**S&P'22**, TOPS'20, USENIX'18]

*"**Base**" case/instance*

Example vulnerability (using "DES"):

```
cipher.getInstance("DES");
```

RQ$_2$ — Do the tools detect vulnerabilities *as developers expect them to*?

[**S&P'24**]

Real-world implications of failures of (automated) security analysis

[**USENIX'24**, CCS'22, USENIX'22]

Context: Static Analysis Security Tools (SAST) used for detecting *crypto-API misuse*

CryptoGuard | ShiftLeft | CogniCrypt | QARK | SpotBugs | Coverity | sonarqube | Xanitizer ... because security matters

[1] https://web.archive.org/web/20201128090742/https://github.com/OWASP/Benchmark/issues/92

Rigorously Evaluating Security Tools

RQ$_1$ — Do security tools and techniques detect the *vulnerabilities that they claim to detect*?

[**S&P'22**, TOPS'20, USENIX'18]

*"**Base**" case/instance*

Example vulnerability (using "DES"):

```
cipher.getInstance("DES");
```

```
String var = "DES"
cipher.getInstance(var);

cipher.getInstance("des");

cipher.getInstance
("des".toUpperCase());

…
```

*"**Variants**" of the base case*

RQ$_2$ — Do the tools detect vulnerabilities as developers expect them to?

[S&P'24]

Real-world implications of failures of (automated) security analysis

[**USENIX'24**, CCS'22, USENIX'22]

Context: Static Analysis Security Tools (SAST) used for detecting *crypto-API misuse*

CryptoGuard · ShiftLeft · CogniCrypt · QARK · SpotBugs · Coverity · sonarqube · SANITIZER … because security matters

[1] https://web.archive.org/web/20201128090742/https://github.com/OWASP/Benchmark/issues/92

Rigorously Evaluating Security Tools

RQ$_1$ **— Do security tools and techniques detect the *vulnerabilities that they claim to detect*?**

[**S&P'22**, TOPS'20, USENIX'18]

Example vulnerability (using "DES"):

*"**Base**" case/instance*

```
cipher.getInstance("DES");
```

```
String var = "DES"
cipher.getInstance(var);

cipher.getInstance("des");

cipher.getInstance
("des".toUpperCase());

…
```

*"**Variants**" of the base case*

Existing Evaluation?: *Manually-curated, ad-hoc, benchmarks*, often developed by the tool designers, which contain few, if any, challenging variants, and are sometimes even incorrect

Context: Static Analysis Security Tools (SAST) used for detecting *crypto-API misuse*

CryptoGuard ShiftLeft CogniCrypt QARK SpotBugs Coverity sonarqube XANITIZER … because security matters

[1]https://web.archive.org/web/20201128090742/https://github.com/OWASP/Benchmark/issues/92

Rigorously Evaluating Security Tools

RQ$_1$ — Do security tools and techniques detect the *vulnerabilities that they claim to detect*?

[**S&P'22**, TOPS'20, USENIX'18]

Existing Evaluation?: *Manually-curated, ad-hoc, benchmarks*, often developed by the tool designers, which contain few, if any, challenging variants, and are sometimes even incorrect

ECB mode **considered not vulnerable until March '20** in **OWASP[1]!**

*"**Base**" case/instance*

Example vulnerability (using "DES"):

```
cipher.getInstance("DES");
```

```
String var = "DES"
cipher.getInstance(var);

cipher.getInstance("des");

cipher.getInstance
("des".toUpperCase());

…
```

*"**Variants**" of the base case*

Context: Static Analysis Security Tools (SAST) used for detecting *crypto-API misuse*

CryptoGuard ShiftLeft CogniCrypt QARK SpotBugs Coverity sonarqube Xanitizer … because security matters

[1] https://web.archive.org/web/20201128090742/https://github.com/OWASP/Benchmark/issues/92

Rigorously Evaluating Security Tools

RQ$_1$ — Do security tools and techniques detect the *vulnerabilities that they claim to detect*?

[**S&P'22**, TOPS'20, USENIX'18]

RQ$_2$ — Do the tools detect vulnerabilities as developers expect them?

[S&P'24]

Real-world implications of failures of (automated) security analysis

[**USENIX'24**, CCS'22, USENIX'22]

***"Base"** case/instance*

Example vulnerability (using "DES"):

```
cipher.getInstance("DES");
```

```
String var = "DES"
cipher.getInstance(var);

cipher.getInstance("des");

cipher.getInstance
("des".toUpperCase());

…
```

***"Variants"** of the base case*

ECB mode **considered not vulnerable until March '20** in **OWASP[1]!**

Context: Static Analysis Security Tools (SAST) used for detecting *crypto-API misuse*

CryptoGuard ShiftLeft CogniCrypt QARK SpotBugs Coverity sonarqube SANITIZER … because security matters

[1] https://web.archive.org/web/20201128090742/https://github.com/OWASP/Benchmark/issues/92

Rigorously Evaluating Security Tools

RQ1 — Do security tools and techniques detect the *vulnerabilities that they claim to detect*?

[**S&P'22**, TOPS'20, USENIX'18]

Research Gap: The lack of a *systematic* approach for *rigorously* and *automatically* evaluating security tools; crypto-API misuse detectors (or crypto-detectors) in particular

ECB mode **considered not vulnerable until March '20** in **OWASP[1]!**

***"Base"** case/instance*

Example vulnerability (using "DES"):

```
cipher.getInstance("DES");
```

```
String var = "DES"
cipher.getInstance(var);

cipher.getInstance("des");

cipher.getInstance
("des".toUpperCase());

…
```

***"Variants"** of the base case*

Context: Static Analysis Security Tools (SAST) used for detecting *crypto-API misuse*

CryptoGuard ShiftLeft CogniCrypt QARK SpotBugs Coverity sonarqube SANITIZER … because security matters

[1]https://web.archive.org/web/20201128090742/https://github.com/OWASP/Benchmark/issues/92

Mutation Analysis for evaluating Static Crypto-API misuse detectors (*MASC*)[1]

Contextualize mutation testing to ***systematically*** generate variants, and ***rigorously*** evaluate SASTs

[1] Ami, Amit Seal, Nathan Cooper, Kaushal Kafle, Kevin Moran, Denys Poshyvanyk, and Adwait Nadkarni. "***Why crypto-detectors fail: A systematic evaluation of cryptographic misuse detection techniques.***" In the *2022 IEEE Symposium on Security and Privacy (S&P).*

Mutation Analysis for evaluating Static Crypto-API misuse detectors (*MASC*)[1]

Contextualize mutation testing to ***systematically*** generate variants, and ***rigorously*** evaluate SASTs

[1] Ami, Amit Seal, Nathan Cooper, Kaushal Kafle, Kevin Moran, Denys Poshyvanyk, and Adwait Nadkarni. "***Why crypto-detectors fail: A systematic evaluation of cryptographic misuse detection techniques.***" In the *2022 IEEE Symposium on Security and Privacy (S&P).*

What to mutate?

Crypto API Misuse Taxonomy

105 Crypto API misuse cases

Categorized to 9 Clusters based on *Semantic* Meaning

Covers last 20 years of Study from Industry & Academia

Required Over 2 person months to extract misuse cases

...ent & Server Secrecy (20)

Small Key Size
- Using RSA with < 1024 bit key (7)
- Using RSA with < 2048 bit key (3) +
- Using RSA with 2048 bit private key (1)

Weak Algorithm
- Using RSA with CBC (1)
- Using RS...
- Using R...

Weak Certi...
- Improp...
- Trustin...
- Missing...
- Improp...

Weak Hostname Management
- Allowing all hostnames (10) ✔
- Using D...

Weak SSL P...
- Using w... {SSL...
- Using S...
- Using S...
- Using S...
- HMAC for TLS with SHA1 (1)
- Using CBC for SSL/TLS with AES (1) *
- Using TLS < v 1.2 (1)
- Using TLS < v 1.1 (3)

Compromising Randomness (5)

Misuse of Randomness
- Bad derivation of IV (file/text) (4) ✔
- Low entropy in key generation/ RNG (3)
- Using static seeds for Secure Random RNG (7)
- Not using Secure Pseudo RNG (7)
- Using Setseed (3)

Compromising Non-Repudiation (3)

Key Signing Misuses
- Low entropy with DSA (1)
- Low entropy with ECDSA (1)
- Using 1024 bit DSA (2)

Unclustered (6)

- Inscure pinning with ambiguous values
- Trusting Self-signed Certificates +
- Using unencrypted server socket
- Using unencrypted socket
- Using export quality ciphers
- Using stateless encryption

API/Program Specific Misuses (17)

API/Program Specific
- Apache `HTTPClient` no host verification (1)
- `Gnutls_certificate_verify_peers2` returns 0 when self signed certificate (1)
- Constant password for android keystore (2)
- JSSE checkTrusted method does not check identify if the algorithm field is null or empty string (1) ✔
- ...droid Webview incorrect certificate verification ...
- ...va defaults to `ECB` for encryption with "`AES`"
- ...berknecht does not have host verification (1)
- ...ing `DefaultHttpClient` (due to no TLSv1.2) (1...
- ...oring `onReceivedSSLError` (3)
- ...`LSocketFactory` without verifying Hostname
- ...using counter value in encryption (2)
- Apache `HttpHost` data allows mixed schemes (1)
- Using obsolete algorithm (11) ✔
- Storing sensitive data in Java `String` (3)
- ...ing `Socket` directly for connection (1)
- ... `clearPassword` call after using `PBEKeySpec`
- ...`EKeySpec` initialized without salt (2)

...romising Secret Keys (12)

Secret Key Misuses
- Using low entropy seeds in key generation (1)
- Password Based Key Derivation Function (PBKDF) Using < SHA224 (1)
- Not using Salts while hashing password (1)
- PBKDF Using HMAC (1)
- PBKDF Using MD5 (3)
- PBKDF Using MD2 (2)
- IVs generated w/o random num generator (1) ✔
- Static IV (4) ✔
- Zeroed IV (2)
- Using hardcoded key / password (3)
- Using Constant Encryption Key (9)
- Using < 64bit salt for password (2)

Compromising Communication Secrecy with Intended Receiver (6)

Communication Secrecy Compromised
- Use of a key past its expiration date (1)
- HTTP and HTTPs mixing (3)
- Key Exchange without Entity Authentication (1)
- Improper Check for Certificate Revocation (1) ✔
- Improper Validation of Certificate with Host Mismatch (1) ✔
- Untrusted CA Signed Certificate (1) ✔

Compromising Secrecy of Cipher Text (26)

Insecure Key Size
- ECC < 224 bit (2)
- Using AES with < 128 bit key (1)
- Using RC2 with < 64 bits (1)

Insecure Number of Iterations/Cycles
- Using < 500 iterations for PBE (1)
- ...

...

- ...or
- Using AES with CBC for Encryption * (2)
- Using DESede with ECB (1)
- Using DES with CBC3 SHA (1)
- ...aining

Unsafe Algorithm Usage
- Using RC2 for symmetric encryption (4)
- Using NullCipher to encrypt plain text (1)
- Using Blowfish Algorithm for Encryption (4)
- Using ESAPI Encryptor (1)
- Using 3DES/DESEDE for encryption (4)
- Using RC4 (3)
- Using IDEA Algorithm for Encryption (3)
- Using DES for encryption (8) ✔
- Using EXP1024 for ciphers (1)
- Using Seed Cipher (1)
- Using blowfish with less than 128 bit key (1)

Compromising Integrity through Improper Checksum Use (10)

Compromised Checksums
- Hashing credentials - MD5 (5) ✔
- Hashing Credentials - MD4
- Hashing Credentials - MD2
- Digital Signature Hashes - MD4
- Obsolete Hash Algorithm (7) ✔
- Hashing Credentials - SHA1
- Digital Signature Hashes - MD5 (5) ✔
- Using a custom MessageDigest instead of relying on the SHA-224 (1)
- Digital Signature Hashes - MD2 (4)
- Digital Signature Hashes - SHA1 (5)

* CBC is insecure in TLS/client-server context; + applicable in specific situations; some misuse are newer compared to other in same cluster, # PKCS5 suggestion based

Mutation Analysis for evaluating Static Crypto-API misuse detectors (*MASC*)[1]

Contextualize mutation testing to ***systematically*** generate variants, and ***rigorously*** evaluate SASTs

Research Challenge 1:
What to mutate?

i.e., what vulnerabilities do we want to express?

[1] Ami, Amit Seal, Nathan Cooper, Kaushal Kafle, Kevin Moran, Denys Poshyvanyk, and Adwait Nadkarni. "***Why crypto-detectors fail: A systematic evaluation of cryptographic misuse detection techniques.***" In the *2022 IEEE Symposium on Security and Privacy (S&P).*

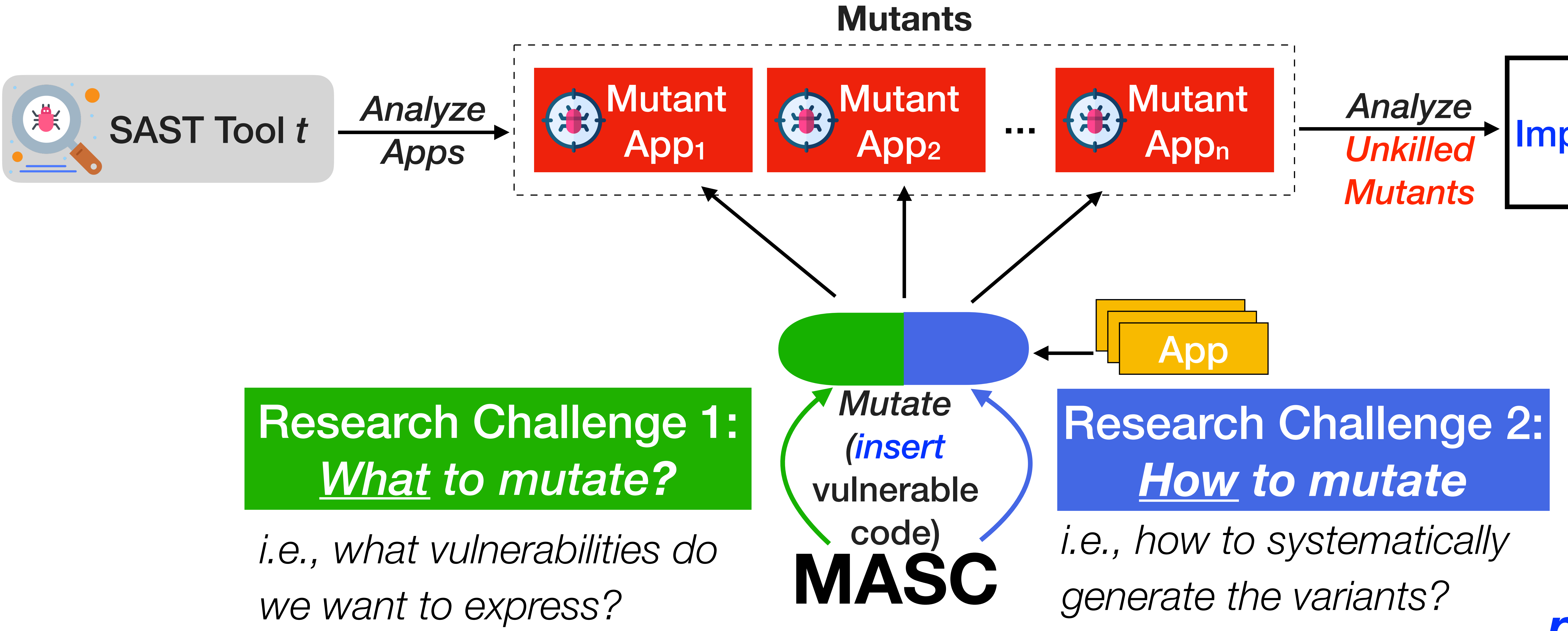
Mutation Analysis for evaluating Static Crypto-API misuse detectors (*MASC*)[1]

Contextualize mutation testing to ***systematically*** generate variants, and ***rigorously*** evaluate SASTs

[1] Ami, Amit Seal, Nathan Cooper, Kaushal Kafle, Kevin Moran, Denys Poshyvanyk, and Adwait Nadkarni. "***Why crypto-detectors fail: A systematic evaluation of cryptographic misuse detection techniques.***" In the *2022 IEEE Symposium on Security and Privacy (S&P).*

Mutation Analysis for evaluating Static Crypto-API misuse detectors (*MASC*)[1]

Contextualize mutation testing to ***systematically*** generate variants, and ***rigorously*** evaluate SASTs



A *threat model* for Crypto-detectors

Threat-aware *mutation primitives: operators* and *scopes*

[1] Ami, Amit Seal, Nathan Cooper, Kaushal Kafle, Kevin Moran, Denys Poshyvanyk, and Adwait Nadkarni. "***Why crypto-detectors fail: A systematic evaluation of cryptographic misuse detection techniques.***" In the *2022 IEEE Symposium on Security and Privacy (S&P).*

Evaluating Crypto-Detectors [S&P 2022]

Evaluating Crypto-Detectors [S&P 2022]

19 Design/Implementation Flaws, 5 Flaw Classes

	Flaw Class (FC)	*# of flaws*	*Description*
FC1	**String Case Mishandling**	**1**	Not detecting an insecure algorithm provided in lower case *e.g., Cipher.getInstance("des");*
FC2	**Incorrect Value Resolution**	**8**	Incorrect resolution of parameters passed to Crypto-APIs, *e.g.,* String alg = "des"; Cipher.getInstance("des");
FC3	**Incorrect Resolution of Complex Inheritance & Anonymous Objects**	**4**	Inability to detect inheritance relationship among classes, *e.g., vulnerable SSL verification in anonymous inner class objects of X509ExtendedTrustManager*
FC4	**Insufficient Analysis of Generic Conditions**	**3**	Inability to identify fake/unrealistic conditions within overridden methods, *e.g., if(true ll session == null) return true; return false;*
FC5	**Insufficient Analysis of Context-Specific Conditions**	**3**	Inability to identify context-specific, fake/unrealistic conditions within overridden methods, e.g., *if(true ll session.getCipherSuite().length()>=0) return true; return false;*

Flaw Class “0” (Incomplete Analysis)

FC0 - 1. Ignore any class with “`.android`” in fully qualified name

Why? To ignore *Android libraries*

Implications: Ignores critical Android apps *and 10% of its evaluation dataset.*

- com.twitter.**android**
- com.lastpass.**lpandroid**
- com.google.**android**.gm

FC0 - 2. Not handling “`multidex`”

Android Studio automatically uses multidex for >Android 5.0

Implications: Does not completely analyze apps built after *2014, and 63% of its evaluation dataset*

- Part of the Eclipse IDE for several years
- Integrated into Oracle’s internal testing suite
- Funded by millions of federal $

19 Design/Implementation Flaws

FC1 (1 flaw) - String Case Mishandling

```
cipher.getInstance("DES");
```
DETECTS

```
cipher.getInstance("des");
```
DOES NOT DETECT

FC2 (8 flaws) - Incorrect Value Resolution

```
Cipher.getInstance("AES");
```
DETECTS

```
String alg = "AES";
Cipher.getInstance(alg);
```
DOES NOT DETECT

Industry leading tool, used in *7.5k+ open source projects*, and by *38k +developers*

```
Cipher.getInstance(
    (obj.A().B().getValue()
);
```
DOES NOT DETECT

flaw in 7/9 detectors

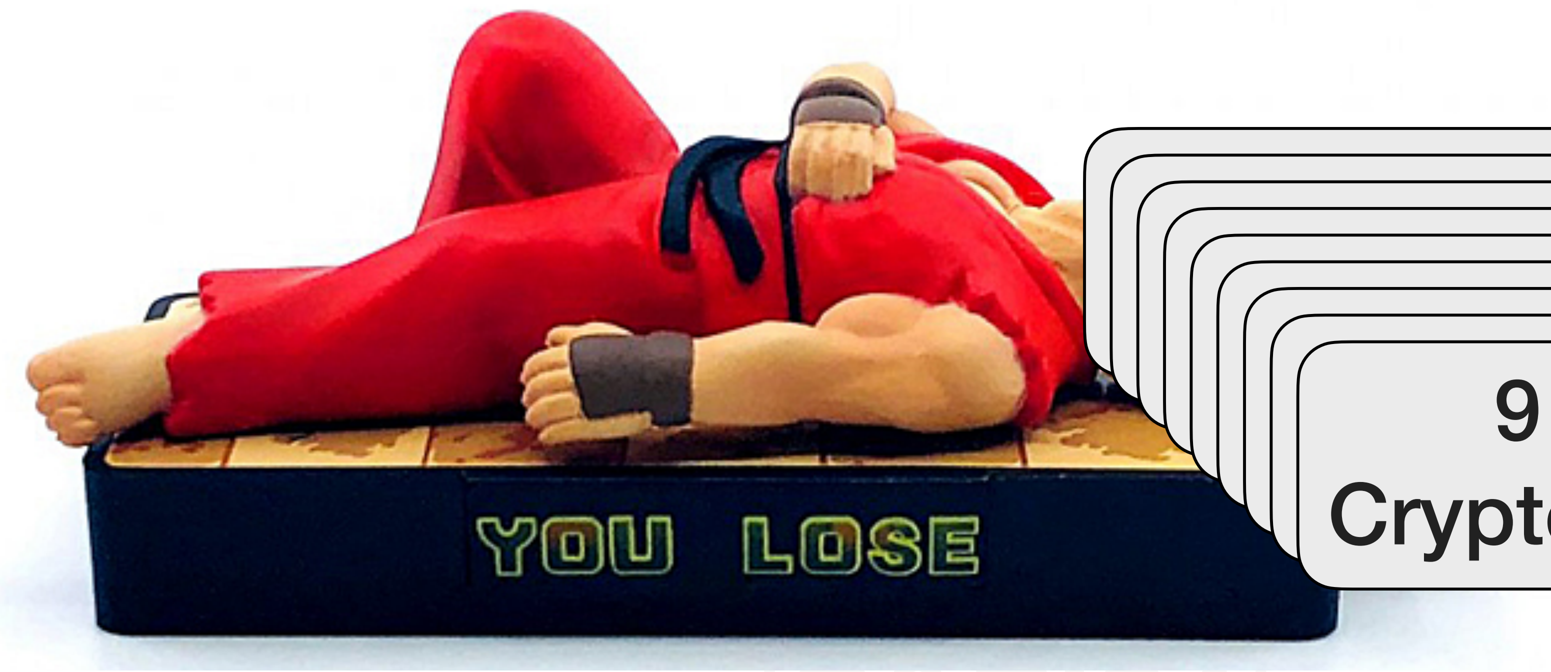
Repository	Stars (GitHub)
Apache Druid	10.3K stars
Exoflayer (Google)	16.8K stars
UltimateAndroid	2.1 K stars
JeeSuite	570 stars
Apache Ignite	3.5K stars

- ***Similar misuse instances were found during our impact study***

Do Security Tools Work?

RQ$_1$ — Do security tools and techniques detect the *vulnerabilities that they claim to detect*?

[**S&P'22**, USENIX'18, TOPS'20]



9 Popular Crypto-Detectors!

[S&P'24]

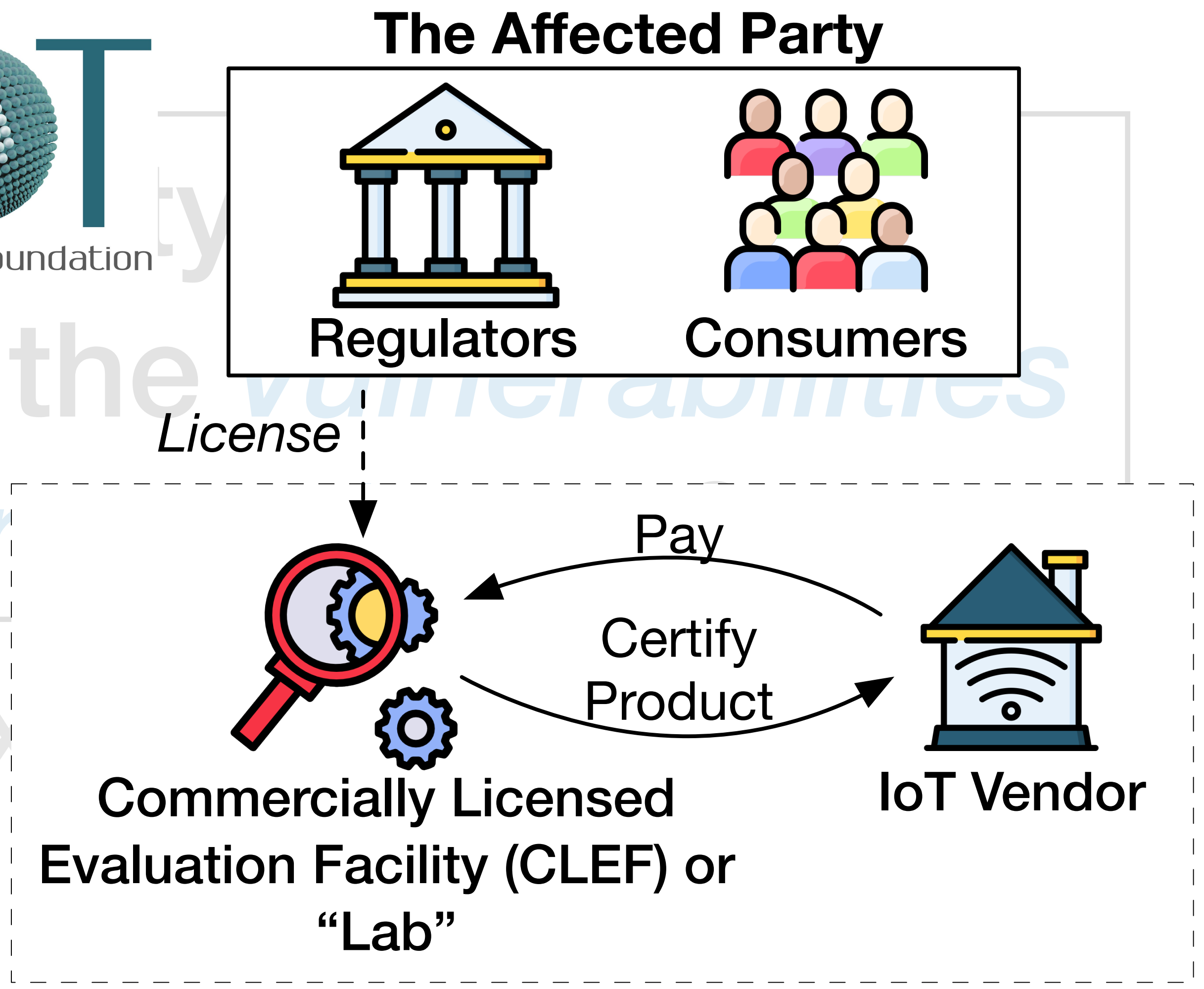
Real-world implications of failures of (automated) security analysis

[**USENIX'24**, CCS'22, USENIX'22]

Context: Static Analysis Security Tools (SAST) used for detecting *crypto-API misuse*

CryptoGuard, ShiftLeft, CogniCrypt, QARK, SpotBugs, Coverity, sonarqube, XANITIZER ... because security matters

Investigating Early Artifacts from IoT Compliance Certification[1]



Preliminary observations from 30 CLEFs: 1) 25/30 certify mobile-IoT apps, **2)** 21/26 use SASTs (unclear for 4/30)

Real-world implications **of failures of (automated) security analysis**

[**USENIX'24**, CCS'22, USENIX'22]

[1] Mandal, Prianka, Amit Seal Ami, Victor Olaiya, Sayyed Hadi Razmjo, and Adwait Nadkarni. "***'Belt and suspenders' or 'just red tape'?*: Investigating Early Artifacts and User Perceptions of IoT App Security Certification.**" In *Proceedings of the 2024 USENIX Security Symposium (USENIX).* (To Appear)

Findings: Security Analysis of Certified Apps

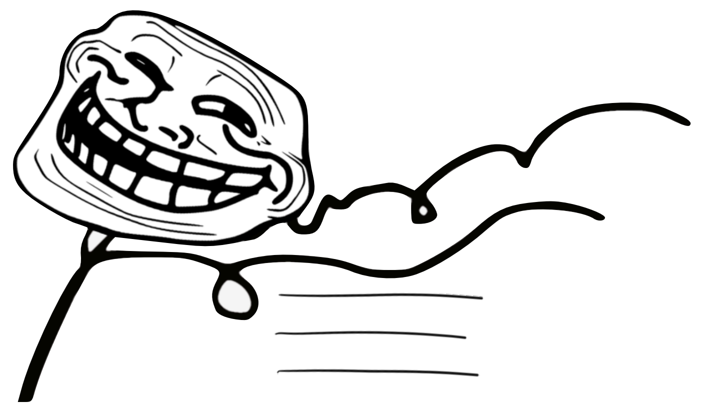
35 crypto-API vulns in 9/11 *certified* apps from ioXt internet of secure things

```
// The string operations below result in: "AES /" + "E" + "C" + "B" + "/NoPadding"
// = "AES/ECB/NoPadding"
```

```
this. ALGO = "AES/" +
            (( char) ("AES/GCM/NoPadding". charAt (4) - 2) ) +
            "AES/GCM/NoPadding". charAt (5) +
            (( char) ("AES/GCM/NoPadding". charAt (6) - 11) ) +
            "/NoPadding";

Cipher cipher = Cipher . getInstance (this. ALGO );
```

T_3: Evasive Developer



An example: Vulnerable Code in an IoT SDK/platform; used by 580k developers (app installed by >5 million users)

Security tools ***did not work!***

Finding 5: CogniCrypt, MobSF, and CryptoGuard, do not detect several of the 35 critical vulnerabilities discovered using manual reverse engineering, i.e., 33/35, 28/35, and 15/35 respectively, and ***none detect the evasive use.***

Why do Crypto-detectors fail? — Perspectives of Tool Designers [S&P 2022]

Security-Centric Evaluation *VS* *Technique-Centric* Design

*"When you really want a protection property to hold, it's vital that the design and implementation be subjected to **hostile review"***

- Security Engineering: A Guide to Building Dependable Distributed System

*"(Tools / approaches) seen so far were <u>technically</u> <u>motivated</u> - **not use-case motivated**.. (e.g.,) should we use alias analysis?"*

- CryptoGuard

*Seemingly-Unlikely/**Evasive flaws** are **within scope** as long as they are **found in the wild***

- CryptoGuard, CogniCrypt

*And should be **frequently observed**!*

- Github Code Scan/LGTM

*"the distinction should not be between 'common' and 'uncommon', but instead between 'can be (easily) **computed statically' and 'can not be computed'**."*.

- Xanitizer

Takeaway: *We need to arrive at a consensus regarding scope (specification grand challenge)*

SASTs prioritize lower FPs *for developers*, but…

Developers would **tolerate high FPs** if the tools ***find something of value***; FNs scare them the most [**S&P 2024**]

Finding 10: Nearly all the practitioners expressed a preference for fewer false negatives, i.e., *as long as the SAST is able to find valid security vulnerabilities, they would tolerate and even prefer few false negatives at the cost of many false positives*

"***I'd rather my security tool be annoying and tell me about every single possible issue*** *over it not telling me anything and* ***just letting <vulnerabilities> slide through.***" *- P14, Law Enforcement*

"*I* ***wouldn't mind wading through 100 false positives,*** *if I thought there were actually going to be genuine issues" - P02, OSS - Java App Server*

"False negative for sure. I just told you the amount of the price of the bug (in millions), so ***I don't care if there are 10 false positives. False negative - that one is going to kill you.***" *- P04, Automobile Sensors*

Re-think How We Design Security Tools

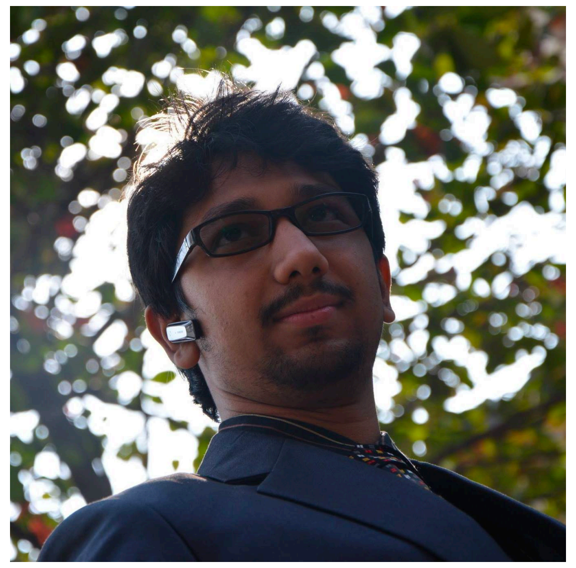
Re-think How We Design Security Tools

- Understand of *how crypto-APIs are actually misused in the wild*
- Focus on *finding something of value,* i.e., *what developers want*

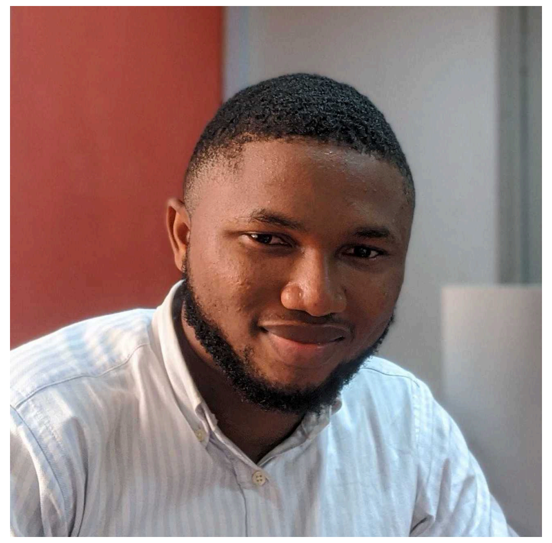
Thank You!

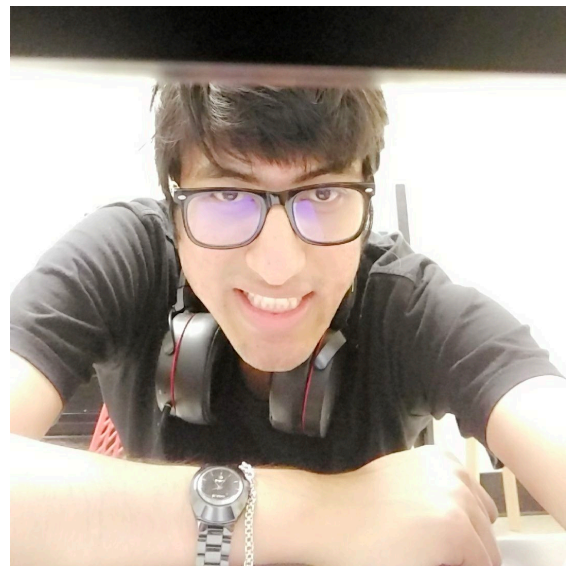
Students


Kaushal Kafle


Amit Seal Ami


Prianka Mandal


Victor Olaiya

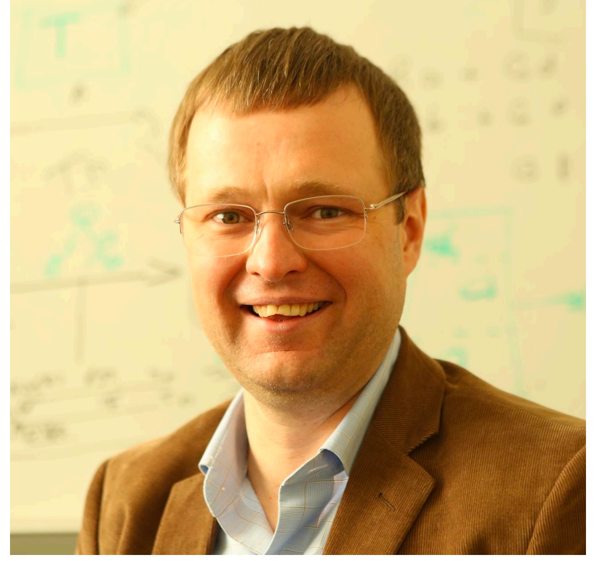

Sunil Manandhar
(PhD'2023, now at IBM Research)

Adwait Nadkarni
apnadkarni@wm.edu

Secure Platforms Lab

WILLIAM & MARY
CHARTERED 1693

Collaborators


Denys Poshyvanyk,
Professor, W&M


Kevin Moran,
Assistant Prof., UCF

Undergrad/MS


S. Hadi Razmjo
(BS CS, Spring'23)

https://spl-wm.github.io/

https://github.com/Secure-Platforms-Lab-W-M/masc-artifact