CS 7880 Graduate Cryptography December 6th, 2017

Lecture 25: Obfuscation

Lecturer: Daniel Wichs *Scribe: Yashvanth Kondi*

1 Topics Covered

- Definition and impossibility of Virtual Black Box Obfuscation for circuits and Turing Machines.
- Introduction to Indistinguishability Obfuscation and its applications (Witness Encryption, PKE).

In this lecture, we are introduced to the notion of obfuscation. Intuitively, an obfuscated program hides all attributes of the internal program whose functionality it computes. This includes hiding useful information such as hard-coded secret values, etc. We see that this powerful cryptographic primitive is impossible to instantiate in its strongest form (VBB) for arbitrary circuits and Turing Machines. We then consider the weaker notion of Indistinguishability Obfuscation (IO), and see how to build various cryptographic primitives using IO and One-Way Functions (OWFs).

2 Virtual Black Box Obfuscation

The properties required of an obfuscator were formalized in the seminal work of Barak et al. [BGI+01]. Correctness of an obfuscator is a natural property:

Definition 1 (Functionality Preserving) For all programs P_n and security parameters $\lambda \in \mathbb{N}$, a VBB obfuscator Obf outputs $\tilde{P}_n \leftarrow \mathsf{Obf}\left(1^\lambda, P_n\right)$ such that $\tilde{P}_n(x) = P_n(x)$ for every x in the domain of P_n. $\diamondsuit$

Consider the following attempt at formalizing security for a VBB obfuscator:

Definition 2 ('Strong' Virtual Black Box) A uniform PPT algorithm Obf is a 'strong' VBB obfuscator if it is functionality preserving and $\forall$ PPT adversaries $\mathcal{A}$, $\exists$ PPT simulator $\mathcal{S}$ such that for all programs $\{P_n\}$ and security parameters $\lambda \in \mathbb{N}$:

$$\left\{\mathcal{A}\left(\mathsf{Obf}\left(1^\lambda, P_n\right)\right)\right\}_n \approx \left\{\mathcal{S}^{P_n}\left(1^\lambda, |P_n|\right)\right\}_n$$

Where $\mathcal{S}^{P_n}$ denotes that $\mathcal{S}$ has unrestricted oracle access to P_n. The number of queries is of course bounded by the running time of $\mathcal{S}$, making it polynomial in λ. $\diamondsuit$

Definition 2 is too strong, as shown by an adversary $\mathcal{A}\left(\tilde{P}_n \leftarrow \mathsf{Obf}\left(1^\lambda, P_n\right)\right) = \tilde{P}_n$. As $\mathcal{S}$ only has oracle access to P_n, it is impossible for $\mathcal{S}$ to output a program that is functionally equivalent on *all* (or even most) inputs.

Instead, Barak et al. [BGI+01] weaken the definition to that of simulating a predicate output by an adversary who is given an obfuscated program.

Definition 3 (Virtual Black Box) A probabilistic algorithm Obf is a VBB obfuscator if it is functionality preserving and $\forall$ PPT adversaries $\mathcal{A}$, $\exists$ PPT simulator $\mathcal{S}$ such that $\forall$ programs $\{P_n\}$ and security parameters $\lambda \in \mathbb{N}$:

$$\left|\Pr\left[\mathcal{A}\left(\mathsf{Obf}\left(1^\lambda, P_n\right)\right) = 1\right] - \Pr\left[\mathcal{S}^{P_n}\left(1^\lambda, |P_n|\right) = 1\right]\right| \leq \mathsf{negl}\left(\lambda\right)$$

Where $\mathcal{S}^{P_n}$ denotes that $\mathcal{S}$ has unrestricted oracle access to P_n. $\diamondsuit$

Unfortunately, Barak et al. show a simple counterexample to demonstrate that even this definition is too strong to achieve for Turing Machines in general.

2.1 Impossibility of VBB Obfuscation for Turing Machines

Consider a Turing Machine $P_{\alpha,\beta,\gamma}$ implementing the following functionality:

$$P_{\alpha,\beta,\gamma}(x) = \begin{cases} \beta & \text{if } x = \alpha \\ \gamma & \text{if } x(\alpha) = \beta \\ \perp & \text{otherwise} \end{cases}$$

If α, β, γ are uniformly random strings, we can make the following observations:

1. Oracle access to $P_{\alpha,\beta,\gamma}$ is highly unlikely to yield anything other than $\perp$ with polynomially many queries.
2. Given $\tilde{P}_{\alpha,\beta,\gamma} \leftarrow \mathsf{Obf}\left(1^\lambda, P_{\alpha,\beta,\gamma}\right)$, the functionality preserving property of Obf ensures that $\tilde{P}_{\alpha,\beta,\gamma}\left(\tilde{P}_{\alpha,\beta,\gamma}\right) = \gamma$.

Combining the above observations yields that $\mathcal{S}^{P_{\alpha,\beta,\gamma}}$ is almost never able to retreive γ, whereas $\mathcal{A}$ is able to do so given $\tilde{P}_{\alpha,\beta,\gamma}$. Any non-trivial predicate computed on γ will therefore not be simulatable with noticeable probability, proving the impossibility of VBB obfuscation for general TMs.

2.2 Impossibility of VBB Obfuscation for Circuits

Proving the impossibility of VBB Obfuscation for circuits takes some more effort, as there is no immediate method of feeding a circuit as input to itself. However, the existence of Fully Homomorphic Encryption implies that VBB Obfuscation is not possible for circuits in general, as demonstrated by the following counterexample.

Consider a circuit $C_{\alpha,\beta,\gamma}$ implementing the following functionality:

$$C_{\alpha,\beta,\gamma}(x) = \begin{cases} \mathsf{Enc}_{\mathsf{pk}}(\alpha) & \text{if } x = 0 \\ \beta & \text{if } x = \alpha \\ \gamma & \text{if } \mathsf{Dec}_{\mathsf{sk}}(x) = \beta \\ \perp & \text{otherwise} \end{cases}$$

Where $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ is a a Fully Homomorphic Encryption scheme, $\mathsf{pk}, \mathsf{sk} \leftarrow \mathsf{KeyGen}(1^\lambda)$, and α, β, γ are uniformly random.

Once more we can observe that oracle access to $C_{\alpha,\beta,\gamma}$ is unlikely to yield anything other than $\mathsf{Enc}_{\mathsf{pk}}(\alpha)$, as α itself is protected by semantic security of the FHE scheme and α, β, γ are uniform.

However, given $\tilde{C}_{\alpha,\beta,\gamma} \leftarrow \mathsf{Obf}\left(1^\lambda, C_{\alpha,\beta,\gamma}\right)$ we can obtain an encryption of β under pk by evaluating $\tilde{C}_{\alpha,\beta,\gamma}$ homomorphically on $\mathsf{Enc}_{\mathsf{pk}}(\alpha)$. As Obf preserves functionality of the obfuscated circuit, we have:

$$\mathsf{Enc}_{\mathsf{pk}}(\beta) = \mathsf{Eval}_{\mathsf{pk}}\left(\tilde{C}_{\alpha,\beta,\gamma}, \mathsf{Enc}_{\mathsf{pk}}\left(\alpha\right)\right)$$

If we feed the outcome of this evaluation to $\tilde{C}_{\alpha,\beta,\gamma}$ we activate the clause of $C_{\alpha,\beta,\gamma}$ which produces output γ, as follows:

$$\tilde{C}_{\alpha,\beta,\gamma}\left(\mathsf{Enc}_{\mathsf{pk}}(\beta)\right) \Rightarrow C_{\alpha,\beta,\gamma}(x) \text{ where } \mathsf{Dec}_{\mathsf{sk}}(x) = \beta \Rightarrow \text{output } \gamma$$

Therefore $\mathcal{A}\left(1^\lambda, \tilde{C}_{\alpha,\beta,\gamma}\right)$ is always able to obtain γ while $\mathcal{S}^{C_{\alpha,\beta,\gamma}}$ is not, implying that any non-trivial predicate computed on γ can not be simulated with only oracle access to $C_{\alpha,\beta,\gamma}$. This proves the impossibility of VBB Obfuscation for general circuits if homomorphic encryption exists.

The original proof by Barak et al. [BGI$^+$01] makes use of a type of homomorphic encryption scheme based on one-way functions, following which they show that that OWFs are implied by VBB Obfuscators. This contradiction proves unconditionally that VBB Obfuscators for circuits do not exist.

2.3 Indistinguishability Obfuscation

Seeing that general purpose obfuscation is impossible in the strong simulation sense, Barak et al. formalized a weaker notion of *indistinguishability obfuscation* (IO).

While the functionality preserving property remains the same as in Definition 1, the obfuscation property is changed to be indistinguishability based as follows:

Definition 4 (Indistinguishability Obfuscation) A uniform PPT algorithm Obf is an indistinguishability obfuscator if for all pairs of circuits C_n, C'_n such that $C_n(x) = C'_n(x)$ $\forall$ inputs x, for all security parameters $\lambda \in \mathbb{N}$, the following ensembles are computationally indistinguishable:

$$\left\{\mathsf{Obf}\left(1^\lambda, C_n\right)\right\}_n \approx \left\{\mathsf{Obf}\left(1^\lambda, C'_n\right)\right\}_n$$

$\diamondsuit$

Interestingly, unlike other cryptographic primitives the existence of IO does not imply that $P \neq NP$.

Lemma 1 *If $P = NP$, indistinguishability obfuscators exist.*

Proof: A simple IO construction is as follows: given C_n, output the smallest circuit that is functionally equivalent to C_n. This will trivially produce the same obfuscated circuit for all functionally equivalent circuits. $\square$

Corollary 2 *The existence of IO does not imply the existence of one-way functions.*

One-way functions and IO can therefore be seen as separate primitives. Interesting cryptographic constructions follow when we combine OWFs with IO. First, we define the notion of *witness encryption* and show how it can be instantiated using IO.

2.4 Witness Encryption

Witness Encryption was introduced and formalized by Garg et al. [GGSW13]. Let L be an NP language with relation R. ie. $\forall$ instances x, $R(x,w) = 1$ iff w is a witness for the statement $x \in L$. A witness encryption scheme consists of the following algorithms:

- $\mathsf{Enc}_{\mathsf{WE}}\left(1^\lambda, x, m\right)$: Given a message $m \in \{0,1\}$ and an instance x, the encryption algorithm outputs a ciphertext ct.
- $\mathsf{Dec}_{\mathsf{WE}}\left(w, \mathsf{ct}\right)$: Given a ciphertext ct and a witness w, the decryption algorithm outputs a bit.

A tuple of algorithms $(\mathsf{Enc}_{\mathsf{WE}}, \mathsf{Dec}_{\mathsf{WE}})$ constitutes a witness encryption scheme if the following properties are satisfied:

Definition 5 (Correctness) For all security parameters $\lambda \in \mathbb{N}$, messages $m \in \{0,1\}$, instances x and witnesses w,

$$\mathsf{Dec}_{\mathsf{WE}}\left(w, \mathsf{Enc}_{\mathsf{WE}}\left(1^\lambda, x, m\right)\right) = m \text{ if } R(x,w) = 1$$

$\diamondsuit$

Definition 6 (Soundness) For all instances $x \notin L$ and security parameters $\lambda \in \mathbb{N}$,

$$\left\{\mathsf{Enc}_{\mathsf{WE}}\left(1^\lambda, x, 0\right)\right\}_{x \notin L} \approx \left\{\mathsf{Enc}_{\mathsf{WE}}\left(1^\lambda, x, 1\right)\right\}_{x \notin L}$$

$\diamondsuit$

We can instantiate such a witness encryption scheme with an indistinguishability obfuscator Obf as follows.

- $\mathsf{Enc}_{\mathsf{WE}}\left(1^\lambda, x, m\right)$:
  1. Construct circuit $C_{x,m}(w) = \begin{cases} m & \text{if } R(x,w) = 1 \\ \perp & \text{otherwise} \end{cases}$
  2. Output $\mathsf{ct} = \mathsf{Obf}\left(1^\lambda, C_{x,m}\right)$
- $\mathsf{Dec}_{\mathsf{WE}}(w, \mathsf{ct})$: output $\mathsf{ct}(w)$

The above construction is clearly correct, as $\mathsf{ct}(w) = C_{x,m}(w) = m$ when $R(x,w) = 1$. Security follows from the indistinguishability of obfuscated circuits; when $x \notin L$ (ie. $\nexists w$ such that $R(x,w) = 1$) both $C_{x,0}$ and $C_{x,1}$ will output $\perp$ on all inputs, making them functionally equivalent. Therefore we have that:

$$\left\{\mathsf{Obf}\left(1^\lambda, C_{x,0}\right)\right\}_{x \notin L} \approx \left\{\mathsf{Obf}\left(1^\lambda, C_{x,1}\right)\right\}_{x \notin L}$$

Public-Key Encryption from WE and PRGs. IO and OWFs are sufficient to build PKE, as can be shown by means of the following public key encryption scheme described by Garg et al. [GGSW13]. The building blocks are a witness encryption scheme $(\mathsf{Enc}_{\mathsf{WE}}, \mathsf{Dec}_{\mathsf{WE}})$ and a PRG $G : \{0,1\}^\lambda \mapsto \{0,1\}^{2\lambda}$.

- $\mathsf{KeyGen}(1^\lambda)$: Sample $\mathsf{sk} \leftarrow \{0,1\}^\lambda$ and compute $\mathsf{pk} = G(s)$.
- $\mathsf{Enc}(\mathsf{pk}, m)$: Create an instance x such that $x \in L$ iff pk is in the range of G. Here, L is an NP-Complete language for which there exists a Karp-Levin reduction. Output $\mathsf{ct} = \mathsf{Enc}_{\mathsf{WE}}(1^\lambda, x, m)$.
- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$: Use sk to obtain a witness w for the statement $x \in L$. Output $\mathsf{Dec}_{\mathsf{WE}}(w, \mathsf{ct})$.

Security of the above scheme can be proven by a hybrid argument. If in the PKE security game the public key pk is replaced by a random string $r \leftarrow \{0,1\}^{2\lambda}$, the output of the adversary must not change non-negligibly (follows from PRG security). The probability that r is in the range of G is $\leq 2^{-\lambda}$, which when considered along with soundness of the witness encryption scheme implies that except with negligible probability $\mathsf{Enc}(r, 0) \approx \mathsf{Enc}(r, 1)$. Given that r can be used to replace pk, we have that $\mathsf{Enc}(\mathsf{pk}, 0) \approx \mathsf{Enc}(\mathsf{pk}, 1)$.

2.5 Special Purpose Obfuscators and IO

Consider a class of circuits $\{C_n\}$ for which there exists a special obfuscator SpObf. We can show that SpObf is made redundant by any indistinguishability obfuscator Obf as follows. Consider any 'special' obfuscation of a circuit, $\mathsf{SpObf}(1^\lambda, C_n)$. As SpObf and Obf both produce functionality preserving indistinguishable programs, we have that,

$$\left\{\mathsf{Obf}\left(1^\lambda, C_n\right)\right\} \approx \left\{\mathsf{Obf}\left(1^\lambda, \mathsf{SpObf}\left(1^\lambda, C_n\right)\right)\right\}$$

Note that C_n must be padded to be of the same length as $\mathsf{SpObf}\left(1^\lambda, C_n\right)$.

References

[BGI+01] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. In *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*, pages 1–18, 2001.

[GGSW13] Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. Witness encryption and its applications. In *Proceedings of the Forty-fifth Annual ACM Symposium on Theory of Computing*, STOC '13, pages 467–476, New York, NY, USA, 2013. ACM.