I certify that this submission is my
original work and meets the Faculty's
Expectations of Originality

Christopher McArthur
40004257
ENCS 393 AA
June 4th 2018

Programming's Singularity

A Singularity is a point beyond which our natural laws break down and we don't know what will happen on the other side. Often super human artificial intelligence is framed in this light because of the concept risk of exponential growth and unknown consequences. Since this has not taken place yet, the current situation, programmers being able to do *anything* is far more pressing. Identity theft used to involve combing through a victim's garbage looking for enough personal information to apply for a credit card. In todays day an age, simply watching the network traffic on a public WIFI you can steal someone's credit card information while sipping coffee. Technologies like Apple Pay and Google Wallet, which are support on all modern phones, means the majority of us are carrying a mobile credit card cloner; however, we don't know how many of those know how to use it, only assume it's a small percentage. This begs the question who is responsible when things go wrong? Is it the individual programmer, the field in its entirety or is it on society for welcoming new technologies without considering the risks? It is the responsibility of both the programmer and that of the industry to prevent new artifacts from being used with malicious intention to create "evil" technologies. Developer's should not blankly leave obvious security vulnerabilities for future exploitation. The industry should not be producing technologies with can be used differently against the same technology.

Examples of developers not morally fulfilling this obligation are everywhere, Facebook's vulnerability which allowed Cambridge Analytica to access 10 of millions of individuals information without concept is a more rest front page story. A more laughable example which really highlights a developer who did not care about the potential fall out of his inadequate job was the T-Mobile clear text password scandal[1][2]. Rumors had been circulating for several months about T-Mobile Australia operating like this, but it wasn't until the official T-Mobile Australia Twitter account confirmed the rumors explained that employees needed the first characters to confirm your identity. When confronted with the fact this was dangerous if the company were ever to experience a cyber breach, the representatives claimed their security was too good for that to happen. Behind this public relations disaster is a developer who acted in such a way that if every ignored obvious security concerns no one would have personal information just public information. How hard would it be to encrypt passwords? So easy you can copy paste the code off stack overflow from the first google hyperlink. There was no technical reason why this feature could not have been delivered, even if it was not a project requirement its an up sell that could easily be pitched to the product owners. The developers on the project knew they were storing passwords and displaying them, they should have seen the risks to anyone using their product. None of the users of T-Mobile Australia web platform consented to have their information insecurely stored so the developers had a responsibility to protect the information of those using the service.

[1] Rumors being confirmed on Twitter - https://twitter.com/tmobileat/status/981418339653300224?lang=en
[2] News article regarding the issue - https://www.theregister.co.uk/2018/04/07/security_roundup/

The computer science industry is constantly producing new technologies, and many of these have little ripple effect on our lives. Mobile pay is one of these, allowing your smart phone to act in place of a credit card's flash pay; simply open your mobile pay app and place the phone above the credit card terminal. When the transaction is received it will vibrate and you can remove the phone and grab the coffee on your way out. What many commercial users of this do not know, is this same technology can be used to read credit cards. It's so easy to use the SDK provided for this, that anyone can use this to clone credit cards. Thankfully most humans wouldn't want their credit cards stolen so they don't steal others. There's also no practical use for a cloned card without the individual's personal identification number, PIN, however since a clear majority use a birthday from someone in their immediate circle and all that information is on Facebook… It's not very hard to imagine the difficulty, or lack thereof, in finding enough information (which is the hard of computer science) to make cloning credit cards very dangerous. Why did the industry produce a technology that could so easily be turned into another more sinister technology? This is just one of many examples of new technologies which are two halves of a coin of good and evil.

Computer science is flout with moral grey zones and those who exploit its capabilities. This begs the question who is responsible for the output of technology which can be misused. Its on the shoulders of the programmer to ensure what ever he is developing is secure and does not put others at unnecessary risk. Regarding releasing new technologies, its on the industry to make sure whenever they are making public can not be misconstrued but those with less then honourable intentions. This shift in mindset could reduce the number of exploits and potentially augment the relationship between computer science and the public.