EP2500/FEP3200 Networked Systems Security

Final Exam Fall 2023 (HT23)

Total: 300 points (plus 80 points as extra credit).
Pass (MSc students): 165 points. **Pass** (PhD students): 225 points.

Guidelines

Closed books, notes, other references, dictionaries, etc.
No computers, smart-phones, calculators, cameras, etc.
Please write your name on this exam handout and return it intact with your answers.
Please read carefully Section 1, "Answering guidelines & Notation" before you start answering.

Best of Luck!

Panos Papadimitratos
Networked Systems Security Group
www.eecs.kth.se/nss

January 9, 2024

Contents

Notation

Please consider and use the notation in Table 1. If you think you need any additional notation, please define and justify this explicitly and state any assumptions you make. Consider an adversary that can never compromise the secret or private keys of the involved entities, but she can intercept, forge, modify or erase any transmitted message. For each answer you give, e.g., for each protocol you design, please give a brief (no more than 3-4 lines, unless absolutely necessary) explanation. When you present a protocol, please do not forget to present the message exchange and the operations (computations) on both (all) sides (involved entities).

Table 1: Notation of cryptographic primitives.

Symbol	Description
$A : \alpha$	Entity A takes action α
$x \leftarrow y$	The value y is assigned to variable x
$\mathtt{RNG}$	A random number is generated
$x == y$	Check if x is equal to y
$\mathtt{Pub}_A$	Public Key of A
$\mathtt{Priv}_A$	Private Key of A
$\mathtt{Cert}_A$	Certificate of A containing $\mathtt{Pub}_A$
$\mathtt{Cert}_A^X$	Certificate of A registered with domain X (containing $\mathtt{Pub}_A$)
K_{AB}	Symmetric secret shared by A and B
K_A / K_{AB}	Symmetric secret generated by A / shared by A and B
$c \leftarrow E_K(m)$	Encrypt m with the symmetric key K and generate the ciphertext c
$m \leftarrow D_K(c)$	Decrypt c with the symmetric key K and recover the message m
$c \leftarrow \mathtt{Enc}_{\mathtt{Pub}_A}(m)$	Encrypt m with the public key $\mathtt{Pub}_A$ and generate the ciphertext c
$m \leftarrow \mathtt{Dec}_{\mathtt{Priv}_A}(c)$	Decrypt c with the private key $\mathtt{Priv}_A$ and recover the message m
$\sigma \leftarrow \mathtt{Sign}_{\mathtt{Priv}_A}(m)$	Signature σ on message m computed with the private key $\mathtt{Priv}_A$
$\mathtt{Success/Failure} \leftarrow \mathtt{Ver}_{\mathtt{Pub}_A}(\sigma)$	Verification of signature σ on message m computed with the public key $\mathtt{Pub}_A$
$h \leftarrow H(m)$	Hash of m, i.e., the output of a one-way/hash function, H, with input m
$A \rightarrow B : (x, y, \ldots)$	Message sent from A to B containing x, y, $\ldots$
t_{clock}^A	The clock value of entity A
N_A	Nonce chosen by A

1 "Right or Wrong"

Exercise 1 Quick answers (25 points)

Please read each of the following statements and answer briefly "Right" or "Wrong" and please add a mandatory "because:" two-line justification.

1. (2.5pt) Asymmetric key cryptographic primitives applied on messages, e.g., digital signatures, utilze the same cryptographic key at the sender and receiver.
2. (2.5pt) Clogging (resource-depletion) Denial of Service (DoS) attacks cannot be mitigated by stateless firewalls.
3. (2.5pt) IPsec Authentication Header (AH) in tunnel mode hides the source and destination host addresses.
4. (2.5pt) Adding pepper makes password cracking impossible.
5. (2.5pt) Kerberos cannot support access to services in an organization different than the one that provided a user with credentials.
6. (2.5pt) Border Gateway Protocol (BGP) routers in an Autonomous System (AS) that uses Resource Public Key Infrastructure (RPKI) digitally sign the prefix announcement and the AS-PATH.
7. (2.5pt) Secure and fault tolerant communication protocols add significant communication and computation overhead.
8. (2.5pt) The Domain Name System Security Extensions (DNSSEC) authenticates records provided by the authoritative name servers.
9. (2.5pt) Random key pre-distribution implies that two neighboring sensor nodes need to perform a so-called key discovery.
10. (2.5pt) Two distinct pseudonymous certificates, attached to digitally signed messages by the same entity, cannot be linked by an eavesdropper that is given only the two pseudonymous certificates and the corresponding signatures.

Answer of exercise 1

1. **Wrong.** Asymmetric key cryptographic primitives use a public key for encryption and a private key for decryption, which are not the same. This is essential for functions like digital signatures, where a message is signed with a private key and verified with the corresponding public key.
2. **Right.** Stateless firewalls inspect packets in isolation and may not maintain state information that can be used to mitigate some DoS attacks. Other strategies such as rate limiting or IP blacklisting can still provide some level of mitigation against DoS attacks.
3. **Wrong.** IPsec AH provides data origin authentication, data integrity, and replay protection. However, it does not encrypt the IP packet, which means that while the packet contents are authenti-

cated, they are not hidden; the source and destination IP addresses in the outer IP header remain visible even in tunnel mode.

4. **Wrong.** Adding 'pepper' can make password cracking more difficult because it adds an additional layer that an attacker must overcome, but it does not make it impossible.
5. **Wrong.** Kerberos can support access to services across different organizations through the use of authentication, where two Kerberos trust each other's authentication process.
6. **Wrong.** RPKI does not digitally sign the route announcement. Instead, it only provides a mean for origin authorization. It does not sign the path, therefore given RPKI, the path can be tampered with.
7. **Wrong.** Secure and fault-tolerant communication protocols do not necessarily add significant communication and computation overhead. Some protocols are designed to be lightweight and efficient, adding minimal overhead while still providing security and fault tolerance.
8. **Wrong.** DNSSEC is designed to provide authentication for all DNS records, not just those provided by authoritative name servers. It ensures the authenticity and integrity of all DNS responses, regardless of their source within the DNS infrastructure.
9. **Right.** Random key pre-distribution indeed requires that two neighboring sensor nodes perform key discovery to find a common key among the subsets they hold. This is necessary because each node does not initially know which keys its neighbors possess.
10. **Right.** An adversary cannot link the two pseudonymous, since the signatures are generated using two different keys.

2 Asymmetric key management

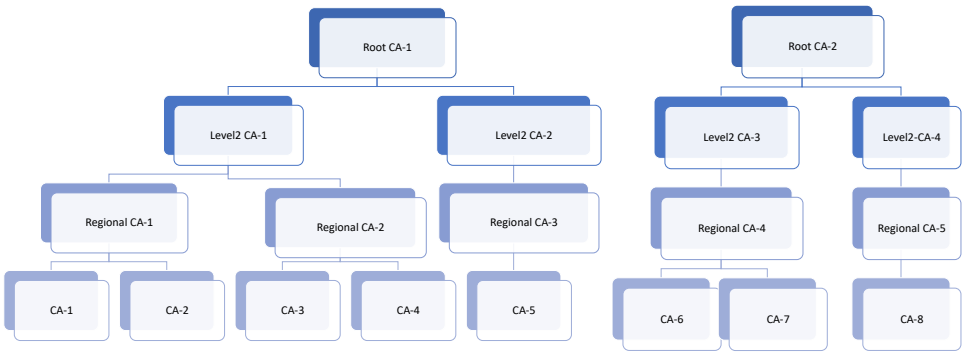

Figure 1: Public Key Infrastructure (PKI) hierarchy.

For Ex. 2, Ex. 3, and Ex. 4 below, please consider the PKI hierarchy in Fig. 1: each Certificate Authority (CA) CA-1, CA-2, ..., CA-8 interacts with users (A, B, C, ... Z) and it is provided with a certificate from the CA at the hierarchical level above, e.g., CA-5 from *Regional CA-3*; which in turn gets its certificate from *Level2 CA-2*, etc. The two root CAs have self-generated certificates. Each user registers with at most one bottom-level CA and obtains a certificate.

Exercise 2 Certificate issuance and use (70 points)

Consider A registered with CA-3 and U registered with CA-8. Each CA has its own registration authority that establishes $\texttt{registration}^{\texttt{A}}_{\texttt{info}}$ for A and CA-3 and $\texttt{registration}^{\texttt{U}}_{\texttt{info}}$ for U and CA-8 respectively.

1. (10 pt) Consider A with $\texttt{registration}^{\texttt{A}}_{\texttt{info}}$ requesting CA-3 a *certificate*. To ensure accountability, who generates A's public/private key pair and why?
2. (20 pt) What should the *certificate signing request* sent by A to CA-3 include? What are key elements/fields in the newly acquired certificate of A?
3. (10 pt) Consider B, registered with CA-5, running a Transport Layer Security (TLS) handshake with A. Please describe what A needs to do to validate B's certificate. (Please do not write the TLS protocol).
4. (10 pt) Assume now C, also registered with CA-5, runs TLS with A. How can A, having interacted with B earlier, reduce computation overhead when validating C's certificate?
5. (20 pt) Consider next U communicating with A. Can A validate U's certificate? Please explain how and add needed steps (if any).

Answer of exercise 2

1. (10 pt) A generates its own key pair, create an *certificate signing request* encapsulating newly created public key, signing it with the corresponding private key, and sends it to CA-3 with $\texttt{registration}^{\texttt{A}}_{\texttt{info}}$. If all is well, CA-3 signs the request with its private key, creating a certificate. The resulting certificate is sent back to A.
2. (20 pt) CSR should include the public key of A, signature generated by A's private key, expiration date, serial number, name of the user (subject name). The certificate should include everything that CSR has, the name of the CA (issuer field) that signed the certificate, and signature generated by CA's private key.
3. (10 pt) A will try to verify B's certificate but it will see that it is signed by CA-5. A will check if it has the public key of CA-5. If it does not, it will go to issuer, CA-5, and get the certificate of CA-5 which includes its public key. However, A needs to verify the newly obtained certificate of CA-5. It will check if it has the public key of Regional CA-3. If it does not, it will again go to the issue and get the certificate. This process will continue until it reaches the root CA-1 which A trusts. If all the certificates are valid, A will be able to verify B's certificate by tracing the chain.
4. (10 pt) In the previous step, A has already verified the certificate of CA-5. Therefore, it does not need to verify it again. It can directly verify the certificate of C by the public key of the CA-5. This will happen if the certificate of CA-5 is not expired and A cached it previously.
5. (20 pt) With the current construction, A cannot validate U's certificate since the parties have different root CAs. To make this work, root CAs need to cross certify each other. Simplest way to do this, A's root CA signs the certificate of U's root CA and vice versa, given that they trust each other. A can ask its own root CA about the root CA-2, whether it is trusted or not. If it is, root CA-1 can then return the certificate of root CA-2 to A, signed with its private key.

Exercise 3 Certificate revocation (70 points + 20 extra credit)

1. (20 pt) Consider CA-3 informed that A's private key is compromised. How can W registered with CA-3 be informed that A can no longer be trusted (prior to establishing a secure communication channel with A)? Please explain briefly two solutions.
2. (20 pt) How can X registered with CA-4 also be informed that A can no longer be trusted? How can X validate a Certificate Revocation List (CRL) by CA-3?
3. (10 pt) How can U do the same?
4. (20 pt) What if there are many incidents of users registered with CA-3 that are also compromised? How is each solution handling this?
5. (**extra credit** 20 pt) Consider the two solutions for revoking any user certificate, e.g., A's, and compare their efficiency in terms validating any certificate. What about their effectiveness in terms of timely provision of revocation status?

Answer of exercise 3

1. Method 1: With a CRL list, which is publicly available, W can check if any certificate issued by CA-3 is revoked.
   Method 2: OSCP stapling! The sender attaches a status report for the certificate.
2. The CRL is publicly distributed and signed by CA-3. X in CA-4 can cross check the signature on the CRL leveraging REG-CA-2, which signs both CA-3 and CA-4.
3. U cannot do the same, as there is no cross validation between ROOT_CA_1 and ROOT_CA_2
4. For method 1 the CRL will contain the revoked certificates at the time of its issuance.
   For method 2, when the communication happems, the sender will attach the status report of his certificate.
5. Method 1: The CRL is efficient as it pushes the burden of checking which keys are valid and which are not on the client but it is not necessarily timely. Usually CRLs are distributed at a certain frequency and the method relies on the nodes actually checking the CRL.
   Method 2: OSCP stapling is efficient and low overhead (and shifts the burden to the sender), but it is not yet supported everywhere.

3 Secure and fault tolerant communication

Exercise 4 Secure and fault tolerant queries (70 points + 40 extra credit)

Consider B and U registered with their respective CA as above. Assume they run a protocol, with B sending periodically queries to U's network. It suffices that U or one of its peers, $U_1, U_2, \ldots, U_K$ respond to B's query.

1. (20 pt) Please describe how the queries by B can be authenticated and their integrity and freshness be protected? How can the response by U be safeguarded in the same manner as the query? Reuse concisely steps needed from the previous exercises.

2. (20 pt) Although one response suffices, by any of the U_i ($i = 0, \ldots, K$, where $U = U_0$) receiving the query, a naive design would return $K + 1$ responses. Assuming all U_i are in the same network, propose a protocol that (i) returns a few if not ideally a single response, and (ii) lets the rest of the U_i know the query was serviced.

3. (20 pt) Consider now an adversary, M, in the network of the U_i but not controlling their communication. How can M attempt stopping U_i responses? Can M mislead several (or even all) U_i to respond unnecessarily? Please explain.

4. (10 pt) How would you change the protocol to have all aforementioned security and functionality and confidentiality for U_i responses?

5. (**extra credit** 20 pt) Consider now that one of the U_i is adversarial, let's denote this as U_m. Please explain if, given your protocol, U_m can provide B with an incomplete answer. If so, please modify your solution (without sacrificing response confidentiality) to either prevent or at least allow the rest of the U_i detect the wrong-doer. You can assume that all U_i have the same data based on which responses are formed. What is the maximum number of wrong-doers your solution can handle?

6. (**extra credit** 20 pt) Discuss how to change your protocol to efficiently protect the confidentiality of the queries by B to the U_i cluster. If you cannot achieve that, please explain the challenge and the overhead. Can you solve this by changing the network structure on the side of the U_i? Please explain how and briefly point out trade-offs.

Answer of exercise 4

1. Queries and responses can be signed with the private keys and verified based on the certificates. The certificates need to be verified based on the answers from the above exercises.

   $B \rightarrow U : Query, t^{B}_{clock}, Sign_{Priv_B}(H(t^{A}_{clock}, Query)), Cert_B$
   $U_i \rightarrow B : Response, t^{U_i}_{clock}, Sign_{Priv_{U_i}}(H(t^{U_i}_{clock}, Query, Response)), Cert_{U_i}$

2. When U_i receives a query, it will wait for a short duration before generating the response. The duration is a random value from an acceptable time delay range. The response will be both sent to B and broadcasted within the network of U. If any response to the same query was received before the waiting expire, then no response is generated.

3. M can stop U_i response by immediately sending out a bogus response to a query without waiting, so that all U consider that the query was served. M can mislead multiple or even all U respond if it erase any broadcasted U_i response from U network.

4. The response can be either encrypted with the public key directly or with a symmetric key that is encrypted with the public key.

$m \leftarrow U_i : E_{K_{U_i B}}\{Response\}, t^{U_i}_{clock}, E_{Pub_B}\{K_{U_i B}\}$
$U_i \rightarrow B : m, Sign_{Priv_{U_i}}(H(m)), Cert_{U_i}$

5. Yes, the adversarial U_i can provide dishonest answers by immediately responding without random waiting.

   All U_i should share a symmetric key, K_U, with the U network.

   $U_i \rightarrow B : E_{K_{U_i B}}\{Response\}, t^{U_i}_{clock}, E_{Pub_B}\{K_{U_i B}\}, E_{K_U}\{K_{U_i B}\}, Sign_{Priv_{U_i}}(H(Response, t^{U_i}_{clock}, K_{U_i B})), Cert_{U_i}$

   Any U_i can see the response, and an honest U_i can respond with an honest response if a dishonest response is received.

   B can report conflicting responses if there is any authority is able to figure out which one is dishonest. Otherwise, B has to accept a response with the majority of votes. Any honest U_i should respond until the number of honest responses is greater than the (identical) dishonest response. Depending on what is the percentage threshold of majority votes, the solution can handle at most $K - \lceil precentage * K \rceil$ wrong-doers.

6. B should send a signed hello message attached with its certificate. Any U_i respond with a signed hello-response based on random waiting mechanism attached with U_i certificate. B can encrypt the query with a symmetric key encrypted with U_i public key. Then U_i responds with encrypted response.

   Impossible to solve without introducing extra messages if no key is shared in advance. B can preload the certificate of U_{GW}, a gateway of U network. Then the query can be encrypted with the public key of U_{GW}, and U_{GW} can decrypt and encrypt with a shared symmetric key within the U network before broadcasting to all U_i. U_i respond with the random waiting mechanism, a response encrypted based on the answer for question 4.

4 Secure Routing

Exercise 5 Secure inter-domain routing (65 points + 20 extra credit)

Consider the (fictitious) snapshot of inter-domain connectivity of Autonomous Systems (ASes) in Fig. 2 and assume that all implement a combination of RPKI and Route Origin Authorization (ROA) and BGP security (BGPsec).

Let one Relying Party (RP) have the following list of Validated ROA Payloads (VRPs) in its list (recall: tuples *(AS number, ROA prefix, prefix length, maxlength)*. Without *maxlength* in an ROA, the AS is authorised to announce only the specified prefix.

1. $(AS512, 130.229.0.0, 16, 24)$
2. $(AS312, 130.28.0.0, 16, -)$

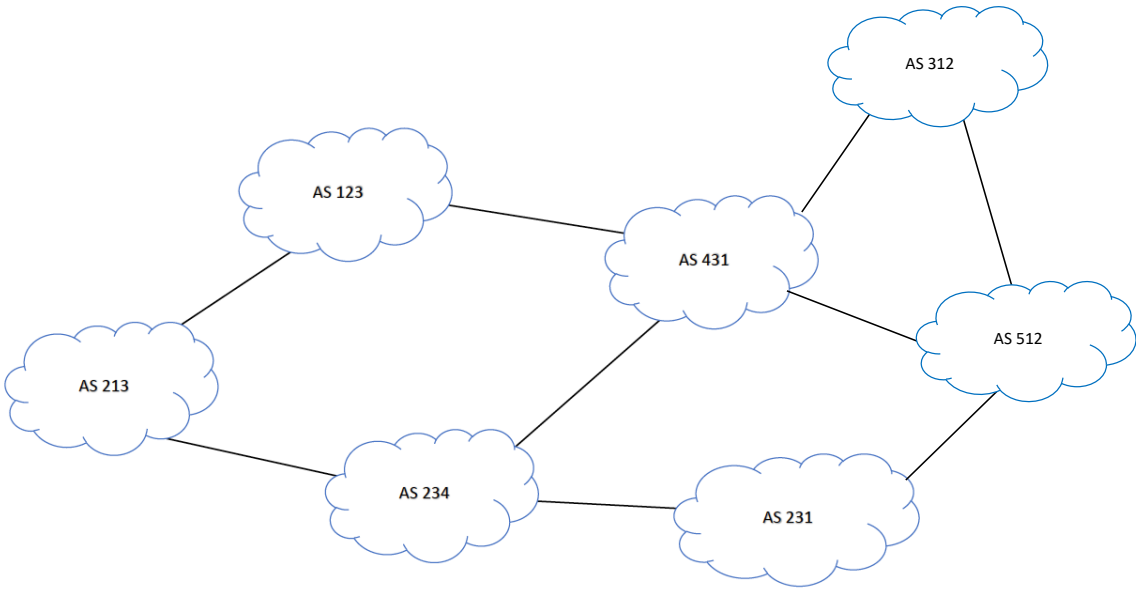

Figure 2: Interconnected AS topology.

The signing routers' public keys are distributed to all ASs with the help of the RPKI. For simplicity, you can assume a single router per AS and use the AS identifier.

1. (30 pt) Please consider AS431 receiving the following BGP announcements:
   - 130.28.0.0/22, AS-PATH: AS312
   - 130.229.0.0/22, AS-PATH: AS512
   - 130.229.0.0/23, AS-PATH: AS234

   Please explain if each of the above, based only on RPKI and ROAs, is valid, invalid or unknown?
2. (35 pt) Please consider now BGPsec, acting together with RPKI.
   (a) (5 pt) Please explain which cryptographic protection BGPsec adds to the above announcements and write the message format for one valid announcement.
   (b) (10 pt) Then, assume the announcement propagates through AS431 to AS123 to AS213. Please write the successive messages by BGPsec.
   (c) (10 pt) Assuming AS123 is adversarial, please explain if it can remove AS431 from the AS-PATH before forwarding the announcement to AS213.
   (d) (10 pt) Can AS123 forward the announcement without removing anything but without adding itself in the AS-PATH?
3. (**extra credit** 20 pt) Now consider AS123 being adversarial and *colluding* with AS231: they have a AS123-AS231 tunnel (communicating directly, in an encrypted manner, e.g., over the AS234 –

AS213 path). Let AS231 tunnel the original announcement of 130.229.0.0/20 (by AS512) to AS123, which in turn forwards it to AS213 (with all the BGPsec cryptographic protection). Please write the messages and explain how AS213 validates this announcement. What if the tunnel is not encrypted?

Answer of exercise 5

1. The BGP announcement can be validated as follows:
   - Although it is *covered* by VRP number 2 It is *invalid*, becuase it is too specific.
   - It is *valid* as it *matches* with VRP number 1 with regard to *maxlength*.
   - Although the IP prefix is *matched* by VRP number 1, the ASN is different, so it is *invalid*.
2. The answer are as follows:
   (a) BGPSec adds digital signature over the AS-PATH, meaning each AS would sign the next authorized AS to propagate the announcement:

   $AS512 : BGP_1 \leftarrow \{130.229.0.0/22, \{AS431, AS512\}\}, \{130.229.0.0/22, \{AS431, AS512\}\}_{Sign_{Priv_{AS512}}}$

   $AS512 \rightarrow AS431 : BGP_1$

   (b) $AS431 : BGP_2 \leftarrow \{130.229.0.0/22, AS123, BGP_1\}, \{130.229.0.0/22, AS123, BGP_1\}_{Sign_{Priv_{AS431}}}$

   $AS431 \rightarrow AS123 : BGP_2$

   $AS123 : BGP_3 \leftarrow \{130.229.0.0/22, AS213, BGP_2\}, \{130.229.0.0/22, AS213, BGP_2\}_{Sign_{Priv_{AS123}}}$

   $AS123 \rightarrow AS213 : BGP_3$

   (c) No, as the next hop is signed, this attack is not possible.

   (d) No, as the next hop is signed, it should add itself, otherwise the next AS inline is not legitimate to propagate the announcement.
3. The original announcement of AS512 to AS231 is as follows:

   $AS512 : BGP_1 \leftarrow \{130.229.0.0/22, \{AS231, AS512\}\}, \{130.229.0.0/22, \{AS231, AS512\}\}_{Sign_{Priv_{AS512}}}$

   $AS512 \rightarrow AS231 : BGP_1$

   Given that AS231 is authorized by AS512 to propagate its anouncement in the network, it can simply sign AS123 as the next hop, and send the BGP announcement through the tunnel:

   $AS231 : BGP_2 \leftarrow \{130.229.0.0/22, AS123, BGP_1\}, \{130.229.0.0/22, AS123, BGP_1\}_{Sign_{Priv_{AS231}}}$

   $AS231 \rightarrow AS123 : BGP_2$

   With this announcement, AS123 then signs AS213 as the next hop and introduces this ASPATH in the network.

   $AS123 : BGP_3 \leftarrow \{130.229.0.0/22, AS213, BGP_2\}, \{130.229.0.0/22, AS213, BGP_2\}_{Sign_{Priv_{AS123}}}$

   $AS123 \rightarrow AS213 : BGP_3$

Since all the signature are valid, AS123 would accept this path as valid with respect to RPKI and ROA and BGP security (BGPsec).

Even if the tunel is not encrypted, the ASes would not realise the attack, as it would be considered the communication between the colluding nodes and not the BGP announcement in the network.