CS 155
Homework #2
Yang Du

Problem 1

a.

P should check (1) if the HTTPS cert is issued by a trusted CA or if the CA is chained by trusted CAs where each non-leave CA's CA bit is set to 1, and the root is a trusted CA, (2)the HTTPS cert is valid (e.g. not expired), and (3) the common Name is in cert matches domain in URL, either explicitly matching, or wildcard matching.

The design will not be vulnerable to ssl strip because it can use https directly so that the attacker will not have a chance to do ssl strip.

b.

The company can embed the updates' public key into the product P and making it immutable and invisible. Thus, for each updates, the product P can use the embedded public key to decrypt the updates getting from the HTTPS web server and install.

c.

Because the company wants to use a distribution network such as BitTorent, it is impossible to guarantee that the each machine participated in the distribution network is trusted and will not change the updates. Thus, it is safer to encrypt the updates once and obtains the signature s so that no one in the middle will be able to modify the updates because of the signature.

The most time consuming crypto operations for using private key is to encrypt the whole update using the secret key. The most time consuming crypto operations for the design in part c is to use the private key to obtain a signature s. The part c design is more efficient because it only needs to encrypt one time and distribute as many times as possible without encrypt it again to get the signature. However, the other design needs to encrypt the product using the exchanged symmetric key every time it makes a https request.

Problem 2

a.

Because the CA bit is not set to 0, so that the attacker can sign its own code using the cert. he bought. Thus, the browser will trust the attacker's code because it is signed by the chained CA.

The attacker can buy a CA from a trusted CA, for its own CA, lets call it CAAttacker. Then, the attacker can use the CAAttacker to sign his attacking code. When the user checks the cert. it will know that CAAttacker is sign by the trusted CA so it is trustable, and then because the attacking code is sign by CAAttacker where CAAttacker is chained with the trusted CA, the attacking code is also trusted. Thus, the attacking code will be run on the user's machine.

b.

The user need to check from the beginning of the CA chain that each CA's parent CA bit is set to 1. If the CA bit is set to 0, then it is unable to sign a document because it is not a CA.

Symantec need to set the CA bit to 0 for every cert. it sells to its customer if Symantec don't want the given cert. be a CA.

Problem 3

a.

After the user has login to a good site and displays private information. And then the user go to the attacker's site, the attacker's site can have a canvas embed the images from the good site. Then, it can use GetImageData() to read the embedded image. Thus, the attacker can get the private information from the image, and do a GET or POST to the attacker's server to receive the information.

b.

I think the GetImageData() should use same origin policy, where it can only get the image data if the image if from the same origin as the website. Otherwise, doesn't allow reading the image data. That is, only the image's origin can read the image data.

c.

The attacker can put a canvas on top of an iframe, where the iframe can contain secret non-image information, and make the canvas transparent. Then the attacker can use getImageData() to get the canvas image so that the underling iframe information can be gotten by the attacker because the canvas is transparent. Thus, the GetImageData() should only return the canvas data even if it is transparent it should return the transparent pixel instead of the underlying pixel.

Problem 4

a.

After the user has logged into the bank website, and then the user goes to the attacker's website. The attacker's website will submit a form to the bank website do

something bad such as transfer money into the attacker's account. Because the user already logged into the bank website, it has a cookie for the bank. Thus, when the transfer money form is submitted to the bank website, the browser will automatically attach the cookie, and the bank will receive the POST request for transfer money.

b.

When the attacker try to send the form, it doesn't know the secret token's value, so that the POST request will be reject because the attacker only have the cookie, not the token.

c.

Yes, this prevents CSRF, because the attacker cannot guess the token value because it changes every time a HTTP request is processed.

d.

No, the attacker can go to the bank directly to get the secret token first. After get the token, use the CSRF to submit the form with the token got in the previous step.

e.

If there is no same-origin policy, the attacker can get the form's token by embedding an hidden iframe inside its attacking page and use the token to submit the form to do something evil.

Problem 5

a.

select * from usertable
where username = 'attacker'
union select * from usertable
union select * from usertable
where username = 'attacker'
and password = '$password'

The red part above can be a attacker's code to a successfully login, because it will union all the user, the num_rows will be greater than 0, and it will successfully login.

b.

Because all the quote will be escaped, the above code, the username will be attacker/' union select * from usertable union select * from usertable where username = /'attacker
So that the injected url will not be executed, the whole thing will be considered as the username, so the attack will be prevented.

c.

attackerbf27
union select * from usertable
union select * from usertable--

When the PHP addslashes function sees 27, it will add 5c before 27, then the character becomes (bf5c) ‘, so that we can attack. The last line is commented out using ‘--‘ so that it doesn’t make the gammar incorrect. The attacking code becomes:

select * from usertable
where username = ‘attacker(bf5c)’
union select * from usertable--
union select * from usertable
where username = ‘attacker’
and password = ‘$password’

Problem 6

a.

If the denial of existence is not authenticated, the attacker can send the user denial of existence before the DNS respond to the user to cause a denial of service attack.

b.

The attacker can first use a non-existing domain name to quest the DNS server to get the denial of existence response. Then, the attacker can still send this response to the users before the DNS respond to the user to cause a denial of service attack.

c.

The resolver can compare to see if the query domain falls in the middle of the NSEC response, which is if the query is lexicographically larger than the response’s smaller name and smaller then the response’s larger name. If it is not lie in the middle, it means it is not the DNS server’s real response. Thus, the resolver will not accept the response and will not cause a denial of service attack.

d.

The attacker can try several different domain name to know all the existing domain name on the server. For example, the attacker can try aa.stanford.edu, and if the DNS tells that it is not exist, and return there is know previous domain, and the next domain is art.stanford.edu, then the attacker will know that the first domain the DNS knows is art.stanford.edu. Then the attacker can try b.stanford.edu, and the response might be DNE with art.stanford.edu, and business.stanford.edu, then the attacker will know there is no existing domain between art and business.

Once the attacker knows what domains the DNS have, it will have a very full image of the website. The attacker can choose any interesting domain to attacker, or the attacker can take all the domain in a website at the same time. In addition, the attacker can have a better understanding of where might be the website's weakness.

e.

The resolver will use the hash function to hash the query domain name and the two returned domain name into a hash code. And then compare the hash codes to see if the query falls in the middle of the two returned domain names' hash codes. Because the hash function is a one-way function, the attacker will not be able to get any information from the two response domain names because it is compare with their hash codes.