0368.4162 Foundations of Cryptography January 18, 2018

Lecture 13: Code Obfuscation

Lecturer: Nir Bitansky *Scribe: Mark Berlin*

1 Previously on Foundations of Crypto

We've seen that starting from standard computational assumptions, we can construct a host of advanced cryptographic primitives, starting from symmetric-key encryption, through public-key encryption, to more advanced primitives like fully-homomorphic encryption and identity-based encryption. In fact, we've even seen protocols that securely compute arbitrary functions.

2 What We Have Yet to Construct from Standard Assumptions

There are still quite a few goals in crypto we have yet to achieve based on standard assumptions. Functional encryption, which we've seen last week, is one of these examples. Let's mention a few other interesting examples:

Witness Encryption [GGSW13]. Say you're a mathematician who's been working your entire life on proving Goldbach's conjecture. Toward the end of your life, and having failed to prove it, you want to leave all your money to the person who can prove it. You would like to publish an encryption of the password to your safety deposit box, so that only someone who proves the conjecture can recover the password.

Witness encryption allows to do exactly that – you can generate an encryption $Enc_x(m)$ under an NP statement x, so that given a corresponding witness w, one can recover m using the witness as the decryption key. However, if the statement is false (or just indistinguishable from being false), then m should remain hidden (formally, $\forall m_0, m_1 : Enc_x(m_0) \approx_c Enc_x(m_1)$).

Time-Lock Puzzles [RSW00]. Imagine that you would like to post your secret diary, but only 20 years after your death. Could you encrypt to the future? Time-lock puzzles allow encrypting a message m to be opened after some prescribed time interval t. That is, there's a public decryption process that can recover m in t steps, but algorithms running in (even parallel) time $\ll t$ cannot learn anything about m.

Deniable Encryption [CDNO97]. Imagine that one day a tyrant rises to power in your country and starts tapping all of your communication with foreign citizens. At any point, they may demand that you reveal the contents of your encrypted messages $ct = Enc_{pk}(m)$ by providing the randomness r used to perform the encryption. In deniable encryption, for any ciphertext ct and any message m', you can produce a fake randomness r' such that $ct = Enc_{pk}(m'; r')$, thereby denying the message m if you choose to.

3 Code Obfuscation

Today we'll talk about a cryptographic primitive that seems to suggest a solution to all of the above problems (and others that we did not mention) known as *code obfuscation*. We've briefly discussed this notion in the context of public-key encryption as the digital analogue of a magic black box in which you can lock a program Π and send it, so that anyone in possession of this box can use it to feed inputs for Π and receive back the outputs, but does not learn anything else regarding Π itself.

We will now formally define this notion using the simulation paradigm. We will restrict attention to the case where programs are represented by circuits.

Definition 3.1 (Virtual Black-Box Obfuscation)**.** *A PPT $\mathcal{O}$ is a VBB obfuscator for a class $\mathcal{C} = \{\mathcal{C}_n\}$ of circuits of polynomial-size $n^{O(1)}$, if it satisfies the following requirements:*

1. **Syntax:** *for any circuit C, $\mathcal{O}(C)$ outputs a circuit.*
2. **Polynomial Slowdown:** *there exists a polynomial q such that for any $C \in \mathcal{C}_n$,*
$$|\mathcal{O}(C)| \leq q(|C|) \ .$$
3. **Functionality:** *for any $C \in \mathcal{C}_n$ and $x \in \{0,1\}^n$,*
$$\mathcal{O}(C)(x) = C(x) \ .$$
4. **Virtual Black-Box:** *For any n.u. PPT $A = \{A_n\}$, there exists a n.u. PPT simulator $S = \{S_n\}$ and a negligible function μ, such that for any function $f : \mathcal{C}_n \to \{0,1\}^*$, and any $C \in \mathcal{C}_n$,*
$$\Pr\left[A_n(\mathcal{O}(C)) = f(C)\right] \leq \Pr\left[S_n^C = f(C)\right] + \mu(n) \ .$$

This definition says that whatever the adversary can learn about the original circuit from the obfuscation, it could also have learned by simply running the circuit as a black box.

Theorem 3.2 (Informal)**.** *VBB obfuscation implies all primitives mentioned throughout this course (including those mentioned above).*

Example: Witness Encryption. Let's see for example how to construct witness encryption from VBB obfuscation. Let R be an NP relation corresponding to a language L. Then given $x \in \{0,1\}^n$, we can encrypt m under x by returning an obfuscation of the circuit (function) $R_{x,m}$ that is defined as follows:

$$R_{x,m}(w) = \begin{cases} m & \text{if } R(x,w) = 1 \\ \perp & \text{otherwise} \end{cases} \ .$$

Note that for any no-instance $x \in \{0,1\}^n \setminus L$, $R_{x,m} \equiv \perp$; in particular, while its code may depend on m, the function it computes does not. A bit more formally, for any efficient A, there exists an efficient simulator S such that for any (equal-length) messages m_0, m_1 and $x \in \{0,1\}^n \setminus L$:

$$\Pr_b\left[A(\mathcal{O}(R_{x,m_b})) = b\right] \leq \Pr_b\left[S^{R_{x,m_b}} = b\right] + n^{-\omega(1)} = \Pr_b\left[S^{\perp} = b\right] + n^{-\omega(1)} = \frac{1}{2} + n^{-\omega(1)} \ .$$

The Bad News: VBB is impossible. It turns out that this notion is just too good to be true, regardless of computational assumptions (this is essentially the first principal impossibility we've encountered in this course).

Theorem 3.3 ([BGI+01])**.** *There exist circuit classes $\mathcal{C}$ that cannot be VBB obfuscated.*

We'll prove here a weaker theorem that shows the above assuming fully-homomorphic encryption (FHE) exists. This gap can be rather easily bridged by showing that VBB implies FHE.

Proof (Assuming FHE). What does it mean that a given circuit class is unobfuscatable? We will show that there is a circuit class $\mathcal{C} = \{\mathcal{C}_n\}$ and functions $f : \mathcal{C}_n \to \{0,1\}^*$ such that for any $C \in \mathcal{C}_n$ it is possible to efficiently learn $f(C)$ from any circuit $\tilde{C}$ that computes the same function as C. On the other hand, we'll show that any efficient simulator S, will fail to learn $f(C)$ given C as a black-box oracle, for almost any $C \in \mathcal{C}_n$. (Do you see why this implies that $\mathcal{C}_n$ cannot be VBB obfuscated?)

We will now define the desired circuit class. Let $(E, D, Eval)$ be a (secret key) FHE scheme, and let a circuit $C_{a,b,sk,r}$, parameterized by $a, b, sk, r \in \{0,1\}^n$, be defined as follows: [1]

$$C_{a,b,sk,r}(x) = \begin{cases} b & \text{if } x = a \\ E_{sk}(a;r) & \text{if } x = 0^n \\ a & \text{if } D_{sk}(x) = b \\ \perp & \text{otherwise} \end{cases} .$$

The proof now relies on the following two claims:

Claim 3.4. *There exists a poly-time extraction procedure A that outputs a for any a, b, sk, r and any circuit $\tilde{C} \equiv C_{a,b,sk,r}$.*

Claim 3.5. *For any n.u. PPT S there exists a negligible μ such that*

$$\Pr_{a,b,sk,r}\left[S^{C_{a,b,sk,r}} = a\right] \leq \mu(n) .$$

Let's start by proving the first claim.

A acts as follows:

1. Runs $\tilde{C}(0^n)$ to obtain $ct = E_{sk}(a)$.
2. Homomorphically computes an encryption of b: $\tilde{ct} = Eval(ct, \tilde{C})$.
3. Runs $\tilde{C}(\tilde{ct})$ to obtain a.

We'll now prove the second claim. Assume on the contrary that for some efficient S and $\varepsilon = n^{-O(1)}$:

$$\Pr_{a,b,sk,r}\left[S^{C_{a,b,sk,r}} = a\right] \geq \varepsilon .$$

We can assume w.l.o.g that whenever S outputs a it makes the query a before (Otherwise, we'll consider a new S' that before producing and outputting a value v specifically makes a query v). Let us denote by Q the first query $Q \neq 0^n$ that S makes and that is not answered with $\perp$ (per our assumption we know that whenever S outputs a, there's at least one such query).

First, note that Q decrypts to b with probability at most $2^{-n} < \varepsilon/2$ (do you see why?). This implies that $Q = a$ with probability at least $\varepsilon/2$ (note that these are the only two options for $Q \neq 0^n$ which is not answered with $\perp$). We now show that we can get an adversary B that breaks the CPA security of the FHE in the following sense:

$$\Pr_{a,sk,r}\left[B(Enc_{sk}(a;r)) = a\right] \geq \varepsilon/2t ,$$

where t is a bound on the number of queries that S makes.

$B(ct)$ guesses the query number $i \in [t]$ where Q is performed. It answers 0^n with ct, and any other query before the ith query with $\perp$. Then we know that the ith query will be a with probability at least $\varepsilon/2t$.

This concludes the proof. □

[1] The circuit formally takes inputs that are as large as ciphertexts encrypting n bits, which will be of some polynomial size. So $x = a$ can be interpreted as a is a prefix of x.

4 Indistinguishability Obfuscation

This is not the end of the story for obfuscation. The same paper that proved the above impossibility result also suggested the following relaxation.

Definition 4.1 (Indistinguishability Obfuscation)**.** *A PPT $\mathcal{O}$ is an IO for a class $\mathcal{C} = \{\mathcal{C}_n\}$ of circuits of polynomial-size $n^{O(1)}$, if it satisfies the following requirements:*

1. **Syntax, Polynomial Slowdown, Functionality:** *same as VBB.*
2. **Indistinguishability:**

$$\{\mathcal{O}(C_0)\}_{C_0,C_1} \approx_c \{\mathcal{O}(C_1)\}_{C_0,C_1} \ \text{, where } C_0 \equiv C_1, |C_0| = |C_1| \ .$$

This is definition is hardly as intuitive as VBB. Let's develop some intuition about it. First, note that the impossibility proof doesn't apply here – the extractor A we constructed would indeed act the same for any two implementations of the function $C_{a,b,sk,r}$. This notion is not known to suffer from any barriers. In fact,

Claim 4.2. *If $P = NP$, then there exists IO (for any circuit class).*

But what does IO protect? So one thing we can say about IO is that it protects *whatever can be protected*. For example, if there exists a VBB obfuscator for some class $\mathcal{C}$, then an IO for all circuits implies a VBB obfuscator for $\mathcal{C}$.[2] Specifically, assume the VBB obfuscator blows up the circuits by a poly q, then by taking any $C \in \mathcal{C}$, padding it to size $q(|C|)$, and applying IO, we get a VBB obfuscator.

This still doesn't say that IO is useful: indeed, figuring out what can be protected and what not seems hard in general, and for a long time, this notion was indeed cast aside. This changed in 2013 (basically by accident), as a group of researchers that were trying to construct functional encryption ended up with a candidate for IO, based on a generalization of bilinear maps known as *multilinear maps* [GGH$^+$13b].

This gave renewed motivation to understanding whether this notion could somehow be useful. The answer turned out to be "YES, big time".

Theorem 4.3 (Informal)**.** *IO (+NP hardness) implies almost all primitives mentioned throughout this course (including those mentioned in the beginning of this lecture).*

The almost is due to some unexpected gaps – for instance, IO isn't known to imply collision-resistant hashing or trapdoor permutations on $\{0,1\}^n \to \{0,1\}^n$ (and in fact, some barriers have been shown here).

Example: Public-Key Encryption. As an example, and to get some intuition on how IO can be useful, we'll construct PKE. The scheme works as follows:

- The secret key sk is a random seed $s \leftarrow \{0,1\}^n$.
- The public key pk is $PRG(s)$ for a length-doubling PRG.
- An encryption $E_{pk}(m)$ is an obfuscation of the following circuit.

$$C_{pk,m}(x) = \begin{cases} m & \text{if } PRG(x) = pk \\ \perp & \text{otherwise} \end{cases} \ .$$

- To decrypt a ciphertext give by a circuit $\tilde{C}$, return $\tilde{C}(s)$

[2] In fact, we don't need IO for all circuits, but only for some closure of $\mathcal{C}$ that includes functionally equivalent circuits up to some size.

To prove security, we would like to argue that, when obfuscated, the program $C_{pk,m}$ is indistinguishable from a program that always returns $\perp$ and is independent of m. However, $C_{pk,m} \not\equiv \perp$ and so we can use IO directly. The point is that $C_{pk,m}$ is not $\perp$ only on some hard-to-find point, so there's still a chance. Specifically, we can first consider generated an indistinguishable fake public key $\tilde{pk} \leftarrow \{0,1\}^{2n}$. Due to pseudorandomness, $pk, E_{pk}(m) \approx_c \tilde{pk}, E_{\tilde{pk}}(m)$. Now, notice that in this new (fake scheme) $C_{\tilde{pk},m} \equiv \perp$ with overwhelming probability (over the choice of $\tilde{pk}$), and thus we can apply IO and finish the security proof.

Note that the above construction can be re-framed as two steps. First, using IO to construct witness encryption, and then using witness-encryption to construct PKE.

5 A Big Open Question

One of the central questions in crypto today is whether IO can be constructed from standard assumptions. This journey has started in 2013 with the mentioned candidate, hasn't ended yet, and has so far been quite bumpy.

Main Enabler: Multilinear Maps. Roughly speaking m-linear maps are groups encodings that come with a map

$$(g^{x_1}, \ldots, g^{x_m}) \stackrel{e}{\longmapsto} h^{x_1 \cdots x_m} \ ,$$

where the bilinear maps we've seen last week are a special case with $m = 2$.

Unlike the case $m = 2$, where such maps are considered standard, constructing such maps for $m > 2$ was a long standing open problem, and first candidates have only appeared since 2013 [GGH13a]. In particular, these candidates only approximate the above notion, and the hardness of corresponding problems is not well understood yet. In particular, natural problems like m-linear Diffie-Hellman have turned out not to be hard in current candidate.

The efforts toward constructing IO can generally be split to several directions (with overlaps):

1. Understand the hardness of current (approximate) multilinear maps and base IO on it.
2. Construct new candidates for multilinear maps.
3. Attempt to reduce to required degree of multilinear maps to two (which is already standard), or even beyond.

Last lecture we mentioned a result that allowed some progress along the third vein:

Theorem 5.1 (Informal (and Inaccurate) [AJ15, BV15])**.** *Assume functional encryption where either:*

- *the complexity of encryption is independent of $s = |f|$ (or even just grows with $|f|^{1-\varepsilon}$).*
- *the adversary can ask for any polynomial number of functional keys.*

Then there exists indistinguishability obfuscation.

State-of-the-art IO constructions use this result as a building block to construct IO from 3-linear maps (and a plausible assumption on low-depth PRGs). This seems tantalizingly close to basing IO on standard assumptions (i.e., 2-linear maps), but we don't really know. It may very well be that there's a substantial gap between 2-linear and 3-linear maps, which cannot be bridged under standard assumptions. Only time would tell.

References

[AJ15] Prabhanjan Ananth and Abhishek Jain. Indistinguishability obfuscation from compact functional encryption. In *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part I*, pages 308–326, 2015.

[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.

[BV15] Nir Bitansky and Vinod Vaikuntanathan. Indistinguishability obfuscation from functional encryption. In *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 171–190, 2015.

[CDNO97] Ran Canetti, Cynthia Dwork, Moni Naor, and Rafail Ostrovsky. Deniable encryption. In *Advances in Cryptology - CRYPTO '97, 17th Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 1997, Proceedings*, pages 90–104, 1997.

[GGH13a] Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. In *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*, pages 1–17, 2013.

[GGH+13b] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 40–49, 2013.

[GGSW13] Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. Witness encryption and its applications. In *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 467–476, 2013.

[RSW00] Ronald L. Rivest, Adi Shamir, and David A. Wagner. Time-lock puzzles and timed-release crypto. Technical Report MIT/LCS/TR-684, MIT, February 2000.