Electronic Evidence

Professor Steven Murdoch

University College London

December 2023

Reliability of electronic evidence depends on whether the computers involved worked correctly

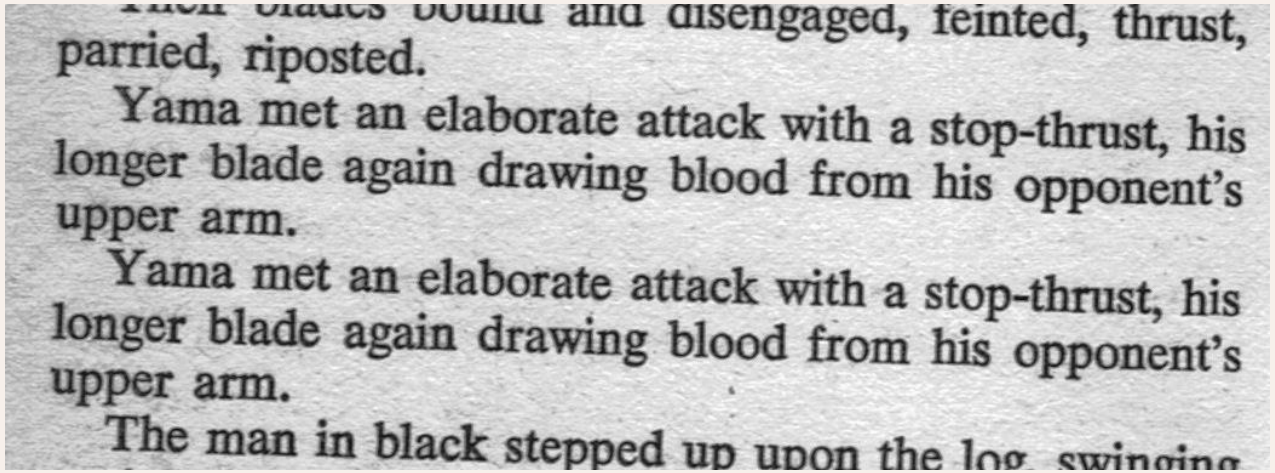
- Computers record what has happened and show this information
  - For example, at a particular time money was withdrawn from an ATM
- Computers are designed to enforce certain rules
  - For example, money can be withdrawn only with the correct card and PIN
- From these we can infer what has happened
  - If money was withdrawn, then the correct card and PIN were used
- Computers do fail to record accurate information and do fail to enforce the rules they are supposed to
  - Software is written by humans, and humans make mistakes: these mistakes are informally called *bugs* and, when triggered, bugs cause a *failure*

In Apple's macOS and iOS a bug was added in 2013 and remained undetected for 4 months

```
if ((err = SSLHashSHA1.update(&hashCtx, &clientRandom)) != 0)
    goto fail;
if ((err = SSLHashSHA1.update(&hashCtx, &serverRandom)) != 0)
    goto fail;
if ((err = SSLHashSHA1.update(&hashCtx, &signedParams)) != 0)
    goto fail;
    goto fail;
if ((err = SSLHashSHA1.final(&hashCtx, &hashOut)) != 0)
    goto fail;

err = sslRawVerify(…);

fail:
    …
    return err;
```


parried, riposted.

Yama met an elaborate attack with a stop-thrust, his longer blade again drawing blood from his opponent's upper arm.

Yama met an elaborate attack with a stop-thrust, his longer blade again drawing blood from his opponent's upper arm.

The man in black stepped up upon the log, swinging

In Apple's macOS and iOS a bug was added in 2013 and remained undetected for 4 months

```
Try to read first part of the data; if there's a problem
    go to fail
Try to read second part of the data; if there's a problem
    go to fail
Try to read third part of the data; if there's a problem
    go to fail
    go to fail
Try to combine all parts read so far; if there's a problem
    go to fail;

Check for security problems in the data

fail:
    …
    Report if there's a problem with the last thing done
```

1) Problem with 2nd part ☑

In Apple's macOS and iOS a bug was added in 2013 and remained undetected for 4 months

```
Try to read first part of the data; if there's a problem
    go to fail
Try to read second part of the data; if there's a problem
    go to fail
Try to read third part of the data; if there's a problem
    go to fail
    go to fail
Try to combine all parts read so far; if there's a problem
    go to fail;

Check for security problems in the data

fail:
    …
    Report if there's a problem with the last thing done
```

1) Problem with 2nd part ✓
2) Security problem ✗
3) Everything's OK ✓

A software failure can be caused by a bug in any piece of software or hardware it relies on

Mobile banking app

iOS operating system

iPhone processors

A software failure can be caused by a bug in any piece of software or hardware it relies on

Mobile banking app	Bank (and others)
iOS operating system	Apple (and others)
iPhone processors	ARM, Qualcomm, …

A software failure can be caused by a bug in any piece of software or hardware it relies on

Mobile banking app	50 thousand lines
iOS operating system	12 million lines
iPhone processors	1 million lines

* very approximate

The Apollo guidance computer project created around 600,000 lines of code

Margaret Hamilton standing next to the 600,000 lines of code she and the team she oversaw developed for Apollo between 1961 and 1969, at the Software Engineering Division of the MIT Instrumentation Laboratory (later the Draper Laboratory)

A typical mobile phone application will rely on over 13 million lines of code

- Typically, software will have between 1 to 25 bugs per 1,000 lines of code
- There will be something like 10,000 to 300,000 bugs that could affect a typical mobile app
- Software is tested by simulating a large number of scenarios and checking for failures, but this will not be comprehensive
- Once software is in use, failures should be investigated to identify and fix further bugs

There will be many bugs, but where electronic evidence is in dispute, did these bugs affect it?

- On the 1997 recommendation of the Law Commission, computers are treated in the same way as mechanical devices and presumed to be operating correctly, but computers are different from machines
- Failures don’t happen at random but are triggered by a particular scenario so could affect some individuals frequently but not others
- Someone malicious could induce the scenario where a bug will cause a failure, even if this scenario is unlikely to occur by chance
- If a failure is claimed to have occurred, it is often unclear whether it was due to a bug, as opposed to other human error or dishonesty
  - Distinguishing the different possibilities can take significant effort

Records kept by any well-run software project can show how reliable the software is

- There are established software-engineering processes for how systems should be developed, maintained and operated
  - Maintaining a record of bugs which have been identified, failures of the software, their effects, and how these were responded to
  - Performing regular audits/tests and properly acting on results
  - Ensuring that software is not modified without proper controls and evidence produced by the software has not been tampered with
- If records show that the software has not been managed well, the number of bugs is high, or there are failures which could explain the matter in dispute, more detailed investigation is justified

Software can be designed to produce evidence that is more reliable and easier to interpret

- Software is often designed without taking into consideration that it might be required to produce evidence that would be presented in court, so facilities to do so are often inadequate for the purpose
- There is growing realisation that change is needed
- Ongoing research in computer science is developing techniques to limit how different pieces of software can interact with each other, to increase reliability and make it easier to reason about software
- Software designed without such techniques will nevertheless be around for many decades so the legal system will have to find ways to deal with their limitations