CS276: Cryptography November 12, 2015

Indistinguishability Obfuscation

Instructor: Alessandro Chiesa *Scribe: Joseph Hui*

Indistinguishability Obfuscation

Last time we discussed the idea of virtual black-box obfuscation and showed that it was impossible. We will now introduce another, weaker definition of obfuscation (indistinguishability obfuscation), in the hopes of finding a useful and achievable notion. We'll be working in circuits for this lecture, but the ideas apply equally well to Turing machines.

Definition 1 *An IO obfuscator for circuits is a ppt algorithm $\mathcal{O}$ such that:*

1. *Correctness: $\forall k \in \mathbb{N}$ and for all circuits C, $\mathcal{O}\left(1^k, C\right) \equiv C$ (where $\equiv$ indicates that two circuits are functionally equivalent, i.e. have the same output for every input).*
2. *Efficiency: $\left|\mathcal{O}\left(1^k, C\right)\right|$ is polynomial in $k + |C|$. Note that this is actually implied by $\mathcal{O}$ being ppt, so there is no need to check it.*
3. *IO security: $\forall C_1, C_2$ s.t. $C_1 \equiv C_2$ and $|C_1| = |C_2|$, we have $\mathcal{O}\left(1^k, C_1\right) \stackrel{c}{=} \mathcal{O}\left(1^k, C_2\right)$. That is, if two circuits are the same size and compute the same function, then the distributions of their obfuscations are computationally indistinguishable.*

We previously showed that virtual black-box obfuscators, combined with one-way functions, yield a public-key encryption system. However, we cannot produce VBB obfuscators. Fortunately, IO obfuscation + OWFs also gives public-key encryption.

Theorem 2 *IO+OWF$\Longrightarrow$PKE*

The proof proceeds in two parts. First, we will define a notion called "witness encryption" and prove that IO$\Longrightarrow$WE. We will then combine WE with one-way functions to make a public-key encryption scheme.

Definition 3 *A witness encryption scheme for some NP relation R is a tuple of ppt algorithms (E, D)such that:*

1. *Correctness: $\forall k, \forall (x, w) \in R, \forall m \in \{0,1\}$, we have $D\left(1^k, E\left(1^k, x, m\right), w\right) = m$ with probability 1.*
2. *Security: $\forall x \notin L(R)$ (where $L(R)$ is the language implied by the relation), $E\left(1^k, x, 0\right) \stackrel{c}{=} E\left(1^k, x, 1\right)$.*

Theorem 4 *IO$\Longrightarrow$WE*

Proof: Let $\mathcal{O}$ be an IO obfuscator. Let $C_{x,m}$ be a circuit that outputs m on input z if $(x,z) \in R$, and otherwise outputs $\perp$. Let $E\left(1^k, x, m\right) := \mathcal{O}\left(C_{x,m}\right)$ and $D\left(1^k, c, w\right) := c\left(w\right)$. Correctness is easy to verify. Security follows because $E\left(1^k, x, 0\right) = \mathcal{O}\left(C_{x,0}\right) \stackrel{c}{=} \mathcal{O}\left(C_{x,1}\right) = E\left(1^k, x, 1\right)$. The middle equality follows because if $x \notin L\left(R\right)$, then $C_{x,0} \equiv C_{x,1}$ (that is, they both only output $\perp$ because there is no valid witness), and we can apply the definition of IO security. □

Theorem 5 *WE+OWF$\Longrightarrow$PKE*

Proof: If we have an OWF, we can produce a PRG, say $\hat{G}$ with expansion factor 2. We also use the witness encryption scheme $\left(\tilde{E}, \tilde{D}\right)$. Now we define our encryption scheme as follows. First, we define a generator $G\left(1^k\right)$: generate a random string s, and let $pk = \hat{G}\left(s\right)$,$sk = s$. Now, we want to apply witness encryption to the relation $R = \left\{\left(\hat{G}\left(s\right), s\right)\right\}_s$. Our encryption algorithm is $E\left(1^k, pk, m\right) := \tilde{E}\left(1^k, pk, m\right)$ and our decryption algorithm is $D\left(1^k, sk, c\right) := \tilde{D}\left(1^k, c, sk\right)$. We want to show that $\left(pk, E\left(pk, 0\right)\right) \stackrel{c}{=} \left(pk, E\left(pk, 1\right)\right)$. This follows because of the following chain of equalities:

$$\left(pk, E\left(pk, 0\right)\right) = \left(\hat{G}\left(U_k\right), E\left(\hat{G}\left(U_k\right), 0\right)\right) \stackrel{c}{=} \left(U_{2k}, E\left(U_{2k}, 0\right)\right) \stackrel{c}{=} \left(U_{2k}, E\left(U_{2k}, 1\right)\right)$$
$$\stackrel{c}{=} \left(\hat{G}\left(U_k\right), E\left(\hat{G}\left(U_k\right), 1\right)\right) = \left(pk, E\left(pk, 1\right)\right)$$

The first equality is by definition. The second follows from the pseudorandomness of $\hat{G}$. The third follows because there are only 2^k strings in $L\left(R\right)$, so the probability that $U_{2k} \in L\left(R\right)$ is negligible; and witness encryption guarantees that the two distributions are indistinguishable when constrained to $U_{2k} \notin L\left(R\right)$. The right side is symmetric, so we are done. □

Best Possible Obfuscation

Let's now consider another definition of obfuscation, "best possible obfuscation," or BPO. We know that VBB obfuscation is too strong, and IO is reasonably weak, and we want a definition that is in between the two. The idea is that there may be some information about a circuit which cannot possibly be hidden by any functionality-preserving obfuscation. We ask only that an obfuscator do the best that it can. In particular, we want an adversary to be able to learn from the obfuscation of C only that information which could be learned from any equivalent circuit (for example, the output for a given input). Formally:

Definition 6 *A BPO obfuscator is a ppt algorithm with correctness and security (defined the same way as in IO), and*

1. *BPO security: $\forall pptA, \exists pptS$ s.t. $\forall$circuits C_1, C_2 s.t. $C_1 \equiv C_2$ and $|C_1| = |C_2|$, we have $A\left(\mathcal{O}\left(1^k, C_1\right)\right) \stackrel{c}{=} S\left(1^k, C_2\right)$.*

First we should check that BPO is indeed weaker than VBB (that is, VBB implies BPO).

Lemma 7 *Every VBB obfuscator is a BPO obfuscator.*

Proof: We will suppose that an obfuscator $\mathcal{O}$ is not BPO, and show that it is also not VBB. Negating the definition of BPO, we know that $\exists \text{ppt} A$ s.t. $\forall \text{ppt}\ S\ \exists C_1, C_2, D$ such that $|\Pr[D(A(\mathcal{O}(C_1))) = 1] - \Pr[D(S(C_2)) = 1]|$ is some non-negligible function $\delta(k)$. In particular, this is true if we set $S(C) := A(\mathcal{O}(C))$. Then $|\Pr[D(A(\mathcal{O}(C_1))) = 1] - \Pr[D(A(\mathcal{O}(C_2))) = 1]|$ is non-negligible. Letting $D' = D \circ A$, we have constructed a distinguisher that distinguishes between $\mathcal{O}(C_1)$ and $\mathcal{O}(C_2)$. Now, if the obfuscator were VBB, we could find an eppt algorithm $S_{D'}$ such that $\left|\Pr[D'(\mathcal{O}(C_1)) = 1] - \Pr\left[S_{D'}^{C_1}\left(1^{k+|C_1|}\right) = 1\right]\right|$ is negligible. But because C_1, C_2 have the same size and functionality, $S_{D'}^{C}\left(1^{k+|C|}\right)$ operates exactly the same with C_1 and C_2, and adding and subtracting, we must have $|\Pr[D'(\mathcal{O}(C_1)) = 1] - \Pr[D'(\mathcal{O}(C_2)) = 1]|$, a contradiction. □

We should also check that BPO is stronger than IO.

Lemma 8 *Every BPO obfuscator is an IO obfuscator.*

Proof: Let A in the definition of BPO security be the identity function. Then $\exists \text{ppt} S$ s.t. $\forall$circuits C_1, C_2 s.t. $C_1 \equiv C_2$ and $|C_1| = |C_2|$, we have $\mathcal{O}\left(1^k, C_1\right) \stackrel{c}{=} S\left(1^k, C_2\right)$. In particular, this also holds for $C_1 = C_2$, in which case $S\left(1^k, C_2\right) \stackrel{c}{=} \mathcal{O}\left(1^k, C_2\right)$. So for any C_1, C_2, we have $\mathcal{O}\left(1^k, C_1\right) \stackrel{c}{=} S\left(1^k, C_2\right) \stackrel{c}{=} \mathcal{O}\left(1^k, C_2\right)$, as required in the definition of IO security. □

Actually, BP and IO are the same! We can prove:

Lemma 9 *Every IO obfuscator is a BPO obfuscator.*

Proof: Fix some A in the definition of BPO security. We will construct a simulator S for it, defined as $S\left(1^k, C\right) := A\left(\mathcal{O}\left(1^k, C\right)\right)$. Now we need to show that $\forall C_1, C_2$ s.t. $C_1 \equiv C_2$ and $|C_1| = |C_2|$, $A\left(\mathcal{O}\left(1^k, C_1\right)\right) \stackrel{c}{=} S\left(1^k, C_2\right)$. But this is just showing that $A\left(\mathcal{O}\left(1^k, C_1\right)\right) \stackrel{c}{=} A\left(\mathcal{O}\left(1^k, C_2\right)\right)$, which is true by IO security, so we are done. □

IO amplification/bootstrapping

There may be some classes of circuits that are easier or harder to obfuscate. For example, it could be that we know how to construct obfuscators that work on shallow circuits but not deep ones. We will show that using fully homomorphic encryption, we can bootstrap an IO obfuscator for circuits in NC_1 to an IO obfuscator for circuits of size polynomial in the input. NC_1 is the class of uniform circuit families of size polynomial in n (the size of the input), and polylog (n) depth (and fan-in 2). We will give the intuition for the proof here, but will not give the complete proof until next lecture.

Theorem 10 *IO for* NC_1 *+ FHE* $\Longrightarrow$ *IO for polysize*

The main idea is that the data and the program are equivalent. We will need a fully-homomorphic encryption scheme, which allows us to apply a function to an encrypted message to get an encryption of the function applied to that message. Informally, instead of using our IO obfuscator to obfuscate the circuit itself, we will encrypt the circuit, and then obfuscate the decryption algorithm. Someone with the encrypted circuit, plus the obfuscated decryption algorithm, will be able to evaluate an input x by producing a function f which, given a circuit C, returns $C(x)$ (that is, it is a universal circuit evaluator with input hardcoded as x); and then using FHE to apply f on the encryption of C to get an encrypted version of $C(x)$; and then decrypting to get $C(x)$. However, since C is encrypted, they will not be able to discover anything about the circuit itself.

Definition 11 *An FHE scheme is a 4-tuple* $(\text{Gen}, \text{Enc}, \text{Dec}, \text{Eval})$ *such that* $(\text{Gen}, \text{Enc}, \text{Dec})$ *is a CPA-secure PKE scheme, and* $\text{Eval}(pk, C, \text{Enc}(pk, x))$ *is in the image of* $\text{Enc}(pk, C(x))$.

Fact 12 *Fully homomorphic encryption schemes, with* $\text{Dec} \in \text{NC}_1$*, can be constructed using ideal lattices.*

Let's sketch out an idea for the obfuscation scheme we described. First, we generate a keypair for the FHE scheme with $(pk, sk) \leftarrow \text{Gen}\left(1^k\right)$. Then we encrypt the circuit as $e \leftarrow \text{Enc}(pk, C)$. Then we provide an obfuscated decryption algorithm, $\hat{\text{Dec}} \leftarrow \mathcal{O}(\text{Dec}_{sk})$. We output the obfuscated circuit $\left(\hat{\text{Dec}}, e\right)$.

Now, there are two problems with this construction. First of all, we are giving away unfettered access to a decryption algorithm $\hat{\text{Dec}}$; an adversary could simply apply it to e, recovering the original circuit C and rendering the obfuscation completely useless. Furthermore, $\hat{\text{Dec}}$ may leak the value of sk, which would allow an adversary to recover the secret key and again apply it to e. We must modify this scheme in order to make $\hat{\text{Dec}}$ refuse to decrypt things that it is not supposed to.