Database Security

E-Clinique: Database Design and Security

Mocanu Alexandru
University of Bucharest, Security and Applied Logics, Group 510

January 2022

1 Introduction

1.1 A short description of the model and its rules

:

- E-Clinique is a system for managing patients' appointments at a healthcare clinic.
- The clinic is comprised of multiple medical sections (such as neurology, ophthalmology etc.).
- Multiple doctors can be part of one section. A medical section must also have one chief doctor. A doctor can see patients each work day for a given time interval (which is the same each day).
- The clinic provides multiple procedures (such as echocardiography, CT imaging etc.) for each medical section. A procedure must be performed by a doctor which is a member of that medical section.
- Patients can register in the system and make appointments for a certain procedure. Each appointment lasts for one hour.
- At the end of an appointment, a doctor may write a prescription for the consulted patient.

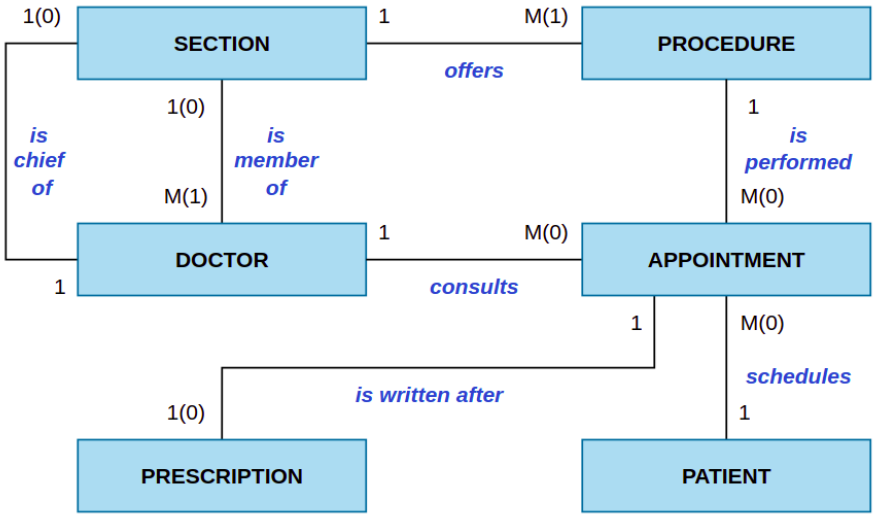

Figure 1: The Entity-Relation diagram.

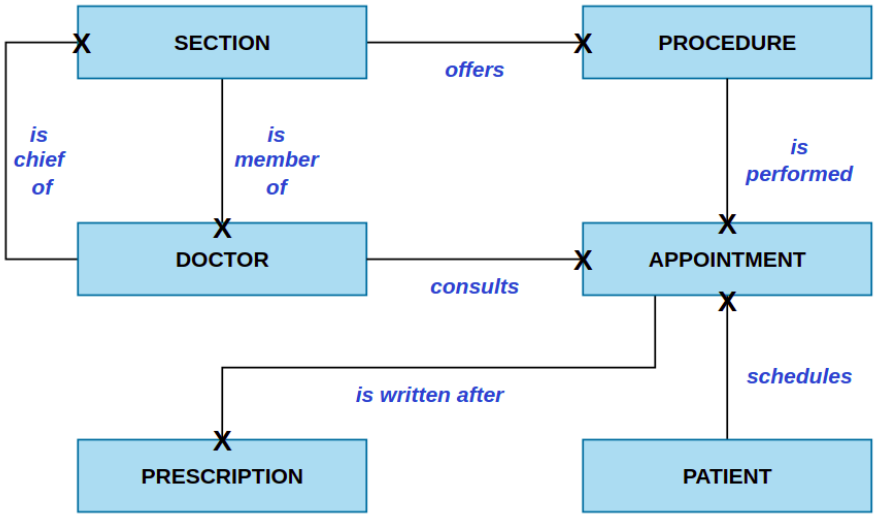

Figure 2: The Conceptual diagram.

1.2 Relation schemas:

DOCTORS (doctor_id#, doctor_name, pnc, section_id, start_hour, end_hour, phone_number, email)

PATIENTS (patient_id#, patient_name, pnc, sex, age, phone_number, email)

SECTIONS (section_id#, section_name, chief_id)

PROCEDURES (procedure_id#, procedure_name, description_, section_id, price)

APPOINTMENTS (appointment_id#, patient_id, doctor_id, procedure_id, appointment_day, appointment_hour)

PRESCRIPTIONS (prescription_id#, appointment_id, details, compensation_percentage, available_until)

1.3 Report outline

In section 2 we will present how table entries were encrypted and how we can verify their integrity. Next, section 3 will present which actions are monitored for audit. Section 4 shows the users, the processes of the application. Also, the process-user, entity-process and entity-user matrices were constructed. The storage allocation per role, session constraing and CPU allocation are also noted. Section 5 showcases how roles, privileges and users were created and associated. In section 6, we show how the application context was used and also give an example of an SQL injection attack. Finally, in section 7 we present a package of function used for data masking.

2 Data encryption

The encryption strategy is first shown on the PATIENTS table, but it may very well be used on any other table containing sensitive, personal information, such as the DOCTORS table.

We create a new table in which we will insert the encrypted data. This table has the same structure as the initial table, but with one additional attribute to help verify data integrity. To encrypt an entry, we concatenate the values of all its attributes and compute their MD5 hash value. Then, each of these attributes (including the hash) are individually encrypted using AES128 (chaining mode CBC and padding mode PCKS5).

For the decryption step, we have a similar, reversed mechanism. We can then decrypt each attribute and compute the MD5 value on the concatenation of all decrypted values (except the hash). The hash value we obtain at this step should be equal to the decrypted hash. If not, it means that one of the attributes of this entry was modified sometime after the data was encrypted.

3 Audit

Three methods of audit were exemplified in this project:

- Standard audit: monitoring any SELECT operations on all tables
- Using trigger: monitoring any INSERT or UPDATE operations on APPOINTMENTS
- Using audit policies: monitoring any INSERTS on PROCEDURES

4 User and computational resources management

4.1 The users of the application and what they can be identified by:

- **Doctor** - name, pnc
- **Chief** - name, pnc, section
- **Patient** - pnc
- **Application admin** - pnc
- **The general public** - email

4.2 The processes within the application:

P1: User administration

P2: Medical section administration

P3: Assign/Remove a doctor as chief of a section

P4: Add/Remove a doctor to/from a medical section

P5: Procedure configuration

P6: Appointment configuration

P7: Write/Cancel a prescription

P8: View the schedule of doctors within a section

P9: View the appointment history of a patient

P10: View the prescriptions of a patient

P11: View all medical sections

P12: View all procedures for a section

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12
Doctor							x		x	x	x	x
Chief				x	x		x	x	x	x	x	x
Patient						x			x	x	x	x
Application Admin	x	x	x				x				x	x
General Public											x	x

Table 1: The process-user matrix.

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12
DOCTORS	I,U			U				S				
PATIENTS	I,U								S	S		
SECTIONS		I,U	U								S	S
PROCEDURES					I,U							S
APPOINTMENTS						I,U		S	S			
PRESCRIPTIONS							D,I,U			S		

Table 2: The entity-process matrix (D=delete, I=insert, S=select, U=update).

	Doctor	Chief	Patient	App Admin	General Public
DOCTORS	S	S,U	S	I,S,U	
PATIENTS	S	S	S	I,U	
SECTIONS	S	S	S	I,S,U	S
PROCEDURES	S	I,S,U	S	S	S
APPOINTMENTS	S	S	I,S,U		
PRESCRIPTIONS	D,I,S,U	D,I,S,U	S	D,I,S,U	

Table 3: The entity-user matrix (D=delete, I=insert, S=select, U=update).

4.3 Resource management

Storage allocation: The admin has unlimited resources, because the majority of the tables can be found in his schema. For doctors, 10M were allocated because they have the PRESCRIPTIONS table in their schema. Patients and Guests do not have any storage resources allocated because their schemas are empty.

Session constraints:

- For Guests - 3 sessions per user are allowed, idle time limit is 5 minutes and connection time limit is 30 minutes.
- For Doctors, Chiefs, Patients - 1 session per user (they each have a personal account), the password must be changed at least every 90 days, at most 5 login failures and 6000 CPU per call limit.

CPU allocation per group: When the CPU is overloaded (100% CPU usage), we have defined rules that limit the CPU usage for the following groups as follows:

20% Admin

35% Doctors

35% Patients and Guests
10% Others

5 Privileges and roles

First we have defined the roles of Doctor, Chief, Patient and Guest and given them privileges exactly as shown in table 3. As an important note, a role for the Admin was not created because he is a unique user.
Next, the following users were created:
clinic_doctor1, clinic_doctor2, clinic_chief3, clinic_patient1, clinic_patient2, clinic_patient3, clinic_guest and clinic_admin.

Finally, for each user their corresponding role was associated:
- Role of Doctor: *clinic_doctor1, clinic_doctor2*
- Role of Chief: *clinic_chief3*
- Role of Patient: *clinic_patient1, clinic_patient2, clinic_patient3*
- Role of Guest (General Public): *clinic_guest*

An important remark is that Doctors have a PRESCRIPTION table in their schema.
clinic_doctor1 and *clinic_chief3* have access only to the prescriptions in their schema, while only *clinic_admin* has access to all of them.

6 Data security

Application context: A Patient can only make appointments between 8:00 and 22:00. In order to verify that, we save this information in the application context.

SQL injection: We have a function that allows a Patient to view his appointments. We have shown that an SQL injection attack leads that patient to view all the appointments, including the ones to which he shouldn't have access to.

7 Data masking

A package called *pack_masking* was defined to include multiple functions which can be applied on our tables:

- *f_masking_name*: the name includes all surnames and first names; only their initials are kept, while the other letters are transformed into '*'.
- *f_masking_pnc*: only the first digit is kept; the rest are randomly replaced by other digits (with no regard to rules like validity of the birth date).
- *f_masking_phone*: only the first three characters are kept; the rest are randomly replaced by other digits.
- *f_masking_email*: we keep the first two and the last two characters before the '@', the rest are replaced with '*'; also, all characters from '@' to the end of the email are kept as they are.

- *f_masking_id*: we keep only the first digit, the rest are randomly generated (important note - if we give the same id twice, the function gives the same result, it is deterministic; if we give two different ids which share the same first digit, the results will also be different)

Note: The id masking function is also applied to foreign keys, not only on the primary key. All the masking functions described above are applied on the DOCTORS table. The id masking function is also applied on the SECTIONS and APPOINTMENT tables. All other functions are applied on the PATIENTS table.