CSE 565 - Homework 5

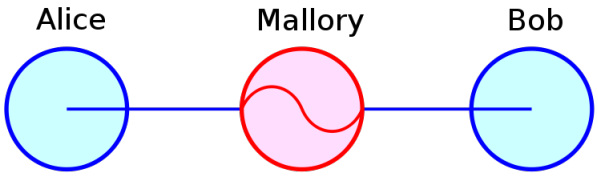
Amlan Gupta (#50288686)

October 2018

1 Question

Consider the following authentication protocol SKID2/SKID3 that uses symmetric cryptography. It assumes that Alice and Bob share a secret key K. SKID2 allows Bob to authenticate to Alice and proceeds as follows:
(a) Alice chooses a random number R_A and sends it to Bob.
(b) Bob chooses a random number R_B and sends to Alice R_B, $MAC_K(R_A, R_B, \mathbf{B})$.
(c) Alice computes $MAC_K(R_A, R_B, \mathbf{B})$ and compares it with what the received from Bob. If the results are identical, then Alice knows that she is communicating with Bob.
SKID3 provides mutual authentication between Alice and Bob. Steps (a)–(c) are the same as in SKID, and then the protocol proceeds with:
(d) Alice sends Bob $MAC_K(R_B, \mathbf{A})$.
(e) Bob computes $MAC_K(R_B, \mathbf{A})$ and compares it with what he received from Alice. If the results are identical, then Bob knows that he is communicating with Alice.
Answer the following questions about the protocol:

(a) If Mallory was to mount a man-in-the-middle attack, describe how she is to proceed. What would happen in the protocol as a result of her attack? Can Mallory succeed?



If Mallory can get access to the shared Key **K**, she can mount a man-in-the-middle attack successfully. Below is the process by which Mallory could succeed:

1. Mallory gets hold of shared Key **K** and the communication channel between Alice and Bob
2. Alice chooses a random number R_A and sends it to Bob.
3. Mallory intercepts the communication, receives R_A and relays it to Bob
4. Bob receives R_A and chooses a random number R_B and sends to Alice R_B, $MAC_K(R_M, R_B, \mathrm{B})$.
5. Mallory again intercepts the communication. Now Mallory is in possession of R_A, R_B, B. If she wants to modify B, along with B message to B'. Along with B', she will send R_B, $MAC_K(R_A, R_{BM}, B')$
6. Alice thinks the message is sent by Bob and Alice sends $MAC_K(R_B, \mathrm{A})$ to Bob
7. Mallory intercepts and updates message as well the hash to $MAC_K(R_B, A')$

8. Bob computes $MAC_K(R_B, A')$ and compares it with the updated message A'. If the results are identical, then Bob knows that he is communicating with Alice.

Following the above method, Mallory can even change the content of Message A and B. We can conclude, if Mallory somehow gets hold of the shared key and hacks the communication channel, she can mount a successfully man-in-the-middle-attack.

(b) Repeat part (a) for a parallel session attack.

In parallel session attack, Mallory can hijack session key from either or both of Alice and Bob. Even if Alice or Bob logs out, the session remains active and Mallory can spoof the messages between them. One of the method is using source-routed IP packets. This will Mallory to participate in a conversation between Alice and Bob by encouraging the IP packets to pass through Mallory's machine.

2 Question

What are advantages of SSL over IPsec and vice versa? Under what circumstances one is preferred over the other?

SSL and IPSec are both equally powerful to solve similar problems.

1.Accessibility and Mobility SSL's primary disadvantage is that it operates at the application layer, limiting access to only those resources that are browser-accessible. IPSec is tied to a specific machine often for a specific user. This can provide stronger security but may limit accessibility and mobility.

2.Encryption : SSL uses 40 or 128-bit RC4 encryption whereas IPsec supports encryption by majorly using the 56-bit DES or 112- or 168-bit Triple DES (3DES) encryption algorithm. IPSec provides better encryption because 3DES is stronger than the other algorithms. However, there are older implementations of IPSec which got broken successfully. So vendors must be aware of pitfalls when implementing IPSec.

3.Authentication: IPSec have to use a certain client software to access the network whereas SSL users can gain access over the network through browsers of any device.

4. Scalability: SSL has high scalability, but since this is tied to applications with the increase in scope the scaling solution must be upgraded as well, IPSec's scalability is much higher as it is platform independent.

5. Complexity and Versatility: IPSec clients may also require manual configuration making them somewhat difficult to use for non-technical workers compared to SSL, which just works. Though both of provides great security features, SSL is easier to manipulate, as holes can be created from individual machines, whereas IPSec is tied to specific machines and implementations.

IPsec should be leveraged in situations where an always-on connection to remote office locations or partners/vendors is required. In these instances, granular access control limitations and missing host-check capabilities should be augmented with a Network Access Control (NAC) system, which can ensure only approved remote hosts are allowed to connect to the enterprise. Enterprises should leverage SSL primarily as a remote access method for the mobile workforce where granular access control capabilities, auditing and logging, and security policy enforcement are crucial.

3 Question

Read following article: M. Georgiev, S. Iyengar, S. Jana, R. Anubhai, D. Boneh, and V. Shmatikov, "The Most Dangerous Code in the World: Validating SSL Certificates in Non-Browser Software," in ACM Conference on Computer and Communications Security (CCS), 2012. Available from www.buffalo.edu/~mblanton/cse565/certificates-article.pdf. Answer the following questions (all description must be your own, not excerpts from the text):

(a) What were the main findings and lessons of the article?

The article demonstrates the pitfalls in SSL (Secure Sockets Layer) implementations in various non-browser software. As SSL was first created by keeping web-browsers in mind, and major browser vendors mostly got the implementation right, however, implementation quality of some of the non-browsers services are abysmal, sometimes even downright broken.

The article demonstrated in many security c-critical application and libraries like Amazon's EC2 Java library, Paypal's merchant SDK, mobile banking app from Chase bank etc implemented broken SSL specifications, which may lead to significant stack-holder loses. Any SSL connection from most of these programs is insecure against a man-in-the-middle attack. The authors found that the underlying cause behind these vulnerabilities is badly designed APIs of SSL implementations and data-transport libraries, which gives developers the option to tune in parameters which may lead to vulnerabilities.

The authors agree that to make the use of SSL in non-browser software safer, better blackbox testing and code analysis tool should be used when developing the code. The APIs should be better designed, the abstraction layer should be properly defined so that inexperienced developers cannot break the security protocol whenever they tinker with it. The libraries which provide implementations for SSL should be properly documented with the critical parameters clearly marked with a warning.

(b) What information did you find to be most surprising? Explain why it was surprising?

The security vulnerability in Zencart payment module surprised me. I have extensive experience in working with Zencart store management system. I had the opportunity to develop a few plugins and themes for applications powered by Zencart e-Commerce. In the two years, I was associated with Zencart I never faced security-related issues with Zencart, however as the article suggested, bugs in SSL modules can be hidden under several layers of abstraction and may take years to find out.

Zencart uses cURL for SSL connections to payment gateways. If cURL is not available, they typically fall back on (insecure) fsockopen. In ZenCart, vulnerable modules include LinkPoint, Authorize.Net, and PayPal Payments Pro, as well as PayPal IPN functionality. This findings at the same time make me surprised and relieved that, I was fortunate enough not to get exploited by this bug.

(c) Choose one example covered in the paper and explain what the weakness from the technical point of view was.

The Android appliction from Chase Bank were insecure against man-in-the-middle-attack.This allows a network attacker to capture credentials of any Chase customer using this app, along with the rest of their session.

```
public final void checkServerTrusted(X509Certificate[] paramArrayOfX509Certificate, String paramString)
{
    if ((paramArrayOfX509Certificate != null) && (paramArrayOfX509Certificate.length == 1))
        paramArrayOfX509Certificate[0].checkValidity();
        while (true)
        {
            return;
            this.a.checkServerTrusted(paramArrayOfX509Certificate, paramString);
        }
}
```

It is evident from the reverse engineered code that, it overrides the default X509TrustManager and the replacement code returns without checking the server's certificate. The invocation of checkServerTrusted is not reachable as the function returns before that. The authors suspect, this was a temporary plug during development that somehow found its way into the production.

References

[1] William Stallings; Lawrie Brown. *Computer Security: Principles and Practice (3rd Edition)*. Abe Books, 2014.

[2] M. Georgiev, S. Iyengar, S. Jana, R. Anubhai, D. Boneh, and V. Shmatikov. The most dangerous code in the world: Validating ssl certificates in non-browser software, 2012.

[3] Inc OpenReach. Ipsec vs. ssl: Why choose?, 2002.