Software Assignments for a Course in Secure E-Commerce

Chris Steketee and Phillip Lock

Advanced Computing Research Centre, School of Comp and Info Science
University of South Australia, Mawson Lakes, SA Australia
{Chris.Steketee, Phillip.Lock}@unisa.edu.au

Abstract. This paper describes a course in computer security for advanced undergraduate students in computer science and software engineering. The aim of the course is to give the student a thorough grounding in the principles and practice of cryptography and secure network protocols, and in the application of these to the development of e-commerce applications. An important part of the learning process is an assignment in which the student develops software for a specified e-commerce application. The paper describes a number of these assignments that have been run over the past several years, and reflects on the lessons learned.

1 Introduction

The teaching of computer security in undergraduate programmes has struggled to fit the burgeoning growth in the field into a limited number of course hours. At the University of South Australia, this has been addressed by building an undergraduate Information Security stream into our Computer Science and Software Engineering programmes, including advanced electives. Currently, two advanced electives are offered: Forensic Computing and Secure E-Commerce, and students can take both, one or neither, depending on their interests. Our information security stream is described in [1] and has continued to develop in line with re-development of the undergraduate programmes. In accordance with the University's orientation towards producing practically-oriented graduates, the goal is to produce in students a well-rounded understanding of IT security and an ability to apply this understanding in industry. Foundation courses take a broad view of the field, from both management and technological perspectives, whereas the advanced electives focus in relatively specialised areas.

Secure E-Commerce is targetted at imparting the understanding and skills needed for the design and implementation of secure software systems for electronic commerce on public networks such as the Internet and cellular telephone networks. Our dual intent is to provide a theoretical basis for our student's understanding of security technologies and at the same time equip them to apply this to secure software development in industry. We aim to impart a practical understanding of the principles of cryptography, secure network protocols and public key infrastructure. We include detailed coverage of the principal cryptographic algorithms (both symmetric and asym-

Please use the following format when citing this chapter:

Steketee, C., Lock, P., 2007, in IFIP International Federation for Information Processing, Volume 237, Fifth World Conference on Information Security Education, eds. Futcher, L., Dodge, R., (Boston: Springer), pp. 113–120.

metric) and cryptographic protocols, and methods of authentication including the use of smart cards and biometrics. The last part of the course concentrates on the application of these building blocks to the production of secure systems for electronic commerce application areas such as digital rights management and electronic payment. The course is taught by means of lectures (including guest lectures from industry), a software assignment and a literature research assignment, and workshops which are used to support both the lecture and the assignment work.

1.1 The Software Assignment

Bearing in mind the aims of the course, it is important for students to understand how theory translates into practice for system designers and programmers, gaining hands-on experience in the use of standard security libraries as building blocks for a secure system. This is the purpose of the software assignment. The assignment gives students experience in applying the concepts taught in lectures to an e-commerce application by developing a prototype software system. In the course of this assignment, students deal with issues such as secure storage of keys, the choice of encryption algorithms and modes, and the use of a complex API for cryptography – in this case, the Java Cryptographic Extension, JCE [2].

In addition to developing the software, students are required to reflect on its security properties and write a brief security analysis. This helps to teach them that security in software is difficult to achieve, and in particular that achieving a secure system requires more than building a working application.

For this assignment, students are given a specification which typically includes a high-level security and software design, but are left a considerable amount of freedom in the way they implement this. They are given simple programs to get them started on JCE, but are expected to spend time familiarising themselves with the JCE by reading the specifications and browsing the API. These are important skills for software engineers to develop.

1.2 Educational Objectives

Bloom's taxonomy of educational objectives [3] divides cognitive skills into six stages: knowledge, comprehension, application, analysis, synthesis and evaluation.

This assignment concentrates on the application, synthesis and evaluation stages. In the *application* stage, students use their experience of software design and implementation and apply it to the requirements of the specific problem. The *synthesis* stage consists of putting together their knowledge of computer security with the security requirements of the assignment and the building blocks supplied by the Java security libraries on which their solution will depend. In *evaluation*, students show an understanding of the security strengths and weaknesses of the system, by analysing possible security attacks and proposing defences against these attacks.

1.3 Related Work

We are not aware of a course in Secure E-Commerce with the same focus as ours. A number of courses with a similar title focus on protection issues, particularly in relation to web sites, and limit the practical work to laboratory sessions – eg University of Gloucester [4] and Rochester Institute of Technology [5]. The University of Aberdeen's course "Security and Privacy" [6], which is aimed at applications in e-commerce, e-health and e-science, includes some coverage of cryptography and security protocols but this is necessarily limited because of the broader scope of the course. Royal Holloway offers a complete Masters degree in Information Security [7], with an opportunity to go into topics in much more depth, but there appears to be little emphasis on laboratory or programming work.

In terms of work similar to the assignments described below, a paper by Rawles and Baker [8] presents an approach to teaching students the concepts of Public Key Infrastructure (PKI) and its application to secure e-mail. The emphasis in that paper is on the hands-on experience of setting up a PKI and secure e-mail system. A PKI programming assignment set by Boneh at Stanford [9] bears similarities to two of our assignments: it requires students to develop a chat room with authentication via SSL, including the requirement to set up a PKI. Another secure chat room assignment by Mitchener and Vahdat [10] forms the basis of our own secure chat room assignment.

2 The Assignments

2.1 Mobile Online Music Store

This assignment illustrates security and cryptography concepts by applying them to a mobile-enabled music store with a simple form of Digital Rights Management (DRM). Students are required to produce a prototype of a music store server and client, which enables the user of the client to download, store and play music tracks, while preventing unauthorised use of the downloaded music.

The client is a mobile phone: this has two consequences. Firstly, it adds interest and currency, as music download has only recently become available for mobile phones. Secondly, it illustrates to students the fact that the mobile phone is a more controlled (and often more secure) environment than a standard desktop computer and therefore amenable to different solutions. For pragmatic reasons, the assignment is actually carried out on a normal computer rather than a phone or a phone simulator, but students need to remember in their security analysis that the system being simulated is phone-based.

The design and implementation complexity of the assignment is reduced by specifying that the music store is operated by a mobile phone service provider, that payment for music is added to the user's phone bill, and that access to the music store is restricted to music-store enabled phones provided by the service provider. This means that payment issues can be excluded from the implementation, and that key distribution issues are simplified.

Security and DRM are based on public key cryptography, with each phone having a key pair. Key pairs are assumed to be generated and distributed securely by the service provider before the phone is issued to the customer: the private key is stored on the phone (only), and the music store maintains a database containing the public key of each phone.

When a phone downloads a music track, the music store server encrypts the music to be downloaded with a random session key, encrypts the session key with the public key of the phone, and sends both to the phone, which stores the music in its encrypted form. The phone's keypair therefore provides the basis for both confidentiality of the music download, and copy protection of the downloaded music. An eavesdropper on the network sees the music in encrypted form only, and any copy made of the encrypted file is not playable on another device. The use of this mechanism illustrates to students the use of encryption on network connections, and one approach to providing a simple form of DRM.

Implementation For this assignment, students were required to implement the RSA algorithm from first principles. This showed students that typical public key encryption algorithms are not difficult to implement, and it also provided a good opportunity to discuss the need for a randomised padding scheme such as PKCS #1 and the attacks it is designed to prevent. On the other hand, for symmetric encryption, they were expected to use the JCE: in part to keep the amount of work required to reasonable bounds, and in part to give an insight into the use of cryptographic libraries.

To keep the amount of work to a manageable size, students were given considerable help in the workshops, including Java source code starting points, and discussion time on RSA implementation.

Security Analysis As well as producing an implementation, students were asked to produce a document analysing the security of this system. The specification was couched in broad terms (what are the security threats and attacks, what system design measures are taken against these, what are the biggest threat and easiest attack remaining?). This meant that students needed to think about the security features of the system: those which are inherent in the specification (such as copy protection), those which they may have added in their implementation (such as protection of the private key), and those which are an inherent weakness (such as the need to trust the client software not to reveal secret information). The security analysis serves to remind students that getting a working implementation is only part of the solution.

Lessons Learned This assignment was quite successful and achieved its aims for most students. The great majority were able to produce at least a basic working implementation, and the better students did everything asked for. However many students produced solutions in which secret information, such as passwords and secret keys, were stored on the client unencrypted. The security analysis was less carefully done than the programming despite accounting for 30% of the available marks. This suggests a need to pay more attention in future presentations of the course, on the one

hand to secure programming practices, and on the other to the skills needed to analyse the security of, and security gaps in, a system.

Extensions and Variations The assignment lends itself to many extensions and variations. We note here just a few of them. A complementary assignment could be set to deal with the process of secure keypair generation and distribution, which is taken as a given for this assignment. Another extension or complementary assignment could focus on a payment mechanism. An alternative version of the assignment could be based on the use of a PKI, with each party having a public key certificate: this could be the basis for an implementation not limited to a single service provider.

2.2 Public-Key Infrastructure and Secure Email

The purpose of this assignment is to give students hands-on experience implementing a PKI, and writing programs which make use of a PKI in order to achieve secure communication. Students first set up a simple Certificate Authority and produce signed certificates for the communicating parties. They then write application code to achieve secure communication, with the signed certificates in the PKI providing the basis for the security.

The specific communication task in this assignment is a simulation of secure email communication between a bank and its customers. This is of particular relevance in the current environment in which phishing attacks based on email are common, and where Internet banking provides a target which is particularly attractive to the phishers. The task is an illustration to students that it is possible to have email over an insecure network which is secure in terms of both confidentiality and authentication; the choice of a commercial setting is deliberate in order to relate the assignment to the aims of this course.

The PKI in this assignment is one that is targeted for email communication between the bank and its customers. The bank runs a root Certificate Authority (CA) which signs the certificates of all communicating parties (or clients) – these are defined to be bank departments and customers. All parties are assumed to use the same client software (programmed for this assignment by the student) to send and receive secure email, configured to trust the bank's root CA certificate. This leads to a simple PKI, since there is only one CA, and all certificates are signed directly by that CA. A further simplification is that certificate revocation is not included.

In order to send an email, a client needs to have a certificate of its own, and also the certificate of the intended recipient. The sender's private key is used to sign the message and the recipient's public key is the basis for encrypting the message via a random session key. The recipient uses its own private key to decrypt the message, and uses the public key of the sender to verify the signature of the message. The recipient also validates the sender's certificate against the trusted root CA certificate and checks its validity dates.

Implementation Java keystores are a convenient way to store the keypairs and certificates for this assignment. For the Certificate Authority functionality we used

the OpenSSL command-line tool [11], which includes a simple CA function. Support was provided in workshops in which the principles of PKI were discussed before they were covered in lectures, in the provision of sample programs to illustrate the use of the JCE for encryption and certificate validation, and in the use of OpenSSL to set up a CA. In order to allow the students to concentrate on the central issues of the assignment, we allowed them to simulate the sending and receiving of secure email by writing and reading disk files, rather than using (say) JavaMail.

Security Analysis The security analysis required for this assignment was specified more explicitly than that for the music store, concentrating on matters like how bank customers are identified and authenticated, the protection against phishing that the use of such a system may provide, and also how secret data such as private keys and passwords are protected.

Lessons Learned Overall, this assignment was less successful than we had hoped. Just two students (10% of the class) completed the work fully and successfully. Despite the considerable amount of help supplied, the others struggled to varying degrees. It is clear that many had difficulties getting to grips with the range of new ideas to learn for this assignment. The main problem may simply have been the complexity of the Java API for cryptography and security, and the students' relative lack of skills in constructing solutions based on the API and simple examples (in terms of Bloom's taxonomy, in the synthesis stage of learning).

Extensions and Variations One area for extensions or complementary assignments is to enhance the PKI functionality, such as adding revocation checking via Certificate Revocation Lists or the OCSP protocol, and generalising to a CA hierarchy with multiple levels. A related extension would be the use of full-function CA software, which would allow the inclusion of certificate revocation and also the periodic re-issue of client certificates. Building a system that sends and receives real emails would be an improvement over simulated email, allowing students to get a clearer picture than they do in an application which is just based on the reading and writing of local files. Variations that apply PKI to some application other than email, for example a payment system, are also possible.

2.3 Secure Chat Room

This assignment requires students to build the infrastructure for secure network communication, and apply it to an Internet chat room application. The basic idea was taken from an assignment published by Mitchener and Vahdat [10]. Students are given an implementation of a chat room application that communicates in plaintext, and are required to convert this to communicate using encrypted messages. Unlike the two assignments described above, this one concentrates on the implementation of a secure protocol, rather than on a specific E-Commerce application.

The protocol which the students are required to implement is essentially a simplified version of Kerberos [12], and is therefore based on symmetric cryptography and the use of a trusted authentication server. Each participant shares a long-term secret

key with the authentication server, derived by hashing a password typed in by a user. This is used to authenticate the user and also to encrypt communication with the authentication server. When participant Alice (eg a chat user) wants to communicate with another participant Bob (eg the chat server), Alice requests a ticket from the authentication server: this ticket contains two copies of a newly-generated session key, one encrypted with Alice's long-term key and the other with Bob's long-term key. When Alice opens communication with Bob, she sends Bob's copy of the encrypted session key, and subsequent communication between Alice and Bob is encrypted with this key. The authentication protocol also includes timestamps as a defence against replay attacks.

Implementation Once again students are expected to implement this assignment in Java, using the JCE to perform the cryptographic operations. A plaintext chat room application is supplied as a starting point.

Security Analysis Topics that students were asked to cover in their analysis included the design measures taken to protect long-term keys on the authentication server, secure registration and maintenance of user accounts, and possible attacks, for example the distribution by an attacker of a modified client.

Lessons Learned The assignment was quite successful, with the great majority of students achieving at least a basic working implementation and demonstrating a reasonable understanding in their security analysis. One shortcoming in most solutions was to build the security implementation directly into the chat application rather than encapsulating it in (say) a secure subclass of the *Socket* class.

Variations Clearly the assignment can be adapted to apply the same concepts to other applications needing authentication and encrypted communication. Given our experience noted above, another variation might be to require students explicitly to define and implement a Java API for secure communication based on this protocol, before using it for a specific application.

3 Conclusions

We have described several software design and implementation assignments in which cryptographic techniques are applied to the development of secure applications in an e-commerce environment. These assignments focus on the use of standard implementations of cryptographic algorithms for security purposes rather than on the development of new implementations. The aim is to develop in students a detailed appreciation of the issues involved in developing secure application software, ranging from the nuts and bolts of using a complex cryptographic API, through design issues such as the secure storage of private keys, to an overall security analysis of the system, including remaining security weaknesses. We feel that these assignments have been successful in achieving this aim, in addition to providing students with experience that is valuable for prospective employers.

These assignments have also provided us with a good feedback mechanism on the strengths and weaknesses of our course. One shortcoming that has become apparent is the difficulty many students have in providing a carefully thought-out evaluation of a system's security: while most students are able to produce a standard list of security threats and defences, the ability to analyse threats to a specific software system that they have produced themselves is often lacking. It is clear attention should be paid to this aspect in future presentations. Another lack that has been revealed in the course is the absence of material on secure programming practices, for example to minimise the exposure to a hacker of sensitive information such as passwords and private keys by ensuring that this material is erased from memory after use.

References

1. J. Slay and P. Lock, "Developing an Undergraduate IT Security Stream: Industry Certification and the Development of Graduate Qualities," presented at Fourth World Conference on Information Security Education, WISE4, Moscow, Russia, 2005.
2. Sun Microsystems, "Java Cryptography Extension (JCE) Reference Guide," 2004, http://java.sun.com/j2se/1.5.0/docs/guide/security/jce/JCERefGuide.html, accessed 08/08/2006.
3. B. S. Bloom, Taxonomy of educational objectives. Boston, MA: Allyn and Bacon, 1984.
4. S. A. Shaikh, "Information Security Education in the UK: a proposed course in Secure E-Commerce Systems," presented at 1st Annual Conference on Information Security Curriculum Development, Kennesaw, GA, USA, 2004.
5. Rochester Institute of Technology, "Secure E-Commerce," 2006, http://register.rit.edu/courseSchedule/4002877, accessed 31/10/2006.
6. University of Aberdeen, "Security and Privacy," 2006, http://www.csd.abdn.ac.uk/~jmasthof/teaching/CS5401/, accessed 31/10/2006.
7. R. Holloway, "Master in Information Security," 2006, http://www.isg.rhul.ac.uk/msc, accessed 31/10/2006.
8. P. T. Rawles and K. A. Baker, "Developing a public key infrastructure for use in a teaching laboratory," presented at 4th Conference on Information Technology Curriculum, Lafayette, Indiana, USA, 2003.
9. D. Boneh, "Cryptography and Computer Security: Programming Project #2," 2004, http://crypto.stanford.edu/~dabo/courses/cs255_winter04/, accessed 14/08/2006.
10. W. G. Mitchener and A. Vahdat, "A Chat Room Assignment for Teaching Network Security," presented at 32nd Technical Symposium on Computer Science Education (SIGCSE), 2001.
11. OpenSSL, http://www.openssl.org/, accessed 14/08/2006.
12. B. C. Neumann and T. Ts'o, "Kerberos: An Authentication Service for Computer Networks," IEEE Communications, vol. 32, pp. 33-38, 1994.