Proof of $\exists$ OWF $\not\Rightarrow$ $\exists$ salted CR hash via compression argument

(in a blackbox way)

Def: A poly-time computable hash function $h : \underbrace{\{0,1\}^*}_{n \text{ bits}} \times \{0,1\}^* \to \underbrace{\{0,1\}^*}_{n \text{ bits}}$ is (salted) CR if $\forall$ PPT $\mathcal{A}$:

$$\Pr_{s \leftarrow \$ \{0,1\}^n}\left[(x, x') \leftarrow \$ \mathcal{A}(1^n, s) : h(s,x) = h(s,x'),\ x \neq x' \right] = \text{negl.}(n)$$

Lemma 1

Description of a random function $f : \{0,1\}^n \to \{0,1\}^n$ requires $n \cdot 2^n - \log^2 n$ bits with prob. $1 - \text{negl}(n)$ (prob. over the choice of the random function F)

proof

Let $\mathcal{F}$ be the set of all functions $f : \{0,1\}^n \to \{0,1\}^n$

Let $\text{enc} : \mathcal{F} \to \{0,1\}^*$ be an injective function.

Want: $\forall m \in \mathbb{Z}_+$

$$\Pr_{f \leftarrow \$ \mathcal{F}}\left[|\text{enc}(f)| \geq m \right] \geq 1 - \frac{2^m}{|\mathcal{F}|}$$

since now, pick $m = n \cdot 2^n - \log^2 n$ and get:

$$\Pr\left[|\text{enc}(f)| \geq n \cdot 2^n - \log^2 n \right] \geq 1 - \frac{2^{n \cdot 2^n - \log^2 n}}{|\mathcal{F}|} = 1 - \frac{2^{n \cdot 2^n - \log^2 n}}{(2^n)^{2^n}} = 1 - \frac{2^{n \cdot 2^n - \log^2 n}}{2^{n \cdot 2^n}} = 1 - 2^{-\log^2 n}$$

$$2^{-\log^2 n} = (\underbrace{2^{-\log n}}_{\frac{1}{n}})^{\log n} = \frac{1}{n^{\log n}} = \text{negl.}(n)$$

How to get this?

Let $S \subseteq \mathcal{F}$ s.t. $\forall f \in S : |\text{enc}(f)| < m$.

Now, there are $\underbrace{2^0}_{\#\text{strings of len } 0} + \underbrace{2^1}_{\#\text{strings of len } 1} + \underbrace{2^2}_{\#\text{strings of len } 2} + \ldots + \underbrace{2^{m-1}}_{\#\text{strings of len } m-1} = 2^m - 1$ strings of length $< m$.

($2^m - 1$: this is the biggest number that can be written using m bits using binary representation: $2^0 \cdot b_0 + 2^1 \cdot b_1 + \ldots + 2^{m-1} \cdot b_{m-1}$ — biggest, when these are all ones. $111_2 = 2^0 \cdot 1 + 2^1 \cdot 1 + 2^2 \cdot 1 = 1 + 2 + 4 = 7 = 8 - 1$, $8 = 2^3$. $\Rightarrow$ next number would be $2^0 \cdot 0 + 2^1 \cdot 0 + \ldots + 2^{m-1} \cdot 0 + 2^m \cdot 1 = 2^m$ $\Rightarrow 2^m - 1$)

Since enc is inj., $|S| \leq 2^m - 1$

$$\Pr_{f \leftarrow \$ \mathcal{F}}\left[|\text{enc}(f)| < m \right] = \Pr[f \in S] = \frac{|S|}{|\mathcal{F}|} \leq \frac{2^m - 1}{|\mathcal{F}|} < \frac{2^m}{|\mathcal{F}|}$$

□

Claim 1 $\exists$ OWF $\not\Rightarrow$ $\exists$ salted CR h

You cannot prove, in a relativizing blackbox way, that

$\exists$ OWF $\Rightarrow$ $\exists$ salted CR h

proof

Suppose we have the oracles:

- f = random function $\{0,1\}^n \to \{0,1\}^n$
- PSPACE — i.e. an oracle that can solve any problem solvable in polynomial (w.r.t. problem size) space
  - why? – in poly space one can exhaustively find preimage of OWF $\Rightarrow$ f is now the only OWF (because PSPACE does not have access to f)

ideal SAM used in lec 2 by Chris:

```
SAM(s, h^f(·,·))
    assert |s| = n AND |·| = 2n
    z ←$ {0,1}^2n
    y ← h^f(s, z)
    z' ←$ {w | h(s,w) = y}
    return (z, z')
```

we can't use randomness in the (coming) compression argument

→ get around that: $\forall h, s, i \in \mathbb{N}$ let $R^i_{h,s} : \{0,1\}^{2n} \to \{0,1\}^{2n}$ be a rnd function s.t. $R^i_{h,s}$ is permutation if i is odd

Now:

Mem = 0 // internal memory of Sam

$\text{SAM}_{det}(s, h(\cdot, \cdot))$

Same behaviour if you don't know $R'_{h,s}$

assert $|s| = n$ AND $|\,| = 2n$
$z \leftarrow R^{Mem}_{h,s}(0^{2n})$
$y \leftarrow h(s,z)$
$Mem \leftarrow Mem + 1$
$z' \leftarrow w$ s.t. $R^{Mem}_{h,s}(w)$ is lexicographically smallest value s.t. $h(s,w) = y$
$Mem \leftarrow Mem + 1$
return (z, z')

Claim 2 relative to the oracles, $\nexists$ salted CR h

proof:

consider the following adversary

$\mathcal{B}(salt)$
$(x,x') \leftarrow SAM_{det}(salt, h)$
return (x,x')

Now $\Pr_{\substack{s \leftarrow \{0,1\}^n \\ R'_{h,s}}}[(x,x') \leftarrow \$ \mathcal{B}(1^n, s) : h(s,x) = h(s,x'), x \neq x']$

$= \Pr[SAM_{det} \text{ returns } (z,z') \text{ s.t. } z \neq z']$

How likely is this?
2^{2n} options for z
2^n options for y at most
$\Rightarrow$ on average $\frac{2^{2n}}{2^n} = 2^n$ many z' of length $2n$ s.t. $h(s,z') = y$
$\Rightarrow$ with prob. $\geq \frac{1}{2}$ there are $\geq 2^n$ options for z'.
$\Rightarrow$ the prob. that $z = z' \leq \frac{1}{2^n}$

$\geq \frac{1}{2}\left(1 - \frac{1}{2^n}\right) = $ non-negl. □

$\Rightarrow$ **Claim 1** □

Claim 3 relative to the oracles, f is a OWF

proof:

assume, for contradiction, that $\exists$ PPT $\mathcal{A}_r^{SAM_{det}, f, PSPACE}$ s.t.

$\Pr_{\substack{x \leftarrow \$ \{0,1\}^n \\ r_A \leftarrow \$ \{0,1\}^{r(n)} \\ f \leftarrow \$ \mathcal{F}}}[\mathcal{A}_r(f(x)) \in f^{-1}(f(x))] = \text{non-negl.}(n) \geq \frac{1}{p(n)}$ ← polynomial

Now, we use averaging argument to get deterministic $\mathcal{A}$:

Claim: $\exists r_A : \Pr_{\substack{x \leftarrow \$ \{0,1\}^n \\ f \leftarrow \$ \mathcal{F}}}[\mathcal{A}_r(f(x)) \in f^{-1}(f(x)) \mid r_A] \geq \frac{1}{p(n)}$

proof: Assume, for contradiction, that $\forall r_A$

$\Pr_{\substack{x \leftarrow \$ \{0,1\}^n \\ f \leftarrow \$ \mathcal{F}}}[\mathcal{A}_r(f(x)) \in f^{-1}(f(x)) \mid r_A] < \frac{1}{p(n)}$

Now

$\Pr_{\substack{x \leftarrow \$ \{0,1\}^n \\ r_A \leftarrow \$ \{0,1\}^{r(n)} \\ f \leftarrow \$ \mathcal{F}}}[\mathcal{A}_r(f(x)) \in f^{-1}(f(x))] = \sum_{r_A} \underbrace{\Pr_{x, r_A, f}[\mathcal{A}_r(f(x)) \in f^{-1}(f(x)) \text{ and } r_A]}_{\Pr_{x,f}[\mathcal{A}_r(f(x)) \in f^{-1}(f(x)) \mid r_A] \cdot \Pr_{r_A}[r_A]}$

$< \sum_{r_A} \frac{1}{p(n)} \cdot \frac{1}{2^{r(n)}}$

$= \frac{1}{p(n)}$ ↯ □

$\Rightarrow \exists$ deterministic poly-time $\mathcal{A}$ (namely $\mathcal{A}_r$ with fixed r_A):

$\Pr_{x,f}[\underbrace{\mathcal{A}(f(x)) \in f^{-1}(f(x))}_{E}] \geq \frac{1}{p(n)}$

$\Downarrow$

$\Pr_f\left[\Pr_x[\underbrace{\mathcal{A}(f(x)) \in f^{-1}(f(x))}_{E}] \geq \underbrace{\frac{1}{2p(n)}}_{p'}\right] = $ non-negl.

Proof using averaging arg.:

Assume for contradiction that $\Pr_{f \leftarrow \$ \mathcal{F}}[\overbrace{\Pr_x[E] \geq \frac{1}{2p}}^{E'}] = $ negl.

Now $\Pr_{x,f}[E] = \sum_f \Pr_{x,f}[E \text{ and } f] = \sum_{f \in \mathcal{F}} \Pr_x[E \mid f] \cdot \Pr_f[f]$

(sum of prob. of disjoint events)

$= \underbrace{\sum_{f \text{ s.t. } E'}}_{\text{negl.} \cdot |\mathcal{F}|} \underbrace{\Pr_x[E \mid f]}_{\leq 1} \cdot \Pr_f[f] + \underbrace{\sum_{f \text{ s.t. } E' \text{ does not happen}}}_{|\mathcal{F}|} \underbrace{\Pr_x[E \mid f]}_{< \frac{1}{2p}} \cdot \Pr_f[f]$ ($\Pr_f[f] = \frac{1}{|\mathcal{F}|}$)

$< \text{negl.} |\mathcal{F}| \cdot 1 \cdot \frac{1}{|\mathcal{F}|} + |\mathcal{F}| \cdot \frac{1}{2p} \cdot \frac{1}{|\mathcal{F}|}$

$= \underbrace{\text{negl.}}_{< \frac{1}{2p}} + \frac{1}{2p} = \frac{1}{p}$ ↯ □

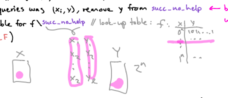
fix one such f

Now, the output out of the following (inefficient) encoder

```
enc^{f,SAM_det}(A):
succ ← {y ∈ {0,1}^n | ∃x : f(x)=y and A(y)=x}   // y that A succesfully inverts
succ_no_help ← ∅   // y that A "knew without making queries"
for y in succ   // iterate over elems in succ in lexicographic order
    append y to succ_no_help
    emulate A(y):
        if A makes an f(x)-query for some x
            if f(x) = y   // hit!
                stop emulation
            if f(x) ∈ succ
                remove f(x) from succ
        if A makes SAM_det(salt, h) query
            if SAM_det(salt, h) returns (x, x')
                remove all f-queries that h^f(salt,x') and h^f(salt,x) make from succ
                if one of the removed queries was (x_i, y), remove y from succ_no_help
rest_of_F ← the y-half of the look-up table for f \ succ_no_help   // look-up table: f:
out ← ([succ_no_help] as 2 numbers, rest_of_F)
return out
```

the for-loop will not cover f(x), if f(x) was not yet covered

← by lec 2, this happens with negl. prob.
$\Rightarrow$ at most $2^n \cdot$ negl. elems are removed from succ_no_help



contains the full information on f.

Why? Because the following (inefficient) decoder can output the full look-up table of f (even though it only has access to output of enc, and $\mathcal{A}$ and $R_{h,s}$'s (and they don't depend on f), not oracles!)

```
dec(out, A):
([succ_no_help] as 2 numbers, rest_of_F) ← out
full_F ← full look-up table for the rest_of_F
for y in succ_no_help   // in lexicographic order
    emulate A(y):
        if A makes a f(x) query
```

this is called a **compression argument**
because the point of the proof is that
- if $\mathcal{A}$ can invert f
- then $\mathcal{A}$ must also be able to compress the description of f
- which is not possible,

if $\mathcal{A}$ makes a f(x) query
    if $x \in$ full_F
        (easy to emulate)
    else
        stop emulation
        add (x, y) to full_F
    if $\mathcal{A}$ makes $SAM_{det}(salt, h)$ query
        (easy to emulate) // no hits (because hits were removed from succ_no_help)
if we did not stop emulation and emulation returned x
    add (x, y) to full_F
return full_F

since f requires long description (by Lemma 1)

$\binom{n}{k} = \frac{n!}{k!(n-k)!} \leq \frac{n^k}{k!}$

However, output out of enc is shorter than $n \cdot 2^n - \log^2 n$

out = ([succ_no_help] as 2 numbers, rest_of_F)

$2 \cdot \log \binom{2^n}{\frac{2^n}{2p'q2t}}$

in succ we have $\frac{1}{p'} \cdot 2^n$ many x s.t. $\mathcal{A}$ inverts f(x)
Suppose that $\mathcal{A}$ makes no SAM_{det}-hits
$\Rightarrow$ at each emulation we remove at most $q \cdot 2 \cdot t$ (q poly, t poly)
in the worst case we always remove the max amount
• how many times k we have to remove, until succ has no elems left?

$k \cdot q \cdot 2 \cdot t = \frac{2^n}{p'} \Leftrightarrow k = \frac{2^n}{p' \cdot q \cdot 2 \cdot t} - 2^n \cdot \text{negl.}$ ($\leq \frac{1}{2p'q2t}$)

$|\text{succ\_no\_help}| \geq \frac{2^n}{p'q2t} - \frac{2^n}{2p'q2t} = \frac{2^n}{2p'q2t}$

rest_of_F $\leq n\left(2^n - \frac{2^n}{2p'q2t}\right)$

$|out| = n\left(2^n - \frac{2^n}{2p'q2t}\right) + 2 \cdot \log \binom{2^n}{\frac{2^n}{2p'q2t}} \overset{?}{<} n \cdot 2^n - \log^2 n$

$\binom{2^n}{\frac{2^n}{2p'q2t}} \leq \frac{(2^n)^{\frac{2^n}{2p'q2t}}}{\frac{2^n}{2p'q2t}!}$

$\log^2 n + 2 \log \frac{2^{n \frac{2^n}{2p'q2t}}}{\frac{2^n}{2p'q2t}!} < n \cdot \frac{2^n}{2p'q2t}$ (with $2p'q2t =: \vartheta$)

Which is a contradiction, since by **Lemma 1**, we know that w.h.p. f (all but negl. fraction of f) requires $n \cdot 2^n - \log^2 n$ bits to describe. □

$\log^2 n + 2 \log \frac{2^{n \cdot \frac{2^n}{\vartheta}}}{\frac{2^n}{\vartheta}!} < n \cdot \frac{2^n}{\vartheta}$

$\log^2 n + 2 n \cdot \frac{2^n}{\vartheta} < n \cdot \frac{2^n}{\vartheta} + 2 \log \frac{2^n}{\vartheta}!$

$2 \log \frac{2^n}{\vartheta}! \geq 2 \log \frac{\left(\frac{2^n}{\vartheta}\right)^{\frac{2^n}{\vartheta}}}{e^{\frac{2^n}{\vartheta}}}$

$= 2 \log \left(\frac{2^n}{\vartheta}\right)^{\frac{2^n}{\vartheta}} - 2 \log e^{\frac{2^n}{\vartheta}}$

$= 2 \frac{2^n}{\vartheta} \log \left(\frac{2^n}{\vartheta}\right) - 2 \frac{2^n}{\vartheta} \log e$

Thm:
$\forall n \in \mathbb{N}: n! \geq \frac{n^n}{e^n}$

proof:
$e^x = 1 + \frac{x^1}{1!} + \frac{x^2}{2!} + \ldots \quad \forall x$
$e^x \geq \frac{x^n}{n!} \quad \forall n, x$
in particular $x = n$:
$e^n \geq \frac{n^n}{n!}$
$n! \geq \frac{n^n}{e^n}$ □

$2 \frac{2^n}{\vartheta} \log e + \log^2 n + n \cdot \frac{2^n}{\vartheta} < 2 \frac{2^n}{\vartheta} \log \left(\frac{2^n}{\vartheta}\right)$

$= 2 \frac{2^n}{\vartheta} \log 2^n - 2 \frac{2^n}{\vartheta} \log \vartheta$

$2 \frac{2^n}{\vartheta} \log \vartheta + 2 \frac{2^n}{\vartheta} \log e + \log^2 n + n \cdot \frac{2^n}{\vartheta} < 2 \frac{2^n}{\vartheta} n$

$\log \vartheta \leq \log n^c = c \cdot \log n$

✓ :)