ICS 180: Introduction to Cryptography 5/11/2004

Solutions to homework 3

1 Authentication Scheme from One-Way Permutations

Let PPT algorithms $(Gen, Sample, Eval)$ define a OWF (or OWP) $\{f_i\}_{i\in\mathcal{I}}$. Suppose that players U and B use the following authentication scheme. For example, say that B is a bank's web portal and C is a web applet run by the bank's client. The scheme is designed to last for one year, and needs to be reinitialized after that:

- **Initialization Protocol:** Let $n = 365$. B runs $Gen(1^\tau)$ to pick a one-way function f_i with security parameter τ and runs $Sample(i)$ to pick a random element $x^{(n)}$ in the domain D_i of f_i. Then B computes, for k going from n down to 1, values $x^{(k-1)} = f_i(x^{(k)}) = Eval(i, x^{(k)})$. (You'll see in a second why we are computing them backward rather than forward.) B keeps for himself $x^{(0)}$ as the "verification value" for C, and gives to C (over some secure channel) the "root authentication secret" $x^{(365)}$. C then re-generates all the $x^{(k)}$ values for $k = 0, ..., 364$ by consecutive applications of f_i. Let's denote k-times repeated application of f_i as a function $(f_i)^{(k)} : D_i \to \{0,1\}^*$. With this notation we have $x^{(n-k)} = (f_i)^{(k)}(x^{(n)})$ for every k.
- **Authentication Protocol:** To authenticate himself to B on day t, C sends to B value $x = x^{(t)}$ and announces that he is "C". B then picks the yesterday's verification value $x^{(t-1)}$ for that client, and authenticates this client as indeed "C" if $f_i(x) = x^{(t-1)}$. If the equation holds B stores x as $x^{(t)}$. (It's easy to generalize this to the case when C contacted B last on any day $t' < t$: Just compute $(f_i)^{(t-t')}$ on $x^{(t)}$ and compare with $x^{(t')}$.)

Assume that the adversary E, who tries to authenticate himself as "C" to B too, can *eavesdrop* on all instances of the (C, B) authentication protocol but cannot interrupt any such instance. On the other hand E can *initiate* an instance of the authentication protocol with B himself and try to make B authenticate him as "C".

1.1 [25 points]

Prove that if the function collection $\{f_i\}$ defined by $(Gen, Sample, Eval)$ is a One Way *Permutation* collection then the above authentication protocol is secure against the eavesdropping adversary E in the following sense: Show that if there exists a PPT E which, after listening to some number $k \in [1, n]$ of authentication sessions (C, B), has a non-negligible chance of being authenticated by B as "C" on a session that E initializes, then you can use such adversary E to create a PPT algorithm A which has a non negligible advantage in an attack against one-wayness of the OWP collection f_i. In other words, algorithm A should succeed with non negligible probability in inverting permutation f_i on value $y = f_i(x)$ for a random $x \in D_i$. This will show that if the above authentication protocol is insecure against eavesdroppers (i.e. there exists a PPT attacker E which cheats the scheme with a significant

enough probability) then the function collection $\{f_i\}$ cannot be a OWP collection. Therefore, by counterpositive, if $\{f_i\}$ *is* a OWP collection then this is a secure authentication scheme. Note that in this way you showed that one can build a provably secure authentication scheme (but against eavesdroppers only!) based on either the DL assumption or the RSA assumption (because either of these assumptions implies a OWP family $\{f_i\}$).

Solution: Assume that there exists an eavesdropping adversary E, which is a PPT algorithm and which succeeds in the attack defined above, with non-negligible probability ϵ. Let k be the number of authentication sessions E listens to before initializing his own session with B.

First question you should ask is: What does the adversary E see? I.e., what are the *inputs* to E? He sees k instances of the authentication protocol, which in our case is simply k numbers $x^{(1)}$, ..., $x^{(k)}$, where for every $j = 1, ..., k-1$, we have $x^{(j)} = f_i(x^{(j+1)})$. Now you should ask yourself: What is it that E *outputs*? We know that with probability ϵ, E succeeds, on day $k+1$, in authenticating himself to B as "C" with probability ϵ. Therefore, with probability ϵ, E must be outputing value x s.t. $f_i(x) = x^{(k)}$.

Let's denote $x^{(k)}$ as y. Let's first see what *distribution* does y come from. I claim that it is simply a randomly chosen number in the domain D_i of f_i. That's because $y = x^{(k)} = (f_i)^{(365-k)}(x^{(365)})$, and first of all, $x^{(365)}$ was picked at random in D_i, and second, since f_i is a permutation on D_i, then function $(f_i)^{(365-k)}$ is also a permuation on D_i, and hence picking its argument, $x^{(365)}$, at random in D_i implies that the function value, $y = x^{(k)}$, is also random in D_i.

Using the above notation, E's inputs are $\{y, f_i(y), (f_i)^{(2)}(y), ..., (f_i)^{(k-1)}(y)\}$, for a randomly chosen y in D_i, and its output, with probability ϵ, is x s.t. $f_i(x) = y$.

Well, this already looks very close to E simply inverting function f_i on random input y in D_i, which would mean that E is really breaking the assumption that f_i is a OWP. But there is a difference: E gets as inputs not just y, but also all these other values, $\{f_i(y), (f_i)^{(2)}(y), ...\}$. However, all these values can all be *efficiently computed* from y, which means that they do not in fact offer any extra help to E, since E could just as well receive just y and then compute the remaining $(f_i)^{(j)}(y)$ values, for $j = 1, ..., k-1$, himself.

This intuition can be formalized as follows: Algorithm E can be used to construct a PPT A which breaks the one-way property of the OWP family $\{f_i\}$. Constructing such A out of E will show that *if* attacker E exists, i.e. *if* the authentication scheme is *insecure*, then this would imply that the function family $\{f_i\}$ is *not* a OWP family. Hence, by the countepositive, since we assume that $\{f_i\}$ *is* a OWP family, this implies that attack E cannot exist, and hence that our authentication scheme is secure.

Here is how A works: On input y, a random number in D_i, A computes f_i on y for $k-1$ times to get a string of number $s = \{y, f_i(y), (f_i)^{(2)}(y), ..., (f_i)^{(k-1)}(y)\}$, and then A feeds this string of numbers to the algorithm E. Whatever output x algorithm E outputs, A passes this x as its own output. Now, it's easy to see that A breaks the OWP property of the $\{f_i\}$ family with probability ϵ: First of all, as we argued above, the set of numbers s is distributed exactly as the inputs that E expects. Therefore, with probability ϵ, value x returned by E will satisfy equation $f_i(x) = y$. Hence, with probability ϵ, algorithm A, on random input y in D_i outputs x s.t. $f_i(x) = y$. Since E is PPT, and the work of A is the work of E plus $k - 1 < n = 365$ computations of f_i, A is polytime if E is. (Formally, since $n = 365$ is a constant, is is definitely polynomial in the security parameter τ.) □

1.2 [10 points]

What goes wrong with this argument if $\{f_i\}$ is only a family of one-way *functions*, i.e. not necessarily permutations? In particular, assume that each for every i, the range of f_i, $R_i = \{f(x) \mid x \in D_i\}$ is twice smaller than its domain, i.e. $|R_i|/|D_i| \leq 1/2$. A one-way function can indeed have this property.[1] What if f_i is "domain-halving" not just on its domain but also on all consecutive iterations too, so that $|R_i^{(k)}|/|D_i| \leq (1/2)^k$ where $R_i^{(k)} = \{(f_i^{(k)})(x) \mid x \in D_i\}$. Assume that n, the number of days you want the scheme to be good for, is polynomial in τ. Show an efficient attack E that breaks this authentication scheme even if $\{f_i\}$ is a OWF collection.

Solution: If these $\{f_i\}$ are "domain-halving" as above, i.e. if the ranged $R_i^{(k)}$ get smaller and smaller, then the value $x^{(1)} = (f^{(n-1)})(x^{(n)})$ must belong to set $R_i^{(n-1)}$ whose size at most $(1/2)^{n-1} * |D_i| = 2^{-n+1}|D_i|$. Since D_i is sampled by a PPT algorithm *Sample*, the binary lenght of elements in D_i must be bounded by some fixed polynomial, say $p(\cdot)$ in the security parameter τ. In other words, for every $x \in D_i$, $|x| \leq p(\tau)$. Therefore, the *size* of the domain D_i can be at most $|D_i| \leq 2^{p(\tau)}$. Hence, prepare the scheme for $n = p(\tau) + 1$ days (which is polynomial in τ) and you get

$$|R_i^{(n-1)}| = 2^{-n+1} * |D_i| \leq 2^{-(p(\tau)+1)+1} * 2^{p(\tau)} = 1$$

So the authenticating value on day 1 is chosen from a set with at most 1 element! So OK, there is only one possible value for $x^{(1)}$, but can a PPT attacker E *find* this value easily? All he needs to do is to take any $\alpha \in D_i$ and try $\beta = (f_i)^{(n-1)}(\alpha)$. Since $(f_i)^{(n-1)}$ maps D_i into a 1-element set, we must have that $(f_i)^{(n-1)}(\alpha) = (f_i)^{(n-1)}(x^{(n)})$, i.e. that $\beta = x^{(1)}$.

(If anyone noticed that the condition that $|R_i^{(k)}|/|D_i| \leq 2^{-k}$ cannot be true for all k, because for some some k_0, all sets $R_i^{(k)}$, for $k \geq k_0$ must have at least one element, you were right! I forgot to take it into account. But it's still true that one-way *function* does not guarantee security in this application. Fortunately, as we have seen, both the Discrete-Log and the RSA assumptions give rise to one-way *permutations*.) □

2 One-Way Encryption Implies One-Way Function [15 points]

Show that if one-way encryption scheme $\Sigma = (KGen, Enc, Dec)$ exists then there exists a OWF collection. Construct such collection $\{f_i\}_{i \in \mathcal{I}}$. (Hint: unlike in the DL and RSA examples of OWF collections, the index set can be very easy here, e.g. $\mathcal{I} = \{1^\tau\}_{\tau=1,2,\ldots}$.) Remember that each function f_i must be deterministic, so all the randomness used by the encryption scheme will probably need to be an input to f_i.

Side Note: Remember that in the previous homework you showed that indistinguishable encryption implies one-way encryption? Taking the two together means that indistinguishable encryption also implies OWFs. In fact, you can show a similar implication for most cryptographic notions, like secure signature schemes, secure authentication schemes, etc. In other words you can show that if any of them exist (i.e. are secure according to their

[1] You can actually easily construct such function from any assumed one-way functions like RSA or Exp and show that it is both "domain-halving" and still one-way if the original one is.

respective definitions) then one-way functions must exist. This is why we call one-way functions a fundamental notion: If they did not exist then none of this other stuff could be ever achieved!

Solution: Let the index set of the OWF collection be indeed $\mathcal{I} = \{1^\tau\}_{\tau=1,2,...}$. In other words, for every security parameter τ there is just one function $f_i = f_\tau$. Assume for simplicity that the encryption scheme Σ is one-way secure on message space $\{0,1\}^{p_3(\tau)}$ for some polynomial $p_3(\cdot)$. In other words, for every PPT algorithm A, probability that $A(c)$ outputs m on c a *random* encryption $Enc(k,m)$ for *random* k output by $KGen(1^\tau)$ and *random* $m \in \{0,1\}^{p_3(\tau)}$, is negligible (in τ).

This sounds very close to one-way functions, but it involves a lot of randomness. So, To convert *probabilistic* algorithms $KGen$ and Enc to a *deterministic* function f_τ, we need to make the random coins of $KGen$ and Enc explicit. We can do that by treating both $KGen$ and Enc as *deterministic* functions of their inputs and of a *random tape* they use. I.e.,

$$KGen : 1^\tau \times \{0,1\}^{p_1(\tau)} \to \{0,1\}^{p_2(\tau)} , \quad KGen(1^\tau, r_1) = k$$

for some polynomials p_1, p_2, where $p_1(\tau)$ bounds the lenght of the random tape r_1 used by $KGen$ on input 1^τ, and $p_2(\tau)$ bounds the lenght of the output key k of such run of $KGen$. Similarly, we can treat Enc as a deterministic function

$$Enc : \{0,1\}^{p_2(\tau)} \times \{0,1\}^{p_3(\tau)} \times \{0,1\}^{p_4(\tau)} \to \{0,1\}^{p_5(\tau)} , \quad Enc(k,m,r_2) = c$$

for polynomials p_3, p_4, p_5, where $p_3(\tau)$ bounds the lenght of the message m, $p_4(\tau)$ bounds the length of the random tape r_2 used by Enc on such (k,m) pairs, and $p_5(\tau)$ bounds the lenght of the output ciphertext c of such run of Enc.

Now we can construct the one-way function f_τ as

$$f_\tau : \{0,1\}^{p_1(\tau)} \times \{0,1\}^{p_3(\tau)} \times \{0,1\}^{p_4(\tau)} \to \{0,1\}^{p_5(\tau)} , \quad f_\tau(r_1,m,r_2) = Enc(KGen(1^\tau, r_1), m, r_2)$$

I.e., $f_\tau(r_1,m,r_2)$ is the ciphertext output by algorithm Enc executed on (k,m) with random tape r_2, where k is the key output by $KGen$ executed on 1^τ with random tape r_1.

Why is f_τ a one-way function if Σ is a one-way secure encryption on message space $\{0,1\}^{p_3(\tau)}$? Assume it's not, i.e. assume a PPT A which given $c = f_\tau(r_1,m,r_2)$ for random $(r_1|m|r_2) \in \{0,1\}^{p_1(\tau)+p_3(\tau)+p_4(\tau)}$, outputs $(r_1'|m'|r_2')$ s.t. $c = f_\tau(r_1', m', r_2')$ with some non-negligible probability ϵ. In this case we could easily use such A to construct a PPT A' which would violate the one-way security assumption of the encryption scheme Σ: A', on input c, runs A on c, interprets the output of A as a triple (r_1', m', r_2'), and returns just the m' part as its output. Note that the input c to A is just what A expects, and therefore with probability ϵ, its output satisfies equation $c = f_\tau(r_1', m', r_2')$. But that means that c is an encryption of m' for a randomly chosen key k'. Hence, with probability ϵ, A' decrypts c and thus violates the one-way security assumption on the encryption scheme Σ.

(To make this reasoning hold for a *public key* encryption scheme Σ, you can define f_τ as $f_\tau(r_1,m,r_2) = (PK,c)$ where $(PK,k) = KGen(1^\tau, r_1)$ and $c = Enc(PK,m,r_2)$.) □

3 Prime Modular Arithmetics [25 points]

Let p be a prime. Recall groups $\mathbb{Z}_p$ and $\mathbb{Z}_p^*$ from lecture and the handout on modular arithmetics. Recall the fact that $\mathbb{Z}_p^*$ is cyclic and has a generator (in fact it has many of them). Recall set QR_p of squares modulo p.

3.1

Write out the elements of group $\mathbb{Z}_{11}$. How many elements are there? Pick an element g which is a generator of this group. For every element $a \in \mathbb{Z}_{11}$, write down $DL_{(g,11)}(a)$.

Solution: There was a typo here: I wanted you to consider group $\mathbb{Z}_{11}^*$ instead of $\mathbb{Z}_{11}$:

$$\mathbb{Z}_{11} = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$$
$$\mathbb{Z}_{11}^* = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$$

The generators of $\mathbb{Z}_{11}^*$ are $\{2, 6, 7, 8\}$. The easiest one is $g = 2$:

$$2^0 = 1,\ 2^1 = 2,\ 2^2 = 4,\ 2^3 = 8,\ 2^4 = 5,\ 2^5 = 10,\ 2^6 = 9,\ 2^7 = 7,\ 2^8 = 3,\ 2^9 = 6,\ 2^{10} = 1$$

and therefore the disrete logarithms of elements

$$\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\} = \{2^0, 2^1, 2^8, 2^2, 2^4, 2^9, 2^7, 2^3, 2^6, 2^5\}$$

are, correspondingly,

$$\{0, 1, 8, 2, 4, 9, 7, 3, 6, 5\}$$

□

3.2

Let g be a generator of $\mathbb{Z}_p^*$. Prove that if $g^x = g^y \bmod p$ then $x = y \bmod p - 1$.

[*Side note:* In lecture we showed that for any p (even composite), if $x = y \bmod \phi(p)$ then $g^x = g^y \bmod p$ for *any* $g \in \mathbb{Z}_p^*$, which of course includes g which is a generator of $\mathbb{Z}_p^*$.]

Solution: Assume that $x \neq y \bmod p - 1$, i.e. that $x = y + r + i(p-1)$ for some $r \in [1, ..., p-2]$. But that would mean that

$$g^x = g^{y+r+i(p-1)} = g^{y+r}(g^i)^{p-1} = g^{y+r} \bmod p$$

(note that by Fermat's theorem, for any $a \in \mathbb{Z}_p^*$, $a^{p-1} = 1 \bmod p$, thus $(g^i)^{p-1} = 1 \bmod p$)

Now, if $g^x = g^y \bmod p$ then $g^x = g^{y+r} = g^y g^r = g^y \bmod p$, and therefore $g^r = 1 \bmod p$. But that means that g could not a generator of $\mathbb{Z}_p^*$, because if $g^r = 1$ for $r < p - 1$ then set $\{g^i \bmod p \mid i \in \mathbb{Z}_{p-1}\}$ cannot cover the whole of $\mathbb{Z}_p^*$, which has $p - 1$ elements. □

3.3

Let g be a generator of $\mathbb{Z}_p^*$. Prove that $y \in QR_p$ if and only if $x = DL_{(g,p)}(y)$ is even. (Note that since g is a generator then for every $y \in \mathbb{Z}_p^*$ there is $x = DL_{(g,p)}(y)$.) Which of the elements of $\mathbb{Z}_{11}$ are squares?

Solution: If $x = DL(y)$ is even, i.e. $x = 2i$ for some i, then $y = g^x = g^{2i} = (g^i)^2 \bmod p$, and hence $y \in QR_p$.

To show the other side of "if and only if", consider $x = DL(y)$ is odd, i.e. $x = 2i + 1$ for some i. Then $y = g^x = g^{2i+1} = (g^i)^2 g \bmod p$. If y was a square, i.e. if there was some s s.t. $y = s^2 \bmod p$ then we'd have that $g = s^2/(g^i)^2 = (s/g^i)^2 \bmod p$. But that would mean that g is a square itself, i.e. $g = r^2 \bmod p$ for $r = s/g^i \bmod p$. In this case g could not be

a generator, because we'd have that $g^{(p-1)/2} = (r^2)^{(p-1)/2} = r^{p-1} = 1 \bmod p$, and hence, similarly as in the above exercise, set $\{g^i \bmod p \mid i \in \mathbb{Z}_{p-1}\}$ can have at most $(p-1)/2$ elements, and hence cannot cover the whole of $\mathbb{Z}_p^*$.

So which elements of $QR_1 1$ are squares? Those with even discrete logs base 2, i.e. $\{2^0, 2^2, 2^4, 2^6, 2^8\} = \{1, 4, 5, 9, 3\}$. □

3.4

Prove that $y \in QR_p$ if and only if $y^{(p-1)/2} = 1 \bmod p$.

Solution: Clearly, if $y = s^2$ for some s then $y^{(p-1)/2} = s^{p-1} = 1 \bmod p$ by Fermat's theorem.

If y is a non-square, then by the above exercise we know that $y = g^{2i+1}$ for some i, where g is a generator. Therefore,

$$y^{(p-1)/2} = (g^{2i}g)^{(p-1)/2} = (g^{2i})^{(p-1)/2} g^{(p-1)/2} = (g^i)^{p-1} g^{(p-1)/2} = g^{(p-1)/2} \bmod p$$

And, as we argued above, this cannot be equal to 1 if g is a generator.

Since modular exponentiation is polytime, this gives us an efficient way to test if a number is a square in *prime* modular group. (Btw, if n is a composite, testing if $y \in QR_n$ is possible only if you know factorization of n !) □

3.5

Prove that if y, z are quadratic residues modulo p then so is $yz \bmod p$ and $y^{-1} \bmod p$. (In fact this holds for any p, not necessarily prime one.)

Solution: If $y = s^2 \bmod p$ and $z = r^2 \bmod p$ then $yz = (sr)^2 \bmod p$ and $y^{-1} = (s^{-1})^2 \bmod p$, so yz and $1/y$ are squares too. □

3.6

Recall that an *order* $ord_p(a)$ of an element $a \in \mathbb{Z}_p^*$ is defined as the smallest i s.t. $a^i = 1 \bmod p$. The order of the group is the maximum order of any of the group's elements. For each element in $\mathbb{Z}_{11}$ tell its order. Can you see a relation between orders of elements of $\mathbb{Z}_{11}$ and number $\phi(11) = 10$? Can you formulate a general hypothesis about orders of group elements and $\phi(p)$ Can you prove it?

Solution: The orders of the elements in $\mathbb{Z}_{11}^*$ are as follows: $ord(1) = 1$, $ord(10) = 2$, $ord(3, 4, 5, 9) = 5$, and $ord(2, 6, 7, 8) = 10$.

Clearly, these orders are all divisors of $\phi(11) = 10 = 2 * 5$.

This in fact always holds, i.e. for every p, a, $ord_p(a)$ divides $\phi(p)$ (this is true for any p, both prime and composite). We'll show it for *cyclic* groups, which have generators, like the $\mathbb{Z}_p^*$ for prime p (e.g. $p = 11$). Let $a = g^i$ where g is a generator of $\mathbb{Z}_p^*$. Assume that $ord_p(a) = \alpha$. Then $1 = a^\alpha = g^{i\alpha} \bmod p$, and hence

$$i\alpha = j\phi(p)$$

for some integer j. If $gcd(\alpha, j) = f \neq 1$ then $i\alpha' = j'\phi(p)$ for (integers) $\alpha' = \frac{\alpha}{f}$ and $j' = \frac{j}{f}$, but that would mean that $a^{\alpha'} = g^{i\alpha'} = g^{j'\phi(p)} = 1 \bmod p$, and since $\alpha' < \alpha$, α cannot be the order of a.

Therefore α cannot have common factors with j, and hence *alpha* must divide *phi*(p). □

4 Composite Modular Arithmetics [25 points]

Let $n = pq$ for prime p, q. Recall that by the Chinese Reminder Theorem [CRT], for every $r_1 \in \mathbb{Z}_p$ and $r_2 \in \mathbb{Z}_q$ there is a unique $s \in \mathbb{Z}_n$ s.t. $s = r_1 \bmod p$ and $s = r_2 \bmod p$, and vice versa.

4.1

Show how to reconstruct s from (r_1, r_2) if you know elements a, b s.t. $ap + bq = 1$. (Note that $ap = 0 \bmod p$, $ap = 1 \bmod q$, $bq = 0 \bmod q$ and $bq = 1 \bmod p$...)

Solution: $s = (bq)r_1 + (ap)r_2 \bmod n$. Then we have $s \in \mathbb{Z}_n$, $s = r_1 \bmod p$, and $s = r_2 \bmod q$. □

4.2

Consider group $\mathbb{Z}_{35}^*$. List its elements. How many of them are there? What's $\phi(35)$?

Solution:

$G = \mathbb{Z}_{35}^* = \{1, 2, 3, 4, 6, 8, 9, 11, 12, 13, 16, 17, 18, 19, 22, 23, 24, 26, 27, 29, 31, 32, 33, 34\}$

There are $18 = 4 * 6 = \phi(35)$ elements in G, because $35 = 5 * 7$ and 5 and 7 are primes. □

4.3

Consider the CRT theorem for $n = 35$, $p = 5$, $q = 7$. Find a, b s.t. $ap + bq = 1$ (you can find them by hand or by a Euclidean algorithm). Compute $205^{140} \bmod 35$ without a computer. Show your work. (Hint: CRT helps here.)

Solution: We can find a, b by hand, since the formula must be true for some $a \in [1, 6]$ and some $b \in [1, 4]$. Note that $3 * 5 = 15 = 1 \bmod 7$, so $a = 3$ makes $ap = 0 \bmod p$ and $ap = 1 \bmod q$, and that $3 * 7 = 21 = 1 \bmod 5$, and hence $b = 3$ makes $bq = 0 \bmod q$ and $bq = 1 \bmod p$.

To compute $205^{140} \bmod 35$ first note that $205^{140} = 30^{140} \bmod 35$, because $205 = 30 \bmod 35$. Second, note that $30^{140} = 30^{20} \bmod 35$ because $140 = 20 \bmod 24$ and $24 = \phi(35)$. (Recall that $a^b = a^{b \bmod \phi(n)} \bmod n$.)

Third, by the the CRT idea above, to compute $s = 30^{20} \bmod 35$, we can first compute $r_1 = 30^{20} \bmod 5$ and $r_2 = 30^{20} \bmod 7$ and then combine them using the $ap = 15$ and $bq = 21$ values. You'll see how much easier this is than computing $30^{20} \bmod 35$ directly. First, $r_1 = 30^{20} = 0 \bmod 5$ because $30 = 0 \bmod 5$. Then $r_2 = 30^{20} = 2^{20} \bmod 7$ because $30 = 2 \bmod 7$, and then $r_2 = 2^{20} = 2^2 = 4 \bmod 7$, because $20 = 2 \bmod 6$ and $6 = \phi(7)$.

So $s = 205^{140} = (bq)r_1 + (ap)r_2 = 15 * 4 = 60 = 25 \bmod 35$. □