Dear Director (*Airspace and Emerging Technologies*),

I have reviewed the paper regarding 'Emerging Technology Policies' & I have found it to be significantly lacking in at least one key area.

At no point does the policy attempt to address the level of Software Quality (*an aspect relating directly to safety*) to be delivered by Drone Manufacturers. The Aviation Industry suffers from the same affliction; that is, Avionics Software is deployed immediately into production (*on aircraft*) without an attestation from the manufacturer looking like this (*below*):

- We attest that the probability of a defect having escaped detection during our Software Quality Assurance process is less than 0.435%

In other words, all manufacturers in all air related industries globally are not compelled to precisely measure or reveal their Cyber-Risk prior to deployment. Much testing is performed, but without knowledge of the residual Cyber-Risk, one can never define when to 'stop testing' prior to deployment:

- For example:
  - Q: how meaningful is it to know that 1M tests were performed ?
  - A: it isn't meaningful in the absence of knowing that 1M tests buys you 99.999% Cyber-Confidence:
    - 1M tests could buy you 2% Cyber-Confidence & 98% Cyber-Risk; this is why measuring Cyber-Risk is so important

Here is the problem:

1. Software controls everything
2. The source code for the control system is not open source; it is closed source & must be treated as a black-box
3. Software is deployed & utilised without having to meet any <u>measurement</u> standard; I did not say 'a standard' or 'standards', I said specifically <u>measurement</u> standard

The above means that software can be deployed for use containing undetected (*hidden*) defects:

- **Nowadays, it is possible to measure the probability of software defects still living (*hiding*) within software**; utilising Australian technology
- Below is a well known example of what can happen when software is deployed for use without having measured the probability of undetected (*hidden*) defects still living in the code

Boeing is still trying to get the 737 Max back into the air after **software** flaws **caused** two deadly **crashes**. Investigations found that the pilots on both flights struggled to control a malfunction in the automated maneuvering system that forced the planes' noses toward the ground. Feb 29, 2020

How Boeing software errors jeopardized its 737 Max and ...
www.businessinsider.com › Science › News

It should be mandatory that all Drone Manufacturers satisfy a **verifiably measured standard** for their software control systems prior to sale in the Australian market:

- Imagine if a drug company released products without the probability of failure being known; you'd be too frightened to medicate

A drone can be weaponised, or it can accidentally cause a catastrophe. To minimise these possibilities, a Software Quality Measurement Standard should be specified. I've included various examples of how this can be measured utilising existing Australian technology:

- These examples do not include drones, but the principles are identical

Here is the solution (*example requirement*):

- In order to obtain an airworthy certificate (*or equivalent*), manufacturers are required to ensure that all Functional Processes embedded within their software are tested to a level of Cyber-Confidence of no less than 99%:
  - This means that the maximum permissible Cyber-Risk associated with any User Function is to be less than 1%

If you require any further assistance, please reach out to me:

- To my way of thinking, the critical need to understand the probability of defects hiding within software is obvious & beyond question

1.4 **Simplified Examples**

An effective means by which to communicate the decision making power of the solution presented herein is to answer some obvious questions utilising several *simplified* examples, as follows;

1. Cyber-Security Confidence against Port Attack
2. Cyber-Security Confidence against Brute Force Attack
3. Cyber-Risk associated with Internet Banking
4. 'Splunk>Phantom'

1.4.1 Port Attack

If all **T**ransmission **C**ontrol **P**rotocol (TCP) & **U**ser **D**atagram **P**rotocol (UDP) Ports are tested & confirmed to be inactive, how secure is my computer against penetration from external actors according to **O**pen **W**eb-**A**pplication **S**ecurity **P**roject (OWASP) Test Scenarios?

- Answer: Cyber-Security Confidence = **99.83%** as shown in Fig. (1)

Where:

- The number of DIT's = the number of ports tested

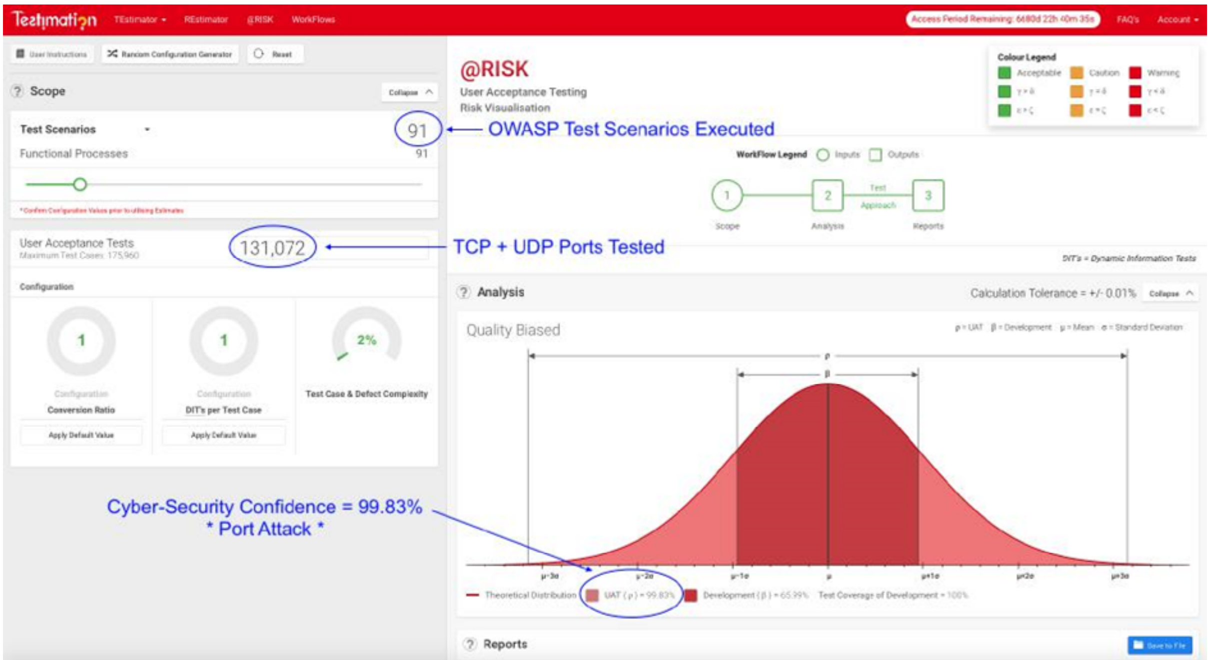

Fig. (1): Cyber-Security Confidence Measurement (*Port Attack*)

1.4.2 Brute Force Attack

What is the Cyber-Security Confidence associated with a Brute Force Attack on a User Login Function, utilising the Oxford English Dictionary as the Test Basis?

- Answer: Cyber-Security Confidence = **99.97%** as shown in Fig. (2)

Where:

- The number of DIT's = the number of words in the Oxford English Dictionary

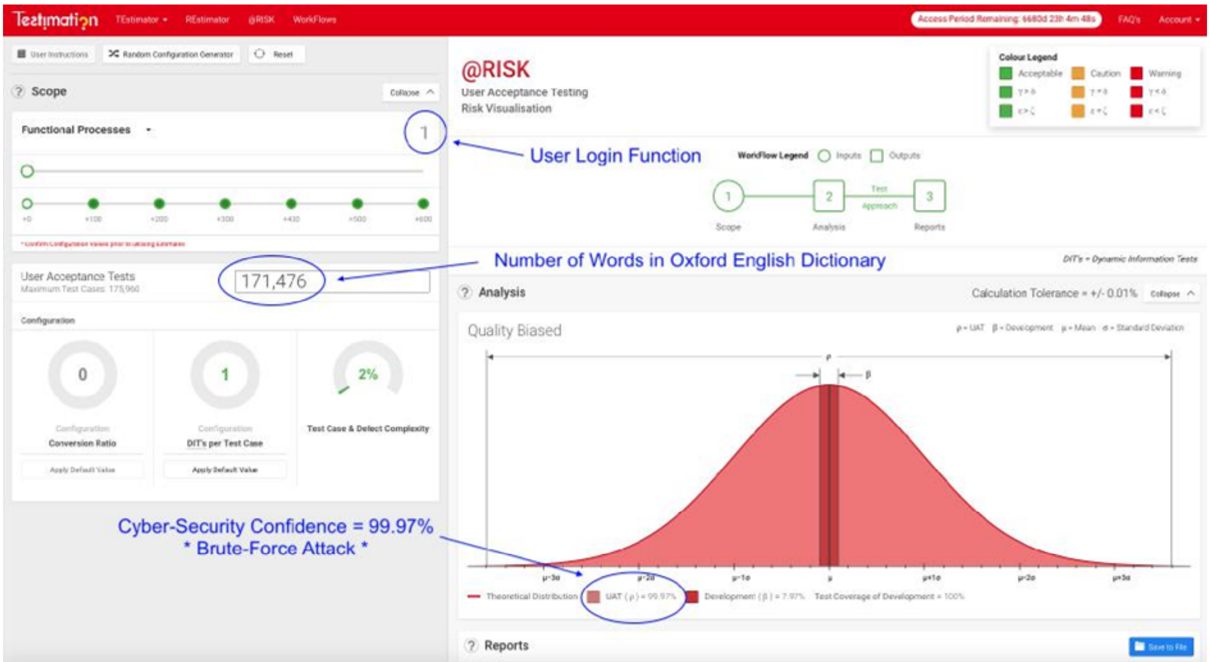

Fig. (2): Cyber-Security Confidence Measurement (*Brute Force Attack*)

1.4.3 Internet Banking

How many Test Cases are required in order to deliver Internet Banking Functionality satisfying the conditions 'Cyber-Risk ≤ 1%' & 'Cyber-Confidence ≥ 99%'?

- Answer: Number of Test Cases = **3,185** as shown in Fig. (3)

Where:

- The number of **U**ser **P**athways (UP's) through the **G**raphical **U**ser **I**nterface = 120
- The number of DIT's per Test Case = the number of Test Steps per Test Case

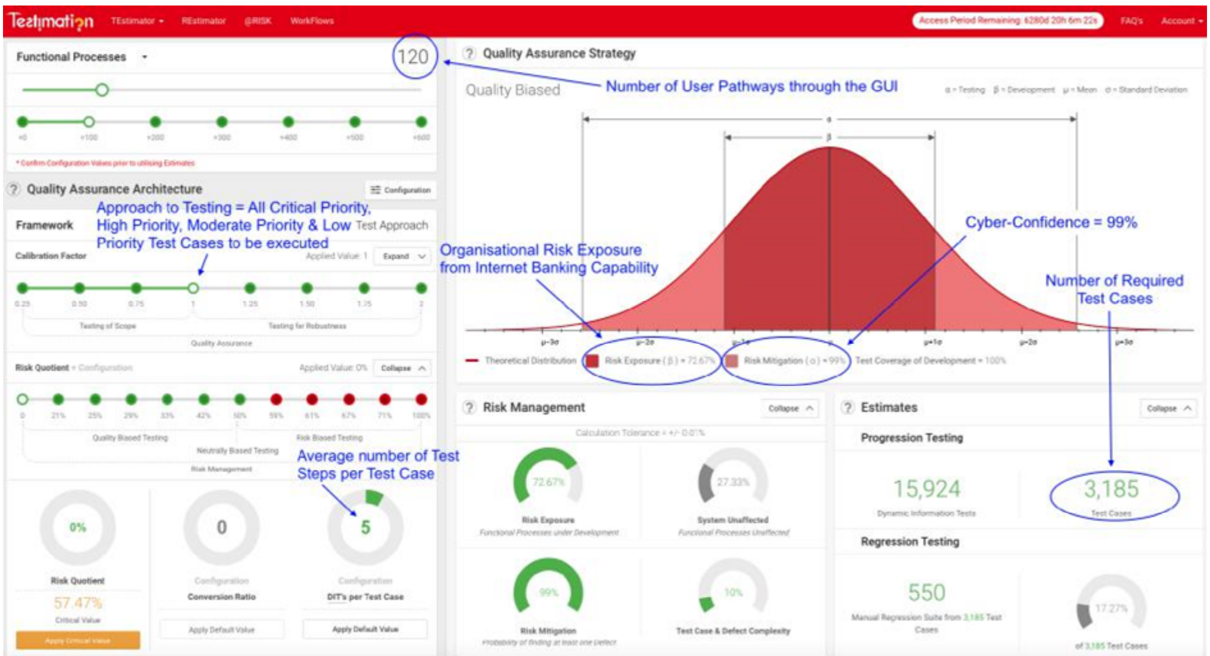

Fig. (3): Cyber-Confidence Measurement (*Internet Banking*)

Thus, designing & executing 3,185 Test Cases with an average of five (5) Test Steps per Test Case yields a 99% probability that the deployed solution is Defect-Free if all Test Cases pass.

1.4.4 **Splunk>Phantom**

'Splunk>Phantom' is a Cyber-Security **W**orkflow **A**utomation **T**ool (WAT). In this example, a **S**ecurity **O**perations **C**entre (SOC) is transitioning to the 'Splunk>Phantom' platform, but the SOC-Team has limited **Q**uality **A**ssurance (QA) experience. How many Test Cases do they need to execute in order to test the Playbook shown in Fig. (4), prior to deploying the automated Cyber-Security solution?

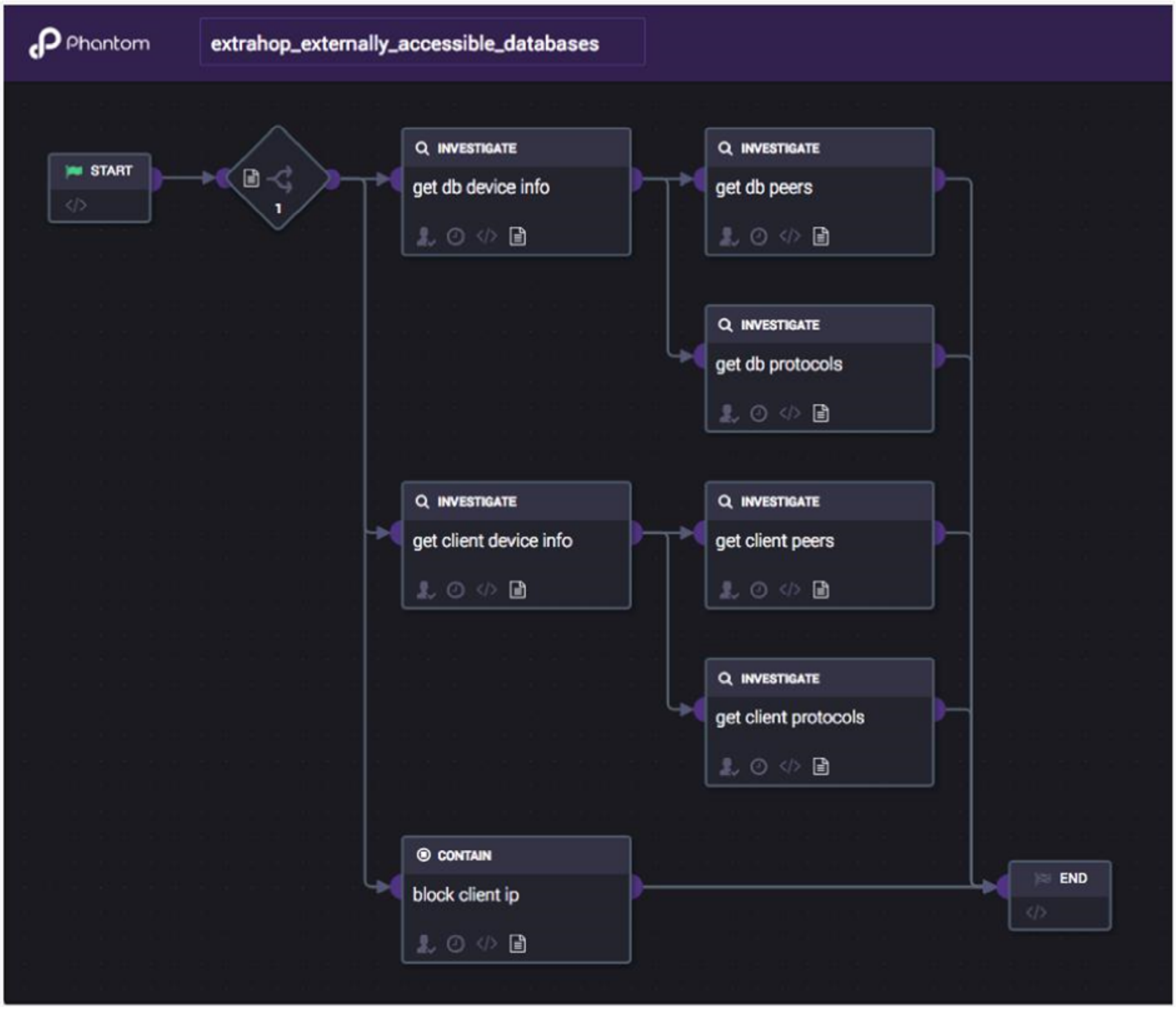

Fig. (4): Example 'Splunk>Phantom' Playbook

Fig. (4) Playbook Facts:

1. Five (5) **I**nformational **F**low **P**athways (IFP's) are drawn from Start-to-End
2. The SOC-Team have estimated or counted an average of seventeen (17) Acceptance Criteria per IFP
3. One (1) Acceptance Criteria = One (1) DIT

Utilising The Cyber-Risk Prediction & Measurement Construct presented herein, we may formulate a **D**ecision **A**ssistance **T**able (DAT) as shown in Tab. (1);

QA-Solution	Cyber-Confidence	Cyber-Risk	Test Cases	Test Cases per IFP
1	0%	100%	0	0
2	63.27%	36.73%	5	1
3	80.22%	19.78%	10	2
4	93.15%	6.85%	20	4
5	97.43%	2.57%	30	6
6	99%	1%	40	8
7	99.6%	0.4%	50	10
8	99.84%	0.16%	60	12
9	99.93%	0.07%	70	14
10	99.97%	0.03%	80	16

Tab. (1): Fig. (4) DAT

Tab. (1) demonstrates that Test Effort increases, as Cyber-Risk tends to zero; so, which QA-Solution should the SOC-Team apply? To answer this question, the SOC-Team need to recognise that each IFP from any Playbook requires a *minimum* of two Test Cases; one testing for success & one testing for failure; *i.e.* one Positive & one Negative Test Case respectively, thus QA-Solution (1, 2) may be eliminated from consideration.

In many commercial environments, QA-Teams are often pressured into minimal testing solutions. To overcome this challenge, we may utilise Fig. (5) to specify the optimal QA-Solution from Tab. (1). In Fig. (5), we see that dimension 'α' is much greater than dimension '13' (α >> 13) for QA-Solution (3). Hence, the Risk Mitigation is much greater that the organisational Risk Exposure associated with the transition from manual to automated Workflows. Therefore, QA-Solution (3) denotes the optimal target such that:

- Risk Exposure = 17.69% α organisational impact of workflow changes
- Risk Mitigation = 80.22% = Cyber-Confidence α test coverage
- Cyber-Risk = 19.78% α the testing not executed

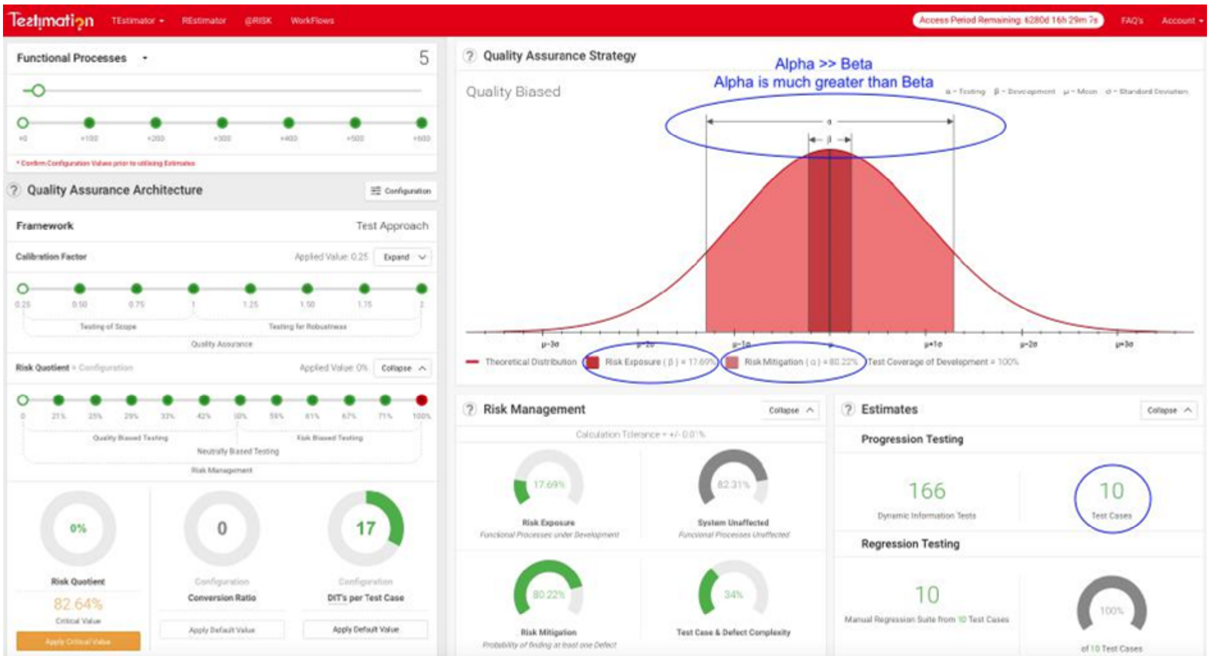

Fig. (5): Cyber-Confidence Measurement (*'Splunk>Phantom' Playbook*)