Security Behaviours

COMS30038 Lecture 10 - Security Ergonomics

Dr Ramokapane

University of BRISTOL

Bristol Cyber Security Group

A Quick Recap So Far

L1 - Humans-in-the-loop

Humans are limited by their **capacity**, **subject knowledge** & **motivation**. Sometimes they just get things wrong.

L4 - Error in Practice

Humans will **always err.** Deliberately or inadvertently it can't really be avoided. But **sometimes we can be heroes**.

L2 - Usable & Inclusive Security

We need to start **fitting tasks to the human**, building systems that are not only **usable**, but also **acceptable and consider** users' **context**.

L5 - Biases

There are **cognitive & physiological underpinnings** as to why humans err.

A (Software Engineering) Problem Space

Security work has often focussed upon technical advancement - the build a better mousetrap approach.

As we've seen, humans - through error or mistake (intentional or otherwise) - are often seen as at fault or to blame for security problems with a culture-of-blame leading "to a misconception that better security comes from better systems." i.e., it's a self fulfilling prophecy.

"The more effort placed into better smarter technology the more likely it is that, in the event of failure, the human is seen as in error and therefore further effort needing to be put into further technological improvement."

But safety research demonstrates that technical improvement can sometimes cater for human (cognitive) capacity there is a point at which "automation... simply shifts the error" – normally to the designer / developer."

What Can We Do?

1. To **cater** for the very real **constraints of humans**, accepting that they are and will likely always be part of the loop

2. **Fit** security related **tasks**, applications and devices **to the human** making them usable

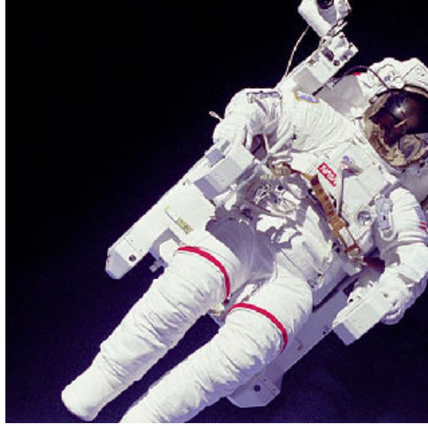
3. Design for **resilience to human error**, accepting it will happen, AND

4. **Avoid that shift in error to ourselves**, and overcome our own fallibility
   - and in turn **avoid baking in latent failures** to the things we build

From its inception **Security Ergonomics by Design** was created to help identify areas of systems at risk, from human active error or latent failure, and to design these out of… systems.



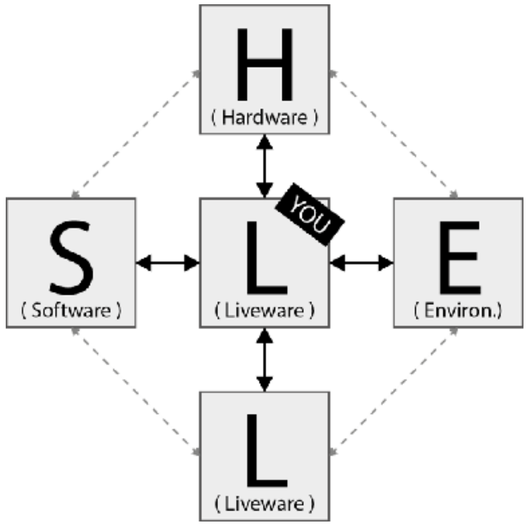




Grounded in Human Factors / Ergonomics

Simply… the aviation industry along with several others concerned with safety have long recognised that the human is often not the entire sole or root cause of failure. Moreover, HF/E looks to…

"the interactions among **humans** and **other elements** of a system… applying theory, principles, data and methods to design in order to optimise human well-being AND overall system performance" *IEA*



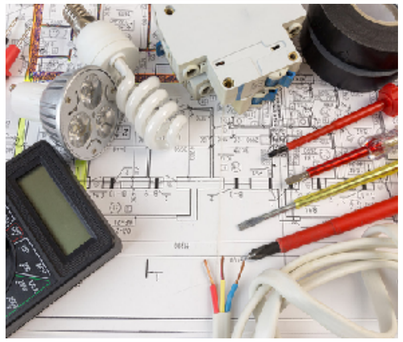
Useful Stuff

Hawkins, F.H. 1987
Human Factors in Flight.
Gower Technical Press.

Six Principles of Security Ergonomics

A Just Culture

Organisations should adopt an open and fair process for reporting security events and error.



Proactive Design

Should proactively leverage knowledge to anticipate future user error & latent failure, as well as engineer out knowns.

Embedded by Default

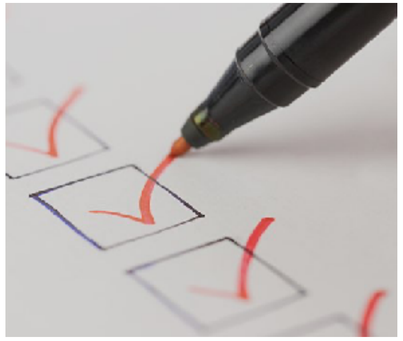
Security design should be embedded in practice, from the outset and not be left as a bolt-on or additional feature.

Encourage Pro-Secure

Designs should encourage users to take basic actions that improves security of a system.

Enforce Non-Alignment

Known error and failure should never be allowed to occur at the same time to avoid safety/security critical events occurring.



Externally Validate

Designers & developers should utilise co-design methodologies and development best practice to minimise latent failures.

1. A Just Culture

A safe environment within which security incidents are captured, evaluated and *post-mortemed*, without automatic blame, in order to provide detail into the Security Ergonomics approach for the design of.. secure.. systems.

Insight

Clear picture of what is going on in organisation. Almost impossible to gain a "complete" understanding, from multiple viewpoints as to what actually occurred without a Just Culture.

Wellbeing

Practitioners will suffer less anxiety… in the wake of an incident. Without a Just Culture it can be detrimental to commitment, job satisfaction and a willingness to step outside defined role.

Innovation

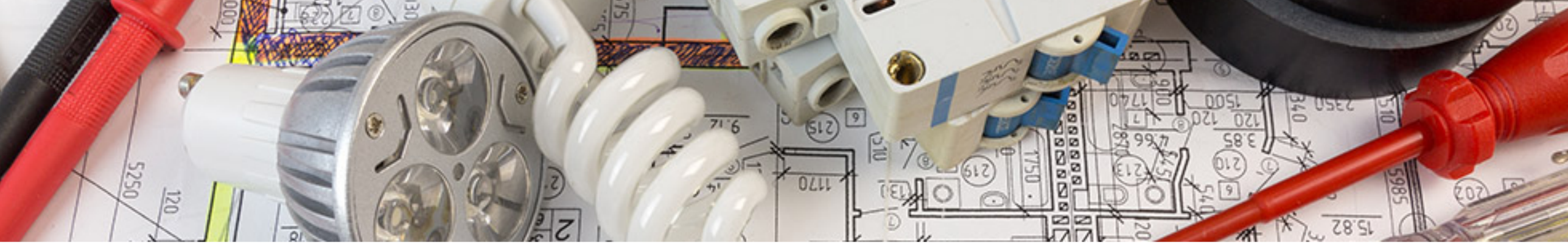
Focus can remain on long-term strategic investment in safe & secure products rather than priorities ending up focussed upon minimising short-term liability.

"a culture of trust, learning and accountability" Dekker 2018

Why the FIRST?

The 2017 Security Ergonomics by Design principles, and especially #1 Proactive Design, were reliant upon knowledge of prior error & failure.

Just Culture provides a method for organisations to gather that historical record of priors to support Security Ergonomics.



2. ProActive Design

Like Privacy by Design (PbD) and Security by Design (SbD) we must embed secure practice from the outset.

Moreover we must learn from history to avoid that which has gone before. Ours and others.

This is especially true when we look back to the “sophistication” of many attacks really laying in long established vulnerabilities - remember the SQL Injection?

Proactive design does not provide remedy for failure!

Proactive design aims to anticipate human error, identify & prevent known latent failure

3. Embedded by Default

All too often, security is reduced akin an enabling task. In the case of system design this can regularly mean that the security being implemented is not treated as a functional requirement at the design stage.

By embedding security ergonomics as a non-functional requirement it is promoted such that it is front of mind at every stage of the design process rather than being seen as an added or bolt-on feature included later.

This is critical. It is very hard to un-engineer things like bias. It's very expensive to re-factor functionality to cater for what should have been known failure.

When the security ergonomics principles are embedded at every stage, each is used to probe design choices.

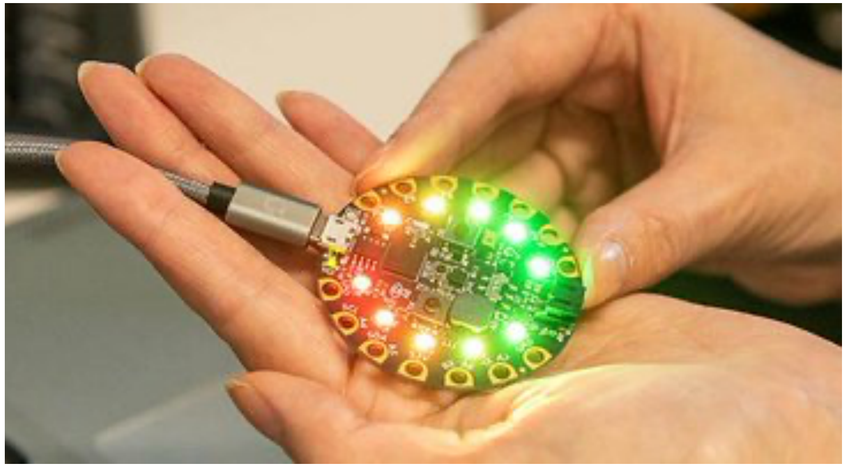
Embedding Security Ergonomics throughout system design & development turns it from enabling into a production task.

4. Encourage Secure Behaviours

Think back to the Mirai example used in the first couple of weeks. One of the key links in the failure chain was the lack of enforced changing of admin credentials.

By designers creating thoughtful interventions - sometimes modally enforcing the action before the user can continue - it is possible to create pro-secure behaviours and habits.

Wherever possible this should be encouraged, better still… enforced, addressing directly any motivational and knowledge constraints.



Useful Stuff

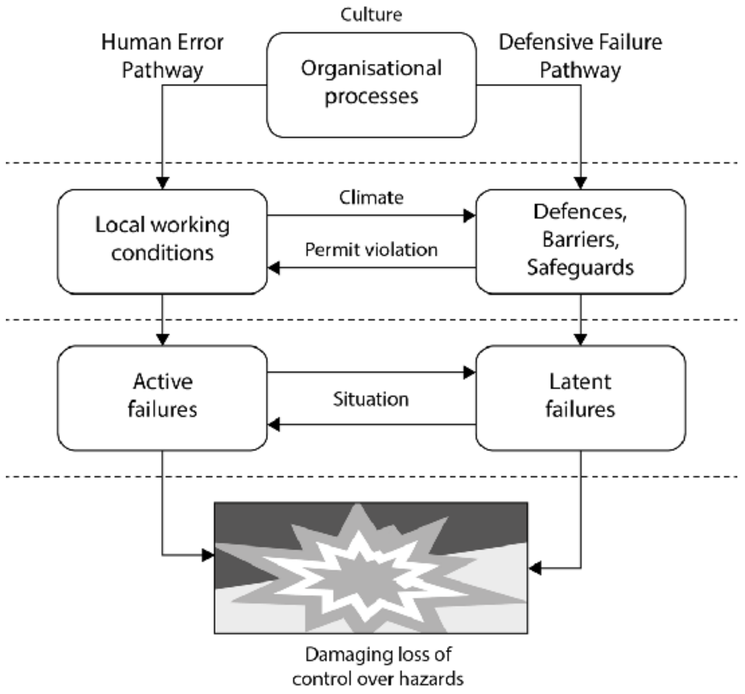
West, R. 2008
The psychology of security,
Commun. ACM, vol. 51, no. 4, pp. 34–40

Gentle Interventions for Security
RISCS
https://www.riscs.org.uk/project/gentle-interventions-for-security-gifs/

5. Non-Alignment by Default

In previous lectures we have seen how latent (system) failure can not only create an environment within human error is all inevitable, but can actually compound its impact when they align.

Principally designing out as much latent failure as is possible reduces alignment risk. However, the fifth principle **specifically requires that with knowledge of latent and human failure modes in a system, that alignment is actively avoided**.



This avoidance can be **enhanced by adding further defences, barriers and safeguards.** For example, consistent and meaningful code naming and notes to ensure developers do not inadvertently utilise incorrect libraries or snippets when they are known to potentially clash with human error.

You are **human**!

You make **mistakes**!

Remember that.

6. External Validation

We must never forget that in the effort for smarter, more automated, less HiL & more usable security what we actually do is shift the burden, the constraints and biases to ourselves as the developers.

There are a myriad of tools, methods and techniques you can use to try to avoid your own constraints coming through in your code. Things like pair coding and code reviews have the advantage of another person getting deep into the work. Open sourcing can have a similar impact.

Oliveira et al. interviewed professional developers and found that they generally trust APIs. Given this general trust, even security-minded developers may not explicitly look for vulnerabilities in API functions, with the result that blindspots cause security vulnerabilities.

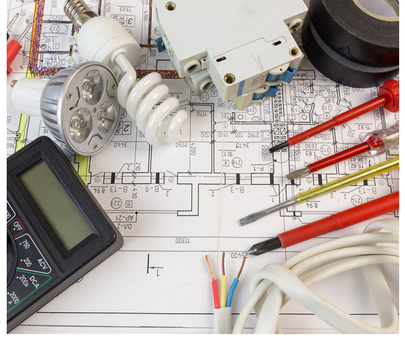
BUT BEWARE

TRUST

BIAS

Useful Stuff

D. Oliveira, M. Rosenthal, N. Morin, K.-C. Yeh, J. Cappos, and Y. Zhuang. 2014. It's the Psychology Stupid: How Heuristics Explain Software Vulnerabilities and How Priming Can Illuminate Developer's Blind Spots. https://dl.acm.org/doi/pdf/10.1145/2664243.2664254



Security Ergonomics by Design

Designed to give system designers basic principles to better **cater** for the very real **constraints of human**s & known latent failure whilst implementing best practice methods for building security applications.

The choice of underpinning tools, techniques and methods is not prescriptive. Different projects, sectors, development styles etc… will have differing “best practice.”

As design principles they are there to guide.

Your Task - Potential Extensions

As we have seen, initially Security Ergonomics by Design was five principles, before it was realised Just Culture was needed to provide the historical viewpoint upon which better systems could be designed.

I fully expect there is a seventh principle. It might be that some sort of **economic pragmatism** may be needed - understanding the risk/reward nature from both the adversarial and design perspectives.

BUT **can you think of where a seventh might lay? Or, indeed find critique for the existing six?**

The End