Security Engineering Specimen Paper: Solution Notes

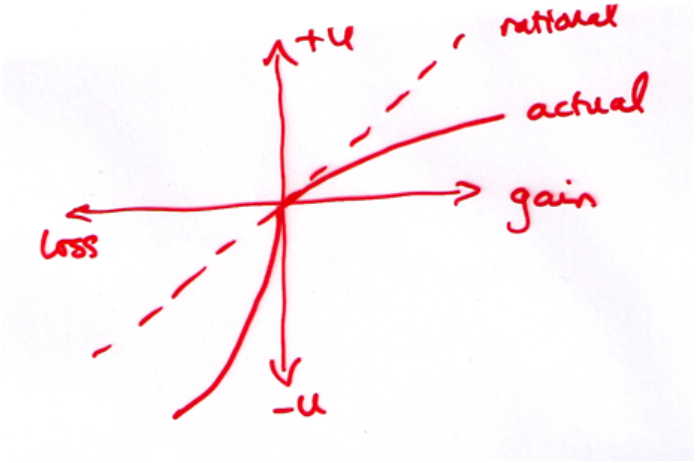
Sam Ainsworth and Ross Anderson

Abstract

Here are notes on one possible set of solutions to our specimen paper. We wouldn't always expect all of these points in every answer, and many other answers would receive marks as long as they were well reasoned and justified. The level of detail may also be expected to be lower and/or higher than given here, depending on the marks allocated to the question.

1

a) Explain Prospect Theory as it applies to decision making. [6 marks]



Expect a graph to be drawn, plotting loss/gain on the x axis, and utility on the y axis, passing through (0,0) and with a concave gain and (steeper) convex loss.

This graph shows that people are risk-averse on a positive frame and risk-seeking on a negative frame: so a scammer (or legitimate business operator) can manipulate users' behaviour in the chosen direction via framing.

b) You are the operator of a scam designed to steal passwords for a popular online shopping website. How would you:

i) utilise attitudes to risk to maximise your earnings potential, [4 marks]

To make people accept more risk, you can put them into a negative frame, by making it appear they will make a loss unless they follow your scam's advice (in this example, to enter their password in a fake website). Examples include sending an email that claims they will lose access to their account and offering a login link. You might even tempt them to do something illegal in order to get some benefit, so they are less likely to report being scammed.

ii) extract value from the scam, [4 marks]

For the scam to be worth doing, there needs to be a method of extracting value. Gift cards are a common channel: they can be sold on to unsuspecting victims or exchanged for cash. Physical goods are another option, but are less convenient.

iii) how would you expect the operator of the shopping website to respond? [4 marks]

The value of the cards depends on the shop's response: they may be able to blacklist them, but it depends on their own incentives (such as whether the gift cards were their own, or issued by a third party), and the timescales (if the fraud is only discovered months later, the cards will have already been used). Although all banks use 2FA, shops almost never do (especially without an opt-in); with banks, regulation takes precedence while for shops it's ease of use. A large shopping website might simply try to ignore the fraud, externalise it by using a third-party gift card, or refund customers if it becomes a PR issue.

c) Hal Varian described the software industry business model as being "bargains then ripoffs": initial low-cost enticement, followed by lock-in. Describe how the software lifecycle impacts security as well as cost. [7 marks]

"We'll ship it Tuesday and get it right by version 3", the Microsoft development strategy in the 1990s and 2000s, is economically rational, as whoever gets the network effects going first is likely to win a market race. So we expect security, and anything else that is an impediment to fast development and adoption, to be a lower priority until a product or service becomes the market incumbent.

Thereafter, incumbent companies are able to play liability games with both users and regulators, and may try to pass more costs on to end users (see example on gift cards in part b), who are less likely to be able to mitigate attacks. When customers are locked in, service degradation – including poor security – is an expected aspect of 'bargains then ripoffs'. This happens in professional service markets too. The SolarWinds hack is an example; SolarWinds was a near monopoly treated as a cash cow by its owners, so it did not receive enough security investment. Large tech companies whose products enable attackers to get at their many customers are also prime targets.

2

You are working as a consultant for a small social-network company, who are worried about being fined for a data breach of their systems, but have the express stated goal of keeping costs down. To do this, the company suggests it wishes to use the services of a cloud operator hosted in the US.

a) What attacks are made easier by the move to a cloud operator? [8 marks]

- Opportunistic exploitation of mistakes caused by your devs being unfamiliar with the complex access controls involved in many cloud systems. A common example is leaving millions of customer records in a world-readable bucket. More complex examples have to do with authenticating users to VMs and VMs to each other.
- Access by state actors: The US intelligence community is legally able to get warrants to access the data of any non-US person regardless of whether there is probable cause against them, so data kept in the USA cannot be protected against IC snooping. The EU requires residents' personal information to be stored in the EU or in countries with equivalent protection so that people can enforce their rights under privacy laws including the GDPR. This has led to regular legal challenges to the methods used to store EU personal data in US cloud services. There are now similar concerns with the UK because of the Investigatory Powers Act; since Brexit an EU resident cannot rely on the European Court of Justice to prevent access to their data if held in UK data centres. A further example, is the move of Chinese iPhone users' data to data centres in China, which means that the Chinese authorities can attach monitoring equipment of their choice to the machines on which it's kept, so the protocols to encrypt it inspire less confidence.
- Multi-tenancy: other actors, including competitors and criminals, may use the same servers as you, especially if using non-private cloud services to lower costs, and so they may be able to access and leak your data whether by using exploits, side channels or your mistakes.
- Remote Access: when significant company assets are held in a remote cloud, remote-access attacks may become easier.
- Virtualization in the cloud adds another level of trusted code base, increasing the attack surface available to capable opponents.

b) What attacks are made harder by the move to a cloud operator? [8 marks]

- Security competence: the security budget of the cloud provider is likely much higher than your own, and so protection mechanisms are likely to be more robust. A cloud provider will employ dozens to hundreds of security professionals to run DDoS prevention, network monitoring, logging, threat hunting, intrusion detection and recovery mechanisms, on a scale and with a sophistication with which most SMEs cannot compete.
- Systems administration: the cost of first-class sysadmins can be amortised over many customers resulting in more professional service and fewer errors, particularly at the level of infrastructure. Much of the admin is automated within containers, so software updates more likely to be shipped, and the configuration itself may be bought from the cloud operator (especially in SaaS and PaaS). Automatic backup services provide more resilience against accidental data loss.

- Architecture: Your entire system must be designed to frustrate attack from co-located processes and network devices, and so attacks from (mis-)trusted devices within the protection domain may end up less likely.
- Hiding in the crowd: the precise location of your data may be more difficult for an attacker to work out.
- Automatic Validation Services: Compliance and resilience may be a core offering of the cloud service; for example, it may implement the GDPR 'right to be forgotten' as a service.

c) The company has listened to your concerns in part a, and suggests that, using a variety of the latest technological mechanisms, these can be effectively mitigated. What such mechanisms may be relevant, and how might the system still be attacked even with the latest technology? [9 marks]

Secure Enclaves, such as Intel SGX, enable data to be processed in an enclave that makes it resistant to observation by the cloud service provider, governments, and rival tenants. Still, attacks such as side channels are officially considered "out of scope", which means that both timing attacks on non-hardened cryptography, as well as transient execution attacks such as Spectre and Foreshadow, are possible vectors. There are also issues around attacks such as Plundervolt, which may limit the guarantees around privacy from the data centre operator. See above remarks on Chinese data centres: capable opponents with physical access will usually get in.

Still, the company's main goal may not be blocking state actors but the avoidance of liability. Following industry-standard practice is likely to achieve this in many cases.

There are also the questions of whether a) your company will be incentivised not to use enclave support, due to performance loss, and b) whether the data centre operator may be incentivised to forgo Enclave support for the same reason (thus breaking Remote Attestation), as seen with AMD's SEV implementation. There are also questions over which other parts of the system may be compromised (supply chain attacks), and a fundamental trust of the processor provider, including legally mandated backdoors.

There is also the extent to which hardware assurance may be a misdirection, when attacks are more likely to be successful as a result of misconfiguration or software bugs that can be exploited (even in an enclave), such as API attacks. And the most common attack is still that developers get complacent or confused at all the security features available from their cloud provider, and leave sensitive data world-readable where opportunists pick it up and publish it.

3

(a) Access control is nominally a triple of (user, program, file). Explain to what extent this manifests in

i. Windows [4 marks]

Windows misses the "program" part of the triple: access control is implemented as an access control list, with every program running as a given user able to access any of that user's files with that user's own permissions. This manifests as a lack of isolation between applications that may not always be trustworthy.

Some minor exceptions exist: "apps" (the things that come from the Microsoft Store you've probably never used, rather than generic Windows binaries) have their own permissions which work a bit like the "program" part of the triple, sandboxes can decide for themselves to limit permissions of code running inside them (e.g. your Java Virtual Machine might decide any code running inside it should be banned from accessing the file system, and thus limit itself), and Mandatory Integrity Control is used to stop things downloaded from the browser (Low) from altering anything with integrity level (Medium, High), at least without an elevation.

ii. Android [4 marks]

Android implements (program, file) instead – each application runs as a separate user via setuid, with permissions granted via adding the program's UID to different groups. In addition, this is backed up via Mandatory Access Control, via SELinux. Both give significantly more isolation between apps that are a) mutually distrusting, and b) often at least semi-malicious even when legitimate – and after all, most phones only have one user anyway.

(Technically, Android uses the Linux kernel, so all of the below *could* be used, but isn't in practice).

iii. Linux [6 marks]

A typical Linux/Unix setup implements the full (user,program,file), but in a convoluted way. Files (of which application binaries are a special case) have user/group/world permissions (of which only one each can be set, unlike the literal ACL in Windows) of Read/Write/Execute – that gives the (user,file) pair. The final part of the triple comes from setuid/gid, which are bits that can be set on applications (as files) that cause the application to be run as the relevant user or group, and inherit their permissions. That means e.g. your printing application could run with a "printer" setuid, where the "printer" user is a member of the group able to access peripherals.

You may also have SELinux, which is a MAC layer separate to the discretionary access control above (and allows arbitrary policies based on Domain-Type Enforcement), but this is only described in brief detail in the course. And there are extensions to some Linux filesystems that allow arbitrary access control lists (rather than the user/group/world limitation), but these don't really change capability (only simplicity of expression).

(b) "*If a process running at root-level privilege is compromised, the security of an Android phone is entirely defeated.*" To what extent is this statement correct? [5 marks]

This is about the parts of SELinux that are covered in the course, namely for Android. The point is that MAC protects kernel applications from having extreme privilege as well as userspace – so even if a process running at root is compromised, the MAC layer might stop you from generating e.g. a root shell to gain arbitrary privilege escalation. The details of various kernel exploits, and how SELinux

could have prevented their use, are given in Section 4.1 of *Security Enhanced (SE) Android: Bringing Flexible MAC to Android*, Smalley and Craig, NDSS 2013 (`https://www.ndss-symposium.org/ndss2013/ndss-2013-programme/security-enhanced-se-android-bringing-flexible-mac-android/`).

Another interpretation is along the lines of "security for who" – a phone with root-level access can still isolate mutually distrusting apps from each other – it just allows the end user to have full control over the device. That level of access is assumed for e.g. a personal Windows laptop, so it may be a funny assumption to make as regards to the security being "entirely defeated."

Yet another (slightly more technical, less covered in the course) is about the baseband – which TrustZone was initially developed to protect (so that phones could less easily be turned into devices to meddle with cellular networks). This is within-enclave, where even a root-level actor should not have access (save for side channels, and other attacks on enclave integrity/confidentiality such as Plundervolt and various others in the course).

(c) Explain how a user may gain "root" access to an Android device, even without a software bug in the kernel. [6 marks]

There are various possible answers to this. One is implied by part b) – that an application with setuid root is compromised instead (to give you a root shell), though SELinux in modern Android will attempt to mitigate where feasible. Another is that the device might just have root-level access already – after all, not giving this to users is a choice by the manufacturer (recall DAC vs MAC from the OS Security Lecture: "*Alternate view (Android): DAC requires permission of the user. MAC requires consent of user, developer AND platform – 3-party consent.*") It is uncommon for phones to have end-user root access by default, but more common on other devices e.g. set-top boxes aimed at the techie market.

Yet another involves a social-engineering route to gaining root. For example, the key to sign OS updates for a given phone might be leaked from the company that designed the device (through coercion, insiders, or poor within-organisation security). That would allow an end-user to replace the OS with a "rooted" version.

Finally, another option is to attack the hardware instead of the software. Differential fault analysis is becoming increasingly popular as a way to leak root keys (`https://wololo.net/2021/11/24/some-fuel-for-the-switch-scene-writeups-on-tsec-tegra-security-processor-exploits/`). Differential power analysis has been used on lightbulbs (Philips Hue example in the hardware security lecture). Rowhammer has been used to root Android phones by flipping permissions bits (`https://arstechnica.com/information-technology/2016/10/using-rowhammer-bitflips-to-root-android-phones-is-now-a-thing/`). And more recent bugs such as Spectre/Meltdown (especially before the latter was patched) could allow root-key leakage under certain circumstances (which are particularly vague, involving a code path with a vulnerable gadget and no barriers in the kernel, for example – so while we haven't seen these yet, assurance isn't great).