# Detecting SSL-related security vulnerabilities in Android applications

Graham Edgecombe

# SSL certificate validation can go wrong

- I am example.com, signed by Symantec

- I am example.com, signed by Mallory

- I am other-site.com, signed by Symantec

- I am other-site.com, signed by Mallory

# SSL certificate validation can go wrong

- I am example.com, signed by Symantec

- I am example.com, signed by Mallory

- I am other-site.com, signed by Symantec

- I am other-site.com, signed by Mallory

# SSL certificate validation can go wrong

- I am example.com, signed by Symantec

- I am example.com, signed by Mallory

- I am other-site.com, signed by Symantec

- I am other-site.com, signed by Mallory

# SSL certificate validation can go wrong

- I am example.com, signed by Symantec

✔

- I am example.com, signed by Mallory

✔

- I am other-site.com, signed by Symantec

✔

- I am other-site.com, signed by Mallory

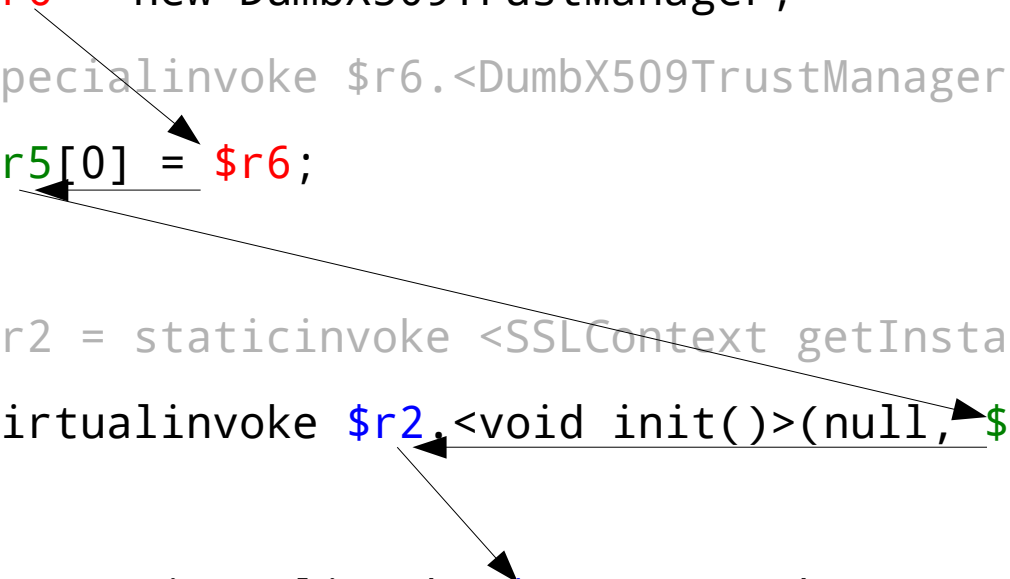
✔

# Static analysis can detect vulnerable certificate validation code

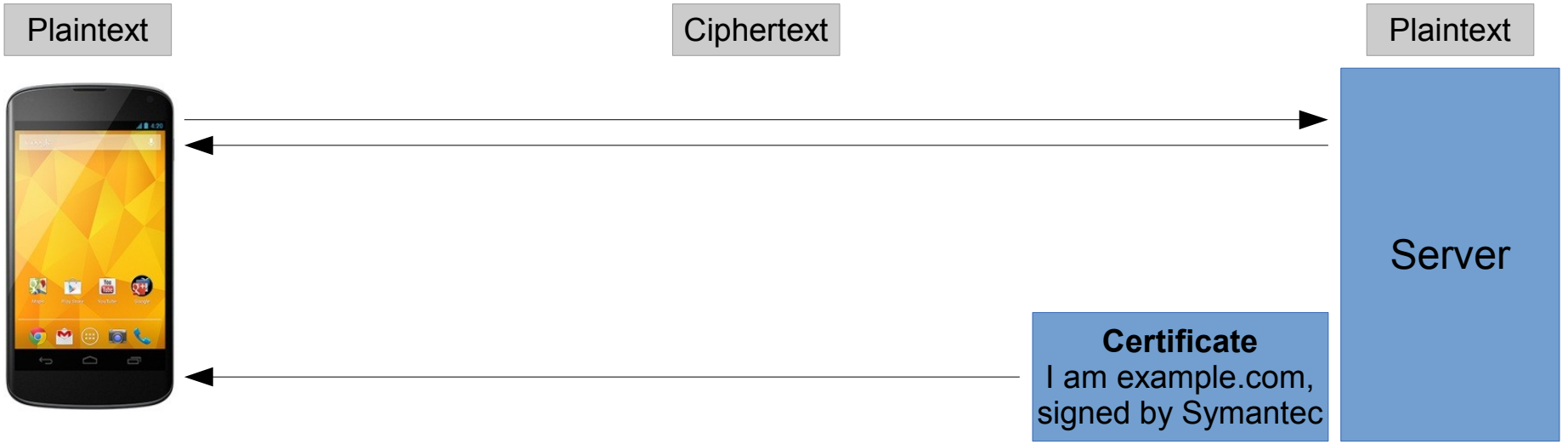
```
$r5 = newarray (TrustManager)[1];

$r6 = new DumbX509TrustManager;
specialinvoke $r6.<DumbX509TrustManager: void <init>()>();
$r5[0] = $r6;

$r2 = staticinvoke <SSLContext getInstance()>("TLS");
virtualinvoke $r2.<void init()>(null, $r5, null);

$r4 = virtualinvoke $r2.<SSLSocketFactory getSocketFactory()>();
virtualinvoke $r4.<Socket createSocket()>("www.example.com", 443);
```

# Static analysis can detect vulnerable certificate validation code

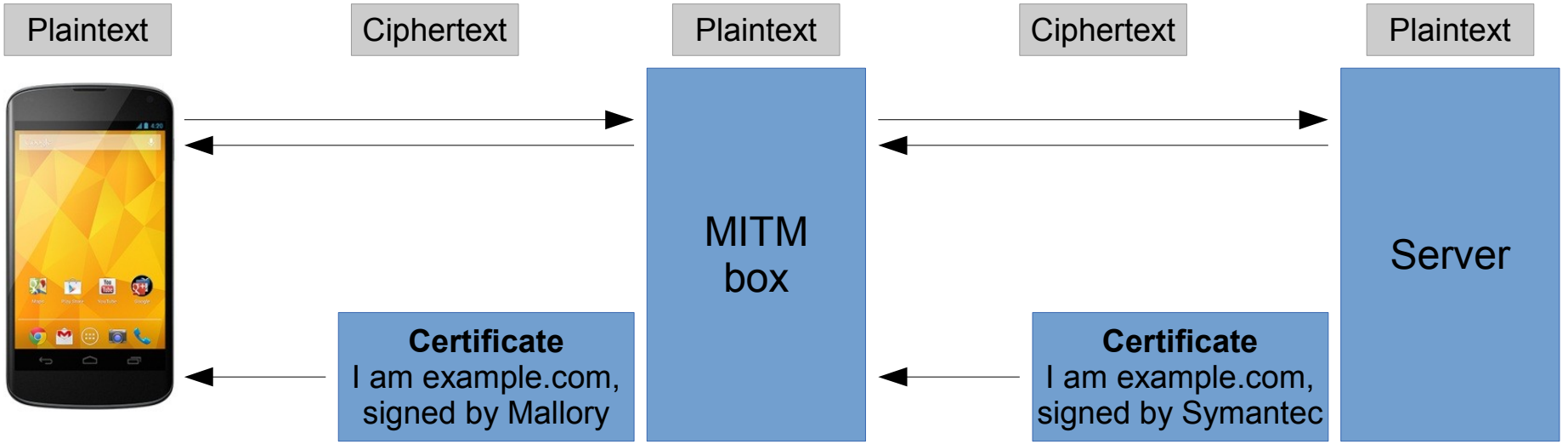
```
$r5 = newarray (TrustManager)[1];

$r6 = new DumbX509TrustManager;
specialinvoke $r6.<DumbX509TrustManager: void <init>()>();
$r5[0] = $r6;

$r2 = staticinvoke <SSLContext getInstance()>("TLS");
virtualinvoke $r2.<void init()>(null, $r5, null);

$r4 = virtualinvoke $r2.<SSLSocketFactory getSocketFactory()>();
virtualinvoke $r4.<Socket createSocket()>("www.example.com", 443);
```



# Man-in-the-middle attacks can exploit incorrect certificate validation



# Man-in-the-middle attacks can exploit incorrect certificate validation



# Results

- Static analysis
  - Tested on 177 apps from the Google Play store
  - Successful for 80% (unstable version of Soot)
  - 58% potentially vulnerable
- Man-in-the-middle
  - Tested on 8 apps so far
  - Can intercept username/password from 2 popular apps (1->5 million installs, 100k->500k installs)