## 1 Introduction

The newly proposed ML platform has caused doubts among AppSEC as it allows sending confidential data to a centralised shared environment. There is heated discussion around the possibility of potential ”cross-talk”, i.e. information from client A, leaks to client B. In this document, we will draft an approach to statistically verify whether a stronger condition, i.e. the function is pure, holds. Once we can verify that it is true, we can guarantee cross-talk does not happen with high confidence.

## 2 Pure function

We follow an analogous definition of pure function in programming literature: a function (or an API call) is pure if its returns values is the same for the same arguments.

What we want to safe-guard is that all three endpoints provided in ML API Server is pure with respect to the arguments. We argue that non-purity is a necessary condition for cross-talk, assuming the only way ML Platform can exchange information with clients are through the function calls, and the following is the draft of a proof.

W.L.O.G., there are two clients A, B calling ML API. When cross-talk occurs, confidential information $a$ from client A is contained in a return value $f_n(x_b)$ to a call from client B. $f_n$ indicates it is the corresponding function to respond the nth invocation of the API call. Note that this information must be originated from a previous call to $f_i, i < n$ from client A, as this is the only way $a$ can be included in $f_n(x_b)$.

Consider the second scenario where $f_1$[1] is called with argument $x_b$ by B. Since this is the first invocation, , it is impossible $a$ is included. That means with the same arguments, two calls return different results. Q.E.D.

As long as we can (statistically) guarantee purity, i.e. $f_i = f_j, i \neq j$, no cross-talk is possible.

## 3 Hypothesis testing on purity

Given that we cannot enumerate all possible inputs to the function, we have to come up with a testing scheme that, given a finite amount of samples, we can guarantee how unlikely non-purity will occur. For that we define the two hypotheses:

$$\begin{aligned} \mathcal{H}_0 &: \text{the function is not pure} \\ \mathcal{H}_1 &: \text{the function is pure} \end{aligned} \tag{1}$$

[1] $f_1$ is the function replying to the first call to $f$

What we would like to prevent is type I error, i.e. when $\mathcal{H}_0$ is true but we reject $\mathcal{H}_0$. We would like to reduce the likelihood of this happening to a small enough number:

$$\mathbb{P}(\text{we reject } \mathcal{H}_0 | f \in \mathcal{H}_0) \tag{2}$$

In the following, we describe how we model this probability.

Let a sequence of inputs be $x_1, ..., x_n$ and its corresponding return values be $y_1, ..., y_n, y_i = f_i(x_i)$. We model the purity by the following probability

$$\begin{aligned} p(y_1, ..., y_n | x_1, ..., x_n) &= \prod_i p(y_i | x_{1...i}) \\ p(y_i | x_{1...i}) &= p(y_i | t_i = 1, x_i) p(t_i = 1) \\ &+ p(y_i | t_i = 0, x_{i...1}) p(t_i = 0) \\ &= \mathbf{I}(y_i = f_1(x_i))\alpha + C(1-\alpha)\mathbf{I}(y_i \neq f_1(x_i)) \end{aligned} \tag{3}$$

where $p(t_i)$ is the likelihood of $f_i(x_i) = f_1(x_1)$, and we assume this likelihood is constant, i.e. equals to $\alpha$, across different invocations of the function. $C$ is an unknown function in the form of $C(y_i, x_1, ..., x_i)$, and $C\mathbf{I}(y_i \neq f_1(x_i)) = p(y_i | t_i = 0, x_{i...1})^2$.

If$f$ is pure, $p(t_i = 1) = \alpha = 1$.

We can set the test-statistics to be the number of mismatches, $\sum_i \mathbf{I}(f_i(x_i) \neq f_1(x_i))$, and the inference rule to be the following

1. If there is more than one mismatch, accept $\mathcal{H}_0$
2. If there is 0 mismatch reject $\mathcal{H}_0$, accept $\mathcal{H}_1$

Under this rule, we would like to reduce Type I error. We consider two cases separately.

If there is one mismatch, we won't reject $\mathcal{H}_0$, so no type I errors will be made.

If there is no mismatch, n matches, the likelihood of Type I error will be

$$\mathbb{P}(\text{we reject } \mathcal{H}_0 | f \in \mathcal{H}_0, \text{n matches}) = \alpha^n \tag{4}$$

If we want to bound the error to a confidence level $\delta$, then we only need to set $n >= \log_\alpha(1-\delta)$

For example, if we pessimistically assume that the likelihood of $f$ return the same value as $f_1$ is very high when $f$ is not pure, we can set it to 0.9999. Moreover we want to make sure our check is with confidence level $\delta = 0.999$. Getting all these numbers into the formula we will need at least 70000 matches and zero mismatch to the returned values of $f_1$ during the test.

[2] We don't know C, and we do not need to know it, if we know $f_1(x_i)$

## 4 When no golden set is available

The test above requires us to have the golden responses, i.e. $f_1(x_i)$, to a list of requests. In practice, this may be difficult to collect because the only way we can guarantee that a response is golden is that we restart the pod, i.e. purge all the states in the runtime, for every request. In the following we describe an alternative strategy that requires only running the same requests in multiple runs.

In each run, we shuffle the ordering of the requests, and get their corresponding responses. Let $X = \{x_1, ..., x_n\}$ be the sequence of requests, $\pi_j \in N \to X$ be the permutation function[3] of the j-th run, and we call $f$ in the orders of $f_1(\pi_j(1)), ..., f_n(\pi_j(n))$. To simplify discussion, we assume permutation function for the first one does not shuffle any element. i.e. $\pi_i(j) = x_j$. $X_j = (x_{j,1}, ..., x_{j,n})$ is the set of requests in j-th run that is generated from X shuffled by $\pi_j$.

We denote $Y_j = \{y_{j,1}, ..., y_{j,n}\}$ be the results from the j-th run and it has been reordered so that $y_{j,k} = f_{\pi_j^{-1}(x_k)}(x_k)$.

We change the test statistics to be $\sum_{\substack{i\\ j\neq 1}} I(y_{j,i} = y_{1,i})$

The acceptance/rejection rule is modified to the following

1. If there is more than one mismatch, reject $\mathcal{H}_0$
2. After T runs on N requests, there is no mismatch of each run to the first run, reject $\mathcal{H}_0$, accept $\mathcal{H}_1$

To analyse the likelihood of Type I error, we assume that

$$\begin{aligned}
p(Y_1, ..., Y_n | X_1, ..., X_n) &= \prod_j p(Y_j | X_j) \\
p(Y_j | X_j) &= \prod_i p(y_{j,i} | X_j) \\
p(y_{j,i} | X_j) &= p(y_{j,i} | t_i = 1, X_j) p(t_i = 1) \\
&\quad + p(y_{j,i} | t_i = 0, X_j) p(t_i = 0) \\
&= \mathbf{I}(Y_{j,i} = f_1(x_i))\alpha + C(1-\alpha)\mathbf{I}(y_{j,i} \neq f_1(x_i))
\end{aligned} \tag{5}$$

This is an extension of the previous model where this jointly models the probability of multiple-runs. Let $E$ be $\{X_1, ..., X_T\}$ , we would like to model the probability of

$$\begin{aligned}
&\mathbb{P}(\text{we reject } \mathcal{H}_0 | f \in \mathcal{H}_0, \text{ no mismatches after T runs on n requests}) \\
=&p(Y_2 = Y_1, ..., Y_T = Y_1 | E) \\
=&\prod_i p(y_{2,i} = y_{1,i}, ..., y_{T,i} = y_{1,i} | E)
\end{aligned} \tag{6}$$

[3] $\pi_j$ is bijective.

$$\begin{aligned}
&p(y_{2,i} = y_{1,i}, ..., y_{T,i} = y_{1,i}|E) \\
=&p(y_{2,i} = y_{1,i}, ..., y_{T,i} = y_{1,i}|E, y_{1,i} = f_1(x_i))p(y_{1,i} = f_1(x_i)|E) \\
+&p(y_{2,i} = y_{1,i}, ..., y_{T,i} = y_{1,i}|E, y_{1,i} \neq f_1(x_i))p(y_{1,i} \neq f_1(x_i)|E) \\
=&p(y_{2,i} = y_{1,i}, ..., y_{T,i} = y_{1,i}|E, y_{1,i} = f_1(x_i))\alpha \\
+&p(y_{2,i} = y_{1,i}, ..., y_{T,i} = y_{1,i}|E, y_{1,i} \neq f_1(x_i))(1-\alpha)
\end{aligned} \tag{7}$$

$$\begin{aligned}
&p(y_{2,i} = y_{1,i}, ..., y_{T,i} = y_{1,i}|E, y_{1,i} = f_1(x_i)) \\
=&\prod_{j\neq 1} p(y_{j,i} = y_{1,i}|E, y_{1,i} = f_1(x_i)) \\
=&\alpha^{T-1}
\end{aligned} \tag{8}$$

$$\begin{aligned}
&p(y_{2,i} = y_{1,i}, ..., y_{T,i} = y_{1,i}|E, y_{1,i} \neq f_1(x_i)) \\
=&\prod_{j\neq 1} p(y_{j,i} = y_{1,i}|E, y_{1,i} \neq f_1(x_i))
\end{aligned} \tag{9}$$

Since we have no knowledge at all on the probability $C$, we assume that we can bound the probability by a constant $C \leq \beta$. In other words, we assume, given that in the first run of the experiment, response of $f$ to an input $x_i$ does not give a golden response, the maximum likelihood that in the subsequent run, response of $f$ to the same input will be equal to the result form the first run to be $\beta$. The eq. 9 can be bounded as follows:

$$\prod_{j\neq 1} p(y_{j,i} = y_{1,i}|E, y_{1,i} \neq f_1(x_i)) \leq (\beta(1-\alpha))^{T-1} \tag{10}$$

We can bound Type-I error as follows:

$$\begin{aligned}
&\mathbb{P}(\text{we reject } \mathcal{H}_0|f \in \mathcal{H}_0, \text{ no mismatches after T runs on n requests}) \\
\leq&\left[\alpha^T + (1-\alpha)^T \beta^{T-1}\right]^n
\end{aligned} \tag{11}$$

Assume that we only have 2000 requests, and we are very pessimistic about $\alpha$ and $\beta$[4]. If we set $\alpha = \beta = 0.9999$, we want to find out what with confidence level $\delta = 0.999$, what is the minimum T we need to have:

$$\left[\alpha^T + (1-\alpha)^T \beta^{T-1}\right]^n \leq 1 - \delta \tag{12}$$

By numerical solver, we can find out the answer to be $T \geq 34.54$

[4]To reflect the worse case that the likelihood of $f$ returning the same outputs given an input, while the output does not equal to the golden response is high, we set $\beta$ to be high.