1 Syntax of obfuscation

Definition 1.1 (Obfuscator Syntax)**.** An obfuscator for a class of circuits $\mathcal{C}_{\lambda\lambda\in\mathbb{N}}$ is a randomized algorithm obf such that $\mathsf{obf}(C, 1^\lambda)$ returns a functionally equivalent circuit to C for all $C \in \mathcal{C}_\lambda$, formally, for all $C \in \mathcal{C}_\lambda$, we have

$$\Pr_{\tilde{C} \leftarrow\$ \mathsf{obf}(C,1^\lambda)}\left[\forall x \in \{0,1\}^\ell : \tilde{C}(x) = C(x)\right],$$

where $\ell = \mathsf{input} - \mathsf{size}(C)$

2 Impossibility of Virtual Black-Box (VBB) Obfuscation

In 2001, Barak, Goldreich, Impagliazzo, Rudich, Sahai, Vadhan, and Yang (BGIRSY) showed that an obfuscator cannot provide security as good as having only black-box access to a circuit, known as *virtual black-box obfuscation.* Their main idea is to take a circuit describing a point function $f_{x,y}$ for x and y drawn uniformly from $\{0,1\}^\lambda$ and a circuit describing a point testing function $t_{x,y,s}$ where additionally s is drawn uniformly random from $\{0,1\}^\lambda$:

$f_{x,y}(x')$	$t_{x,y,s}(f)$
if $x = x'$	**if** $f(x) = y$
return y	**return** s
else return $0^{\lvert y\rvert}$	**else return** $0^{\lvert s\rvert}$

Now, given an obfuscation of $f_{x,y}$ and $t_{x,y,s}$, one can run $t_{x,y,s}$ on the obfuscation of $f_{x,y}$ and get s as a result. However, when only having black-box access to $f_{x,y}$ and $t_{x,y,s}$, this is not possible. Hence, an obfuscator must always leak more than black-box access. For a detailed discussion and the definition of virtual black-box obfuscation, see `https://eccc.weizmann.ac.il//eccc-reports/2001/TR01-057/index.html`.

3 Indistinguishability obfuscation

BGIRSY explored definitions of obfuscations which where not affected by their impossibility result. They found that *indistinguishability obfuscation* was not affected. 12 years later, Garg, Gentry, Halevi, Raykova, Sahai and Waters presented the first candidate indistinguishability obfuscation construction `https://eprint.iacr.org/2013/451`. Equally importantly, in the same month, Sahai and Waters presented their puncturable program technique which revealed the tremendous usefulness of indistinguishability obfuscation `https://eprint.iacr.org/2013/454`. Earlier this year, Jain, Lin and Sahai, in breakthrough result, based an indistinguishability obfuscator on widely believed assumptions, after 7 years of intense research in the field, see

`https://www.ias.edu/video/indistinguishability-obfuscation-well-founded-assumptions`

for a talk by Lin and `https://eprint.iacr.org/2020/1003` for the paper.
But which level of security does indistinguishability obfuscation actually provide? Indistinguishability obfuscation ensures that, given two circuits C_0 and C_1 *with the same functionality*, we cannot distinguish between obfuscations of C_0 and C_1.

Definition 3.1 (Indistinguishability Obfuscation)**.** Let p be a polynomial and obf be an obfuscator for $(\mathcal{C}_\lambda)_{\lambda\in\mathbb{N}}$, where $(\mathcal{C}_\lambda)_{\lambda\in\mathbb{N}}$ is such that for all $\lambda \in \mathbb{N}$, $|C| \leq p(\lambda)$. We consider a PPT algorithm $\mathcal{S}$ an *equivalent circuit sampler* for $(\mathcal{C}_\lambda)_{\lambda\in\mathbb{N}}$ if for all $\lambda \in \mathbb{N}$, it holds that

$$\Pr_{(C_0,C_1)\leftarrow\$\mathcal{S}(1^\lambda)}\Big[C_0, C_1 \in \mathcal{C}_\lambda \wedge \ell(C_0) = \ell(C_1) \wedge \forall x \in \{0,1\}^{\ell(C_0)} : C_0(x) = C_1(x)\Big],$$

where $\ell(C_0)$ denotes the input length of C_0 and $\ell(C_1)$ denotes the input length of C_1.
obf is *indistinguishable under equivalent circuit sampling* (IND-ECS) if for all equivalent circuit samplers $\mathcal{S}$ for $(\mathcal{C}_\lambda)_{\lambda\in\mathbb{N}}$, it holds that $\texttt{Gind-ecs}^0_{\mathsf{obf},\mathcal{S}}$ and $\texttt{Gind-ecs}^1_{\mathsf{obf},\mathcal{S}}$ are indistinguishable, i.e., for all PPT adversaries $\mathcal{A}$, we have that

$$\left|\Pr\big[1 = \mathcal{A} \rightarrow \texttt{Gind-ecs}^0_{\mathsf{obf},\mathcal{S}}\big] - \Pr\big[1 = \mathcal{A} \rightarrow \texttt{Gind-ecs}^1_{\mathsf{obf},\mathcal{S}}\big]\right|$$

is negligible.

$\texttt{Gind-ecs}^0_{\mathsf{obf},\mathcal{S}}$		$\texttt{Gind-ecs}^1_{\mathsf{obf},\mathcal{S}}$	
Parameters		Parameters	
λ:	security parameter	λ:	security parameter
obf:	obfuscator	obf:	obfuscator
$\mathcal{S}$:	equiv. circ. sampler	$\mathcal{S}$:	equiv. circ. sampler
Package State		Package State	
no state		no state	
$\mathsf{OBF}()$		$\mathsf{OBF}()$	
$(C_0, C_1) \leftarrow\$ \mathcal{S}(1^\lambda)$		$(C_0, C_1) \leftarrow\$ \mathcal{S}(1^\lambda)$	
$C \leftarrow\$ \mathsf{obf}(C_0, 1^\lambda)$		$C \leftarrow\$ \mathsf{obf}(C_1, 1^\lambda)$	
return (C_0, C_1, C)		**return** (C_0, C_1, C)	

Remark. Note that the only difference between $\texttt{Gind-ecs}^0_{\mathsf{obf},\mathcal{S}}$ and $\texttt{Gind-ecs}^1_{\mathsf{obf},\mathcal{S}}$ is that one of the games obfuscates C_0 and the other obfuscates C_1.