Cover Page

Department of Computer Science

Examination paper for TDT4237 Software Security and Data Privacy

Examination date: 01.06.2024

Examination time (from-to): 09:00-13:00

Permitted examination support material: E: No support material is allowed.

Academic contact during examination: Per H. Meland
Phone: +4741108148

Academic contact present at the exam location: NO

INFORMATION ABOUT THE EXAM

Answers can be done in English or Norwegian.

Secure Code Warrior: Some of the code examples and questions are taken from Secure Code Warrior.

OTHER INFORMATION

Get an overview of the question set before you start answering the questions.

Read the questions carefully and make your own assumptions. If a question is unclear/vague, make your own assumptions and specify them in your answer. The academic person is only contacted in case of errors or insufficiencies in the question set. Address an invigilator if you suspect errors or insufficiencies. Write down the question in advance.

No hand drawings: This exam does not include hand drawings. If you receive hand drawing sheets, this is by mistake. **You will not be able to submit the sheets, and they will not be graded.**

Weighting: The weight of each question is on the question. Regarding the **Closed-Ended questions (1 point for each question if the answer is correct, 0 point if the answer is wrong).**

Notifications: If there is a need to send a message to the candidates during the exam (e.g. if there is an error in the question set), this will be done by sending a notification in Inspera. A dialogue box will appear. You can re-read the notification by clicking the bell icon in the top right-hand corner of the screen.

Withdrawing from the exam: If you become ill or wish to submit a blank test/withdraw from the exam for another reason, go to the menu in the top right-hand corner and click “Submit blank”. This cannot be undone, even if the test is still open.

Access to your answers: After the exam, you can find your answers in the archive in Inspera. Be aware that it may take a working day until any hand-written material is available in the archive.

1 Case description (30 points)

Read the case description and tasks from the PDF and answer the 10 tasks below (use numbering):
Fill in your answer here

Format

Words: 0

Maximum marks: 30

2 Strategy to mitigate the security compromise impact (4 points)

Suppose your system takes users' input and can be exposed to injection attacks. List and explain at least four strategies to mitigate the impact of injection attack compromises. (4 points)

Fill in your answer here

Format

Words: 0

Maximum marks: 4

3 Security and Integrity of OTP (4 points)

1. Explain encryption and decryption algorithm of One Time Pad (OTP). (1 point)
2. Explain why it is insecure to use the same key to encrypt two or several messages using OTP. (2 points)
3. Explain why OTP cannot guarantee integrity (1 point)

Fill in your answer here

Format

Words: 0

Maximum marks: 4

4 AI and security (4 points)

What is the difference between malicious abuse of AI and malicious use of AI? Give examples of both.

Fill in your answer here

Format

Words: 0

Maximum marks: 4

5 Privacy-related question (4 points)

How can a controller demonstrate data protection? Give at least 5 examples.

Fill in your answer here

Format

Words: 0

Maximum marks: 4

6 Code to setup password policy (4 points)

Suppose the password policy of a system is as follows.

- The password is between 8-10 characters long
- The password contains characters from 3 of the following 4 categories:
  - standard uppercase characters (A - Z)
  - standard lowercase characters (a - z)
  - numbers (0 - 9)
  - symbols: only from among ! % - _ + = [ ] { } : , . ? < > ( ) ;
- The password **does not** contain information identical to user's first and last name
- The password **does not** contain common passwords
- **Spaces and the letters "æ", "ø" and "å" are not accepted**

Your task is to develop code and configure Password Validators in Django to check the policy.

Here is an old version of the code that partly takes care of the policy:

```
AUTH_PASSWORD_VALIDATORS = [
    {
        'NAME': 'django.contrib.auth.password_validation.MinimumLengthValidator',
        'OPTIONS': {
            'min_length': 10,
        }
    {
        'NAME': 'password_validators.validators.UppercaseValidator',
    },
    {
        'NAME': 'password_validators.validators.LowercaseValidator',
    },
    {
        'NAME': 'password_validators.validators.SymbolValidator',
    },
    {
        'NAME': 'password_validators.validators.NoNorValidator',
    },
]

class UppercaseValidator(object):
    def validate(self, password, user=None):
        if not re.findall('[A-Z]', password):
            raise ValidationError(
                _("The password must contain at least 1 uppercase letter, A-Z."),
                code='password_no_upper',
            )

class SymbolValidator(object):
    def validate(self, password, user=None):
        if not re.findall('[()[\]{}|\\`~!@#$%^&*_\-+=;:\'",<>./?]', password):
            raise ValidationError(
                _("The password must contain at least 1 special character: " +
                  "()[]{}|\`~!@#$%^&*_-+=;:'\",<>./?"),
                code='password_no_symbol',
            )
```

Your task is to update the old code and add the necessary code to check the password policy. (4 points)
(Note: Syntax errors are allowed, especially if you explain the code.)

Maximum marks: 4

7 Circuit breaker pattern (4 points)

What is the purpose of the circuit breaker pattern in the context of microservice architecture security?

Fill in your answer here

Format

Words: 0

Maximum marks: 4

8 Authorization (4 points)

1) In the Bell-LaPadula model, what does the * property mean?
2) What about STRONG * ?

Fill in your answer here

Format

Words: 0

Maximum marks: 4

9 Concept drift (4 points)

What is the concept drift challenge within AI? Give an example.

Fill in your answer here

Format

Words: 0

Maximum marks: 4

10 Penetration Testing (4 points)

What are the pros and cons of penetration testing tools?

Fill in your answer here

Format

Words: 0

Maximum marks: 4

11 Social Engineering (4 points)

Mention principles of persuasion that can be used for social engineering attacks.

Fill in your answer here

Format

Words: 0

Maximum marks: 4

12 Security logging vulnerability fixing (4 points)

production.py

```
from app.settings.common import *
from decouple import config, Csv


# Email admins and managers
# https://docs.djangoproject.com/en/2.1/howto/deployment/checklist/#admins-and-managers
ADMINS = config('ADMINS', default=[('Admin', 'admin@forum.securecodewarrior.com')], cast=Csv())
MANAGERS = config('MANAGERS', default=[('moderator', 'moderator@forum.securecodewarrior.com')], cast=Csv()


# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = False

# https://docs.djangoproject.com/en/2.1/ref/settings/#allowed-hosts
ALLOWED_HOSTS = ['forum.securecodewarrior.com', ]


# SSL/HTTPS
# https://docs.djangoproject.com/en/2.1/topics/security/#ssl-https

SECURE_SSL_REDIRECT = True

SESSION_COOKIE_SECURE = True

# Cross Site Request Forgery
CSRF_COOKIE_SECURE = True
CSRF_TRUSTED_ORIGINS = config('CSRF_TRUSTED_ORIGINS', default=[], cast=Csv())

SECURE_PROXY_SSL_HEADER = ('HTTP_X_FORWARDED_PROTO', 'https')


# https://docs.djangoproject.com/en/2.1/topics/security/
# HSTS
SECURE_HSTS_SECONDS = 5 * 60
SECURE_HSTS_PRELOAD = True
SECURE_HSTS_INCLUDE_SUBDOMAINS = True

# https://docs.djangoproject.com/en/2.1/ref/clickjacking/
X_FRAME_OPTIONS = 'DENY'

SECURE_CONTENT_TYPE_NOSNIFF = True

SECURE_BROWSER_XSS_FILTER = True

# Sessions
# https://docs.djangoproject.com/en/2.1/ref/settings/#sessions
SESSION_COOKIE_AGE = 24 * 60 * 60  # 24 hours, in seconds
SESSION_EXPIRE_AT_BROWSER_CLOSE = True


# Database
# https://docs.djangoproject.com/en/2.1/ref/settings/#databases

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql',
        'NAME': config('DATABASE_NAME'),
        'USER': config('DATABASE_USER'),
        'PASSWORD': config('DATABASE_PASSWORD'),
        'HOST': config('DATABASE_HOST'),
        'PORT': config('DATABASE_PORT'),
    }
}


# Email configuration
# https://docs.djangoproject.com/en/2.1/ref/settings/#default-from-email

DEFAULT_FROM_EMAIL = 'support@forum.securecodewarrior.com'
SERVER_EMAIL = 'admin@forum.securecodewarrior.com'


# https://docs.djangoproject.com/en/2.1/ref/settings/#email-backend
EMAIL_BACKEND = config('EMAIL_BACKEND')
EMAIL_HOST = config('EMAIL_HOST')
EMAIL_PORT = config('EMAIL_PORT', cast=int)
EMAIL_USE_TLS = config('EMAIL_USE_TLS', cast=bool)
EMAIL_HOST_USER = config('EMAIL_HOST_USER')
EMAIL_HOST_PASSWORD = config('EMAIL_HOST_PASSWORD')


# https://docs.djangoproject.com/en/2.1/ref/settings/#file-upload-permissions
FILE_UPLOAD_PERMISSIONS = config('FILE_UPLOAD_PERMISSIONS', default=0o640, cast=oct)
```

Code snippet of common.py

```
# Logging
# https://docs.djangoproject.com/en/2.1/topics/logging/#configuring-logging
```

```

# Disable Django's logging setup
LOGGING_CONFIG = None

LOGLEVEL = config('LOGLEVEL', default='INFO')

# https://docs.djangoproject.com/en/2.1/topics/logging/#custom-logging-configuration
logging.config.dictConfig({
    'version': 1,
    'disable_existing_loggers': False,
    'formatters': {
        'default': {
            # exact format is not important, this is the minimum information
            'format': '%(asctime)s %(name)-12s %(levelname)-8s %(message)s',
        },
        'django.server': DEFAULT_LOGGING['formatters']['django.server'],
    },
    'handlers': {
        # console logs to stderr
        'console': {
            'class': 'logging.StreamHandler',
            'formatter': 'default',
        },
        'file': {
            'level': 'WARNING',
            'class': 'logging.FileHandler',
            'filename': os.path.join(BASE_DIR, 'debug.log'),
        },
        'django.server': DEFAULT_LOGGING['handlers']['django.server'],
    },
    'loggers': {
        # default for all undefined Python modules
        '': {
          'level': LOGLEVEL,
            'handlers': ['console', 'file'],
        },
        # Prevent noisy modules from logging
        'noisy_module': {
            'level': 'ERROR',
            'handlers': ['console'],
            'propagate': False,
        },
        # Default runserver request logging
        'django.server': DEFAULT_LOGGING['loggers']['django.server'],
    },
})
```

The code snippet common.py has vulnerabilities related to security logging and monitoring. The code in production.py provides some background information.

- Explain which lines in common.py are vulnerable and why they are vulnerable. (2 points)

- Explain how to fix the vulnerabilities in the common.py. It is necessary to provide code to show how to fix it. (2 points) (Note: Syntax errors are allowed, especially if you explain the code.)

Fill in your answer here

Format

Words: 0

Maximum marks: 4

13 Supply chain security (4 points)

What are the four steps of software supply chain attacks and what are the corresponding countermeasure strategies? (4 points)

Fill in your answer here

Format

Words: 0

Maximum marks: 4

14 Microservice architecture attack area (4 points)

What are potential attack areas of microservices deployed on a cloud? (4 points)

Fill in your answer here

Format

Words: 0

Maximum marks: 4

15 Privacy motivation (1 Point)

What is the biggest motivation for software companies to work with privacy?

Select one alternative:

- The respect of their customers
- Catching criminals
- This is what management cares about
- Big legal fines
- This is what developers care about

Maximum marks: 1

16 Cryptography keys (1 point)

Which of the following methods is NOT a recommended approach for generating cryptographic keys?

Select one alternative:

- Reusing a previously generated key for a new encryption task
- Collecting entropy from user-generated input, such as mouse movements or keyboard strokes.
- Deriving keys from a passphrase using a key derivation function
- Employing a software-based secure pseudo-random number generator with unique seeds
- Using a hardware random number generator

Maximum marks: 1

17 Threat modeling (1 point)

What is the best way of performing threat modeling?

Select one alternative:

- Attack trees were the first and is still the most recognized way of modeling threats.
- It is better to create multiple threat modeling representations because there is no single ideal view
- DFD is the most widely used threat modeling technique and should therefore be used
- You should create your own threat modeling technique that is tailored for the job.
- Misuse case diagrams were invented at NTNU and considered to be the most useful way.

Maximum marks: 1

18 Injection vulnerability in the code (1 point)

cart.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title> User Cart </title>
</head>
<body>
     {% for item in items %}

         Product Name: {{ item.product_name }}<br>
         Product Id : {{ item.product_id }}<br>
         Transaction Id : {{ item.transaction }}<br>
         Price : {{ item.price }}<br>

     {% endfor %}
<br><br><b><a href="{% url 'shops:logout' %}"> logout</a></b>
</body>
</html>
```

models.py

```
from __future__ import unicode_literals

from django.db import models
from django.contrib.auth.models import User


# Model for shopping order transactions.
class ShoppingTransaction(models.Model):
    order_id = models.CharField(max_length=20)
    date = models.DateTimeField()
    total_amount = models.DecimalField(max_digits=7, decimal_places=2)
    user = models.ForeignKey(User)

    def __str__(self):
        return self.order_id


# Model for products purchased by a customer.
class TransactionDetail(models.Model):
    product_id = models.CharField(max_length=20)
    product_name = models.CharField(max_length=40)
    transaction = models.ForeignKey(ShoppingTransaction)
    price = models.DecimalField(max_digits=7, decimal_places=2)
```

views.py

```
from django.shortcuts import render
from django.http import HttpResponseRedirect
from django.contrib.auth import authenticate, login, logout
from django.contrib.auth.decorators import login_required
from django.core.urlresolvers import reverse
from django.contrib import messages

from shops.models import TransactionDetail, ShoppingTransaction
from shops.forms import LoginForm


# User Login
def user_login(request):
    # Checking the request method
    if request.method == 'POST':
        # Create a form instance and populate it with data from the request:
        form = LoginForm(request.POST)
        # Checking if all the form fields value meet the set criteria
        if form.is_valid():
            # Fetching the username and passwords from POST methods
            user_name = form.cleaned_data['username']
            pass_word = form.cleaned_data['password']
            # Authenticating the user
            user = authenticate(username=user_name, password=pass_word)
            # Checking if the user is successfully authenticated
            if user is not None:
                # Login the user and creating a user session
                if user.is_active:
                    login(request, user)
                    return HttpResponseRedirect(reverse('shops:order'))
                else:
                    messages.error(request, 'User is disabled.')
            else:
                form = LoginForm()
                # Flashing a message on incorrect login credentials
                messages.error(request, 'Incorrect Login Details .. Please try again')
    else:
        # Creating a form instance
        form = LoginForm()

    return render(request, 'shops/login.html', {'form': form})

```

```
43
44 # Fetching all the orders for that user
45 @login_required(login_url='/shops/login/')
46 def user_order_view(request):
47     data_set = ShoppingTransaction.objects.filter(user=request.user)
48     return render(request, 'shops/order.html', {'orders': data_set})
49
50
51 # Order details page
52 @login_required(login_url='/shops/login/')
53 def user_cart_view(request, transaction_id):
54     data_set = TransactionDetail.objects.filter(transaction=transaction_id)
55     return render(request, 'shops/cart.html', {'items': data_set})
56
57
58 # Logout Page
59 @login_required(login_url='/shops/login/')
60 def user_logout(request):
61    logout(request)
62    return render(request, 'shops/logout.html', {})
```

In the above code snippets, which lines are vulnerable?

Select one alternative:

- models.py: 22-23
- views.py: 54-55
- cart.html: 10-11
- models.py: 11-12

Maximum marks: 1

19 CIA triad (1 point)

Which of the following principles is part of the CIA triad?

Select one alternative:

- Auditability: Enables monitoring and recording of system activities for security analysis.
- Availability: Ensures that authorized users can access data when needed.
- Accountability: refers to the principle that an individual is entrusted to safeguard and control equipment, keying material, and information.
- Attacks: involve direct actions against a system or network.
- Authentication: Verifies the identity of users or processes.

Maximum marks: 1

20 Authorization vulnerability in the code (1 point)

models.py

```
from __future__ import unicode_literals

from django.db import models
from django.contrib.auth.models import User


# Model for team participating in competition.
class Team(models.Model):
   name = models.CharField(max_length=15)

   def __str__(self):
       return self.name


# Model for gamer profiles.
class GamerProfile(models.Model):
   alias_name = models.CharField(max_length=40)
   game_name = models.CharField(max_length=30)
   score = models.IntegerField()
   team = models.ForeignKey(Team)
   user = models.ForeignKey(User)

   def __str__(self):
       return self.alias_name
```

views.py

```
from django.shortcuts import render
from django.contrib.auth import authenticate, login, logout
from django.core.urlresolvers import reverse
from django.http import HttpResponseRedirect, HttpResponse
from django.contrib import messages
from django.contrib.auth import decorators
from django.shortcuts import get_object_or_404

from games.models import GamerProfile, Team
from games.forms import LoginForm


# User login process
def user_login(request):
   # Checking the request method
   if request.method == 'POST':
       # Create a form instance and populate it with data from the request
       form = LoginForm(request.POST)
       if form.is_valid():
           # Fetching the username and passwords from POST methods
           user_name = form.cleaned_data['username']
           pass_word = form.cleaned_data['password']
           # Authenticating the user
           user = authenticate(username=user_name, password=pass_word)
           # Checking if the user is successfully authenticated
           if user is not None:
               # Login the user and creating a user session
               if user.is_active:
                   login(request, user)
                   return HttpResponseRedirect(reverse('games:dashboard'))
               else:
                   messages(request, 'User is disabled.')
           else:
               form = LoginForm()
               messages.error(request, 'Incorrect Login Details. Please try ag ain')
   else:
       # Instantiating empty form
       form = LoginForm()

   return render(request, 'games/login.html', {'form': form})


# User gaming dashboard
@decorators.login_required(login_url='/games/login/')
def dashboard(request):
   team = get_object_or_404(Team, user=request.user)
   team_gamers = GamerProfile.objects.filter(team=team.team)
   return render(request, 'games/dashboard.html', {'team_gamers': team_gamers, })


# User Team members
@decorators.login_required(login_url='/games/login/')
def gamer_profile(request, gamer_id):
   gamer_details = get_object_or_404(GamerProfile, pk=gamer_id)
   return render(request, 'games/gamer_details.html', {'gamer': gamer_details, })


# User logout
@decorators.login_required(login_url='/games/login/')
def log_out(request):
```

```
61    logout(request)
62    return render(request, 'games/logout.html', {})
```

settings.py

```
1 # -*- coding: utf-8 -*-
2
3 #
4 # settings file for production environment
5 #
6 # This settings provides the MINIMUM level of security. Additional
7 # settings may be used to hardening the system (not added here because of
8 # potential compatibility issues with the software), like, for example:
9 #
10 #  - SECURE_PROXY_SSL_HEADER
11 #  - SECURE_HSTS_SECONDS
12 #  - SECURE_HSTS_INCLUDE_SUBDOMAINS
13 #  - SECURE_SSL_REDIRECT
14 #  - SECURE_SSL_HOST
15 #
16
17
18 from __future__ import unicode_literals
19
20 import os
21
22 from django.core.exceptions import ImproperlyConfigured
23
24 INSTALLED_APPS = [
25    'django.contrib.admin',
26    'django.contrib.auth',
27    'django.contrib.contenttypes',
28    'django.contrib.sessions',
29    'django.contrib.messages',
30    'django.contrib.staticfiles',
31    'games.apps.GamesConfig',
32 ]
33
34 ROOT_URLCONF = 'website.urls'
35
36 WSGI_APPLICATION = 'website.wsgi.application'
37
38 DEBUG = False
39
40 ALLOWED_HOSTS = [
41    'randomapp.securecodewarrior.com'
42 ]
43
44 CSRF_COOKIE_SECURE = True
45 SESSION_COOKIE_SECURE = True
46
47 try:
48    SECRET_KEY = os.environ['DJANGO__SECRET_KEY']
49
50    DATABASES = {
51        'default': {
52            'ENGINE': 'django.db.backends.postgresql',
53            'NAME': os.environ['DJANGO__DB_NAME'],
54            'USER': os.environ['DJANGO__DB_USER'],
55            'PASSWORD': os.environ['DJANGO__DB_PASSWORD'],
56            'HOST': os.environ['DJANGO__DB_HOST'],
57            'PORT': os.environ['DJANGO__DB_PORT'],
58        }
59    }
60
61 except KeyError, ex:
62    key = ex.args[0]
63    raise ImproperlyConfigured("The environment variable {0} "
64                               "was not found and is required".format(key))
65
66 MIDDLEWARE_CLASSES = [
67    'django.middleware.security.SecurityMiddleware',
68    'django.contrib.sessions.middleware.SessionMiddleware',
69    'django.middleware.common.CommonMiddleware',
70    'django.middleware.csrf.CsrfViewMiddleware',
71    'django.contrib.auth.middleware.AuthenticationMiddleware',
72    'django.contrib.auth.middleware.SessionAuthenticationMiddleware',
73    'django.contrib.messages.middleware.MessageMiddleware',
74    'django.middleware.clickjacking.XFrameOptionsMiddleware',
75 ]
76
77 TEMPLATES = [
78    {
79        'BACKEND': 'django.template.backends.django.DjangoTemplates',
80        'DIRS': [],
81        'APP_DIRS': True,
82        'OPTIONS': {
83            'context_processors': [
84                'django.template.context_processors.debug',
85                'django.template.context_processors.request',
86                'django.contrib.auth.context_processors.auth',
87                'django.contrib.messages.context_processors.messages',
```

```
88             ],
89         },
90     },
91 ]
92
93 AUTH_PASSWORD_VALIDATORS = [
94     {
95         'NAME': 'django.contrib.auth.password_validation.UserAttributeSimilarityValidator',
96     },
97     {
98         'NAME': 'django.contrib.auth.password_validation.MinimumLengthValidator',
99     },
100     {
101         'NAME': 'django.contrib.auth.password_validation.CommonPasswordValidator',
102     },
103     {
104         'NAME': 'django.contrib.auth.password_validation.NumericPasswordValidator',
105     },
106 ]
107
108 STATIC_URL = '/static/'
```

In the above code snippets, which lines of code are vulnerable?
Select one alternative:

- settings.py:71-72
- models.py: 20-21
- views.py:54-54
- views.py:46-47

Maximum marks: 1

21 Crypto vulnerability in the code (1 point)

```
from __future__ import unicode_literals

import hashlib
from datetime import date
from datetime import timedelta

from django.db import models
from django.conf import settings


class Payment(models.Model):
    """
    Represents a payment.

    It could be pending to be processed (`accepted` is None)
    or already processed (`accepted` is True or False).
    """
    description = models.CharField(max_length=50)
    payment_from = models.ForeignKey(settings.AUTH_USER_MODEL, related_name='+')
    payment_to = models.ForeignKey(settings.AUTH_USER_MODEL, related_name='+')
    amount = models.DecimalField(max_digits=9, decimal_places=2)
    accepted = models.NullBooleanField(null=True)


class InvalidSecurityCode(Exception):
    """The provided security code is not valid"""


class SecurityCodeManager(models.Manager):
    @staticmethod
    def encrypt_security_code(plaintext_security_code):
        """
        Encrypt the provided plain-text security code
        :param plaintext_security_code: plain-text security code
        :return: crypted security code
        """
        assert isinstance(plaintext_security_code, unicode)

        hash_inst = hashlib.md5(plaintext_security_code.encode('utf-8'))
        return hash_inst.hexdigest()

    def check_security_code(self, plaintext_security_code):
        """
        Verifies if the provided plain-text security code
        exists in the database, and isn't too old.

        Raises an exception if the code isn't valid.

        :param plaintext_security_code: the security code in plain text
                                        format (as entered by the user)
        :raise InvalidSecurityCode: if code is not valid
        """
        crypted_code = self.encrypt_security_code(plaintext_security_code)

        today = date.today()
        valid_from_date = today - timedelta(days=15)

        # control date and ignore time, we no need such precision
        qs = self.filter(created_at__date__gte=valid_from_date)
        qs = qs.filter(crypted_password=crypted_code)

        if not qs.exists():
            raise InvalidSecurityCode()

    def delete_old_security_codes(self):
        """
        Delete old security codes from the database.

        This should be called periodically to avoid having
        old codes in the database.
        """
        today = date.today()
        valid_from_date = today - timedelta(days=15)

        self.filter(created_at__date__lt=valid_from_date).delete()


class SecurityCode(models.Model):
    """
    Represents a security code to be entered by the user
    to prove the authenticity when processing a payment.
    """
    created_at = models.DateTimeField(auto_now_add=True)
    crypted_password = models.CharField(max_length=1000, db_index=True)

    objects = SecurityCodeManager()

    def set_security_code(self, plaintext_code):
```

```
89          """
90          Crypt and set the security code in this instance, based on the
91          provided plain-text security code.
92
93          Use of this method must call save() to update the database.
94          """
95
96          if len(set(list(plaintext_code.lower()))) < 10:
97              raise InvalidSecurityCode("The provided text is "
98                                        "not valid and can not be used as "
99                                        "a security code")
100
101          self.crypted_password = SecurityCodeManager.encrypt_security_code(
102              plaintext_code)
103
104     def __str__(self):
105          return "Security code {} ({})".format(self.id, self.created_at)
```

Which lines of the above code snippet are vulnerable?

Select one alternative:

- 39-40
- 83-84
- None of the above listed lines are vulnerable.
- 72-75

Maximum marks: 1

22 Security guiding principle (1 point)

Which security guiding principle is related to the blacklisting countermeasure?

Select one alternative:

- Practice defense in depth
- Keep it simple
- Promote privacy
- Keep it difficult
- Be reluctant to trust

Maximum marks: 1

23 OWASP vulnerability in the code (1 point)

```
index.html
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>Random App</title>
6 </head>
7 <body>
8 {# Message notifications #}
9    {% if messages %}
10       <ul class="messages">
11              {% for message in messages %}
12                 <li {% if message.tags %} class="{{ message.tags }}"{% endif %} >{{ message }}</li>
13              {% endfor %}
14       </ul>
15    {% endif %}
16    {# Message notifications end #}
17    <h4>Check host status</h4>
18         <form  method="POST" action="{% url 'host:index' %}">
19             {% csrf_token %}
20              {{ form }}
21          <br />
22             <input type="submit" value="submit" />
23         </form><br>
24    <hr>
25    <br>
26    {% if request.POST %}
27       <h4>{{ output }}</h4>
28    {% endif%}
29    <br>
30
31 </body>
32 </html>
```

```
forms.py
1 from django import forms
2
3
4 # Form architecture to find host status
5 class HostCheckForm(forms.Form):
6     ip = forms.CharField()
```

```
views.py
1 from django.shortcuts import render
2
3 from host.forms import HostCheckForm
4
5 from os import popen2
6
7
8 # View method to check host status
9 def index(request):
10     output = None
11     # Checking request method
12     if request.method == 'POST':
13         # Initialising form with POST request
14         form = HostCheckForm(request.POST)
15         # Validating form inputs
16         if form.is_valid():
17             cmd_string = 'ping -c 3 ' + form.cleaned_data['ip']
18             process_output = popen2(cmd_string, mode='r', bufsize=-1)
19            output = process_output.__getitem__(1).read()
20     else:
21         # Initialising empty form
22         form = HostCheckForm()
23
24     return render(request, 'host/index.html', {'form': form, 'output': output})
```

In the above code snippets, which lines have vulnerability?
Select one alternative:

- forms.py:5-6
- views.py:14-14
- views.py:16-19
- index.html:26-28

Maximum marks: 1

24 CVSS (1 point)

What does CVSS stand for in the context of cybersecurity?
Select one alternative:

- Cyber Vulnerability Secret Service
- Critical Vulnerability Scoring System
- Cybersecurity Vulnerability Severity Scale
- Common Vulnerability Security System
- Common Vulnerability Scoring System

Maximum marks: 1

25 Another OWASP vulnerability in the code (1 point)

index.html

```
<!DOCTYPE html>
<html>
    <head>
        <title>Random App</title>
    </head>

    <body>
   {# Message notifications #}
   {% if messages %}
      <ul class="messages">
             {% for message in messages %}
                <li>{% if message.tags %} class="{{ message.tags }}"{% endif %}>{{ message }}</li>
             {% endfor %}
      </ul>
   {% endif %}
   {# Message notifications end #}
   <h4>Team Collaborator Calendar</h4>
        <form  method="POST" action="{% url 'teams:index' %}">
            {% csrf_token %}
             Email : {{form.email}}
       {{ form.email.errors }}
       Scheduled Task : {{form.event}}
       {{ form.event.errors }}
      Date : {{form.date}}
       {{ form.date.errors }}


           <br />
            <input type="submit" value="submit" />
        </form><br>
   <hr>
   <br>
   {% if calendar_events %}
      <h4>Latest events</h4>
      {% for event in calendar_events %}
      <b>Email: </b> {{ event.email }}<br>
      <b>Task:</b> {{ event.event }}<br>
      <b>Date:</b> {{ event.date }}<br><br>
      {% endfor %}
   {% endif%}

   <br>
<script src="//code.jquery.com/jquery-1.10.2.js"></script>
<script src="//code.jquery.com/ui/1.11.4/jquery-ui.js"></script>
<script>
  $(function() {
    $( ".datepicker" ).datepicker({
      changeMonth: true,
      changeYear: true,
      yearRange: "1900:2012",

    });
  });
  </script>
    </body>
</html>
```

models.py

```
from __future__ import unicode_literals

from django.db import models


#  Model for Calendar App
class Calendar(models.Model):
    email = models.EmailField()
    date = models.DateField()
    event = models.CharField(max_length=1024)
```

views.py

```
from django.core.urlresolvers import reverse_lazy
from django.shortcuts import render
from django.utils.html import mark_safe
from django.views.generic import CreateView, TemplateView
# mark_safe tells django templates that a string should be used AS IS
from teams.forms import CalendarForm
from teams.models import Calendar


# View for scheduling task form and render scheduled task
class Index(CreateView):
    form_class = CalendarForm
    model = Calendar
    template_name = 'teams/index.html'
    success_url = reverse_lazy('teams:success')

    # Custom function
```

```
18     def get_all_events(self):
19         temp_calendar_events = []
20         for events in Calendar.objects.all().order_by('-date'):
21             events.event = mark_safe(events.event)
22             temp_calendar_events.append(events)
23         return temp_calendar_events
24
25     # Method for form/POST data
26    def get_context_data(self, **kwargs):
27         context = super(Index, self).get_context_data(**kwargs)
28         final_events = self.get_all_events()
29         context['calendar_events'] = final_events
30         return context
31
32
33 # View for redirection
34 class Success(TemplateView):
35     template_name = 'teams/success.html'
```

In the above code snippets, which lines are vulnerable?

Select one alternative:

- model.py:10-10
- index.html:22-23
- views.py:20-22
- views.py:28-28

Maximum marks: 1

26 Public key cryptography algorithms (1 point)

Which statements regarding public key cryptography algorithm are FALSE?

1. Message sender and receiver use identical keys when they use public key cryptography algorithms.
2. The public key cryptography algorithms are usually open to public.
3. Stream cipher is a public key cryptography algorithm.
4. ECDSA is not a public key cryptography algorithm.

Select one alternative:

- 1, 2, and 4
- 1, 3, and 4
- 2, 3, and 4
- All of them

Maximum marks: 1

27 Buffer overflow (1 point)

Which of these kinds of inputs can cause a buffer overflow?
1. An environment variable
2. String input from the user
3. A single integer
4. A floating point number
5. File input

Select one alternative:

- 2 and 5
- All of the above
- 1 and 2
- 3 and 4

Maximum marks: 1

28 Security requirements (1 point)

Which of these is a good security requirement?
Select one alternative:

- The system must have good usability
- End user data should be encrypted at rest
- The system shall work just like the previous one, but on a new platform
- The system should be free from vulnerabilities
- The system shall encrypt all confidential data using the RSA algorithm

Maximum marks: 1

29 Cookies (1 point)

What is the security issue of cookie-based tokens?
Select one alternative:

- Web browser can not save the cookie value
- Cookies are unhealthy for the end user
- Web browser can not see the cookie expiration time
- Server does not see which domain sends the cookie
- Server can not save the cookie value

Maximum marks: 1

30 Static code analysis (1 point)

Which approach does NOT belong to static code analysis for vulnerability detection?
Select one alternative:

- Control flow analysis
- Pattern matching
- Penetration testing
- Taint analysis

Maximum marks: 1

31 Supply chain vulnerability countermeasures (1 point)

Which is NOT a transparency countermeasure of software supply chain security?
Select one alternative:

- Dependabot
- Version Locking
- Sigstore
- Software Bill of Materials (SBOM)

Maximum marks: 1

32 Microservice architecture security countermeasures (1 point)

What is the countermeasure to defend against attacks targeting the load balancer in the microservice architecture?
Select one alternative:

- Service-to-service authentication
- Rate throttling
- Secure container
- Service-level authorization

Maximum marks: 1

Question 1
Attached

Case description: Risk assessment of a risk assessment tool for Air Traffic Management (ATM)

SESAR Joint Undertaking defines, develops and deploys technologies to transform air traffic management (ATM) in Europe. These technologies are known as *solutions* and are developed in numerous projects under det SESAR JU programme. Solutions can be used to manage conventional aircrafts, drones, air taxis and vehicles flying at higher altitudes, and need to undergo risk assessments at various stages of their development lifecycle (i.e. concept development, lab experiments, prototypes in realistic environments, proven system in operation development). One of the solutions is a cyber security risk assessment methodology, that is to be applied to the other solutions in order to derive their security requirements and maintain an up-to-date risk picture. A part of this solution is a web-based tool that is intended to make risk assessments easier for other air traffic management solutions. This includes defining scope and goals, identifying assets, threats and vulnerabilities, describing and evaluating risks and deriving security requirements. Since this web-based tool is considered to be a solution by itself, you will also need to perform a risk assessment of it (effectively a risk assessment of a risk assessment tool).

You have been given the following business goals:

- BG1: Support the ATM risk assessment methodology.
- BG2: A user-friendly web-interface that supports various stakeholders involved.
- BG3: Able to retrieve assets from a digital catalogue of reusable items (e.g. flight data, satellite datalink, primary radar, air traffic controller, passengers.).
- BG4: Preserve confidentiality of the assessments of the SESAR solutions.
- BG5: Allow sharing of risk assessment information between authorized people (involved in an assessment).

Part 1 tasks (30 points in total)

In this part you will perform tasks related web-based tool from the case description.

If you feel that any of the tasks require information that you do not find in the text, then you should:

- Document the necessary assumptions (e.g. technology, standards, software, design choices.)
- Explain why you need them.

Your answers should be brief and to the point.

Task 1: As part of defining the scope, list at least five impact dimensions you consider relevant for this assessment. (3 points)

Task 2: What kind of people/stakeholders would you involve in the assessment? Explain why. (3 points)

Task 3: What kind of access control model would you recommend for this solution? (2 points)

Task 4: Identify 5 assets (something of value that needs protection) related to the tool. (3 points)

Task 5: In the use case diagram on the next page, you can see use cases and undefined actors. Define at least 5 suitable actors and describe how they should be connected referring to the use case labels (you can add more actors if needed). (3 points)

Task 6: Define at least 5 corresponding misuse case elements and describe how they should be connected to the use cases (you use the labels as a references). (3 points)

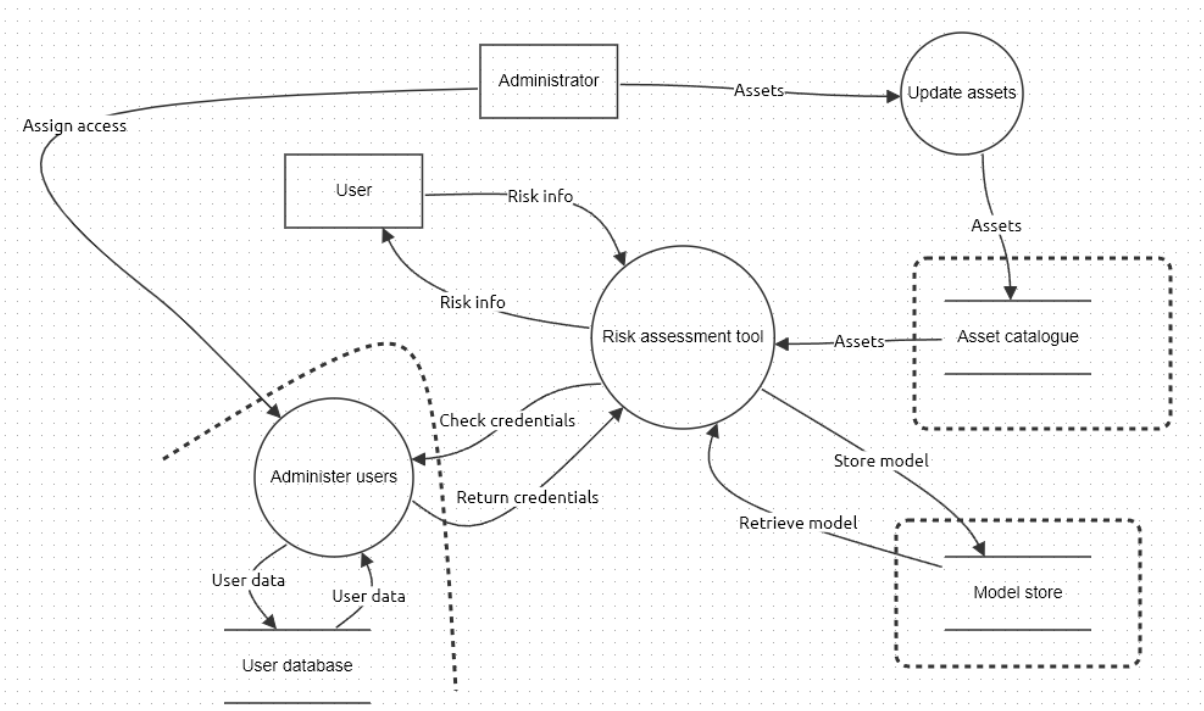
Task 7: The last page shows a DFD. Explain the different types of elements in the diagram. Identify possible attack points in relation to the DFD elements and describe at least 5 threats according to STRIDE. (4 points)

Task 8: Based on the threat models, identify at least 5 technical risks and evaluate them. You should describe necessary assumption related to the technology. (3 points)

Task 9: Select at least 3 technical risks and define one well-formulated security requirement for each. (3 points)

Task 10: Write a short reflection on which threat agents you consider to be significant for this tool. Justify these using principles from attacker-centric threat modelling. (3 points)


UC1: Create risk assessment project for a solution
UC2: Assign access
UC3: Authenticate
U1: ...
U2: ...
U3: ...
U4: ...
U5: ...
UC4: Perform risk assessment
includes
UC5: Define scope
UC6: Identify assets
UC7: Identify threats and vulnerabilities
UC8: Evaluate risks
UC9: Define security requirements
extends
UC10: Retrieve assets
Threat agent


Administrator
Assets
Update assets
Assign access
User
Risk info
Assets
Risk info
Risk assessment tool
Assets
Asset catalogue
Check credentials
Administer users
Return credentials
Store model
Retrieve model
User data
User data
Model store
User database

Case description: Risk assessment of a risk assessment tool for Air Traffic Management (ATM)

SESAR Joint Undertaking defines, develops and deploys technologies to transform air traffic management (ATM) in Europe. These technologies are known as *solutions* and are developed in numerous projects under det SESAR JU programme. Solutions can be used to manage conventional aircrafts, drones, air taxis and vehicles flying at higher altitudes, and need to undergo risk assessments at various stages of their development lifecycle (i.e. concept development, lab experiments, prototypes in realistic environments, proven system in operation development). One of the solutions is a cyber security risk assessment methodology, that is to be applied to the other solutions in order to derive their security requirements and maintain an up-to-date risk picture. A part of this solution is a web-based tool that is intended to make risk assessments easier for other air traffic management solutions. This includes defining scope and goals, identifying assets, threats and vulnerabilities, describing and evaluating risks and deriving security requirements. Since this web-based tool is considered to be a solution by itself, you will also need to perform a risk assessment of it (effectively a risk assessment of a risk assessment tool).

You have been given the following business goals:

- BG1: Support the ATM risk assessment methodology.
- BG2: A user-friendly web-interface that supports various stakeholders involved.
- BG3: Able to retrieve assets from a digital catalogue of reusable items (e.g. flight data, satellite datalink, primary radar, air traffic controller, passengers.).
- BG4: Preserve confidentiality of the assessments of the SESAR solutions.
- BG5: Allow sharing of risk assessment information between authorized people (involved in an assessment).

Part 1 tasks (30 points in total)

In this part you will perform tasks related web-based tool from the case description.

If you feel that any of the tasks require information that you do not find in the text, then you should:

- Document the necessary assumptions (e.g. technology, standards, software, design choices.)
- Explain why you need them.

Your answers should be brief and to the point.

Task 1: As part of defining the scope, list at least five impact dimensions you consider relevant for this assessment. (3 points)

Task 2: What kind of people/stakeholders would you involve in the assessment? Explain why. (3 points)

Task 3: What kind of access control model would you recommend for this solution? (2 points)

Task 4: Identify 5 assets (something of value that needs protection) related to the tool. (3 points)

Task 5: In the use case diagram on the next page, you can see use cases and undefined actors. Define at least 5 suitable actors and describe how they should be connected referring to the use case labels (you can add more actors if needed). (3 points)

Task 6: Define at least 5 corresponding misuse case elements and describe how they should be connected to the use cases (you use the labels as a references). (3 points)

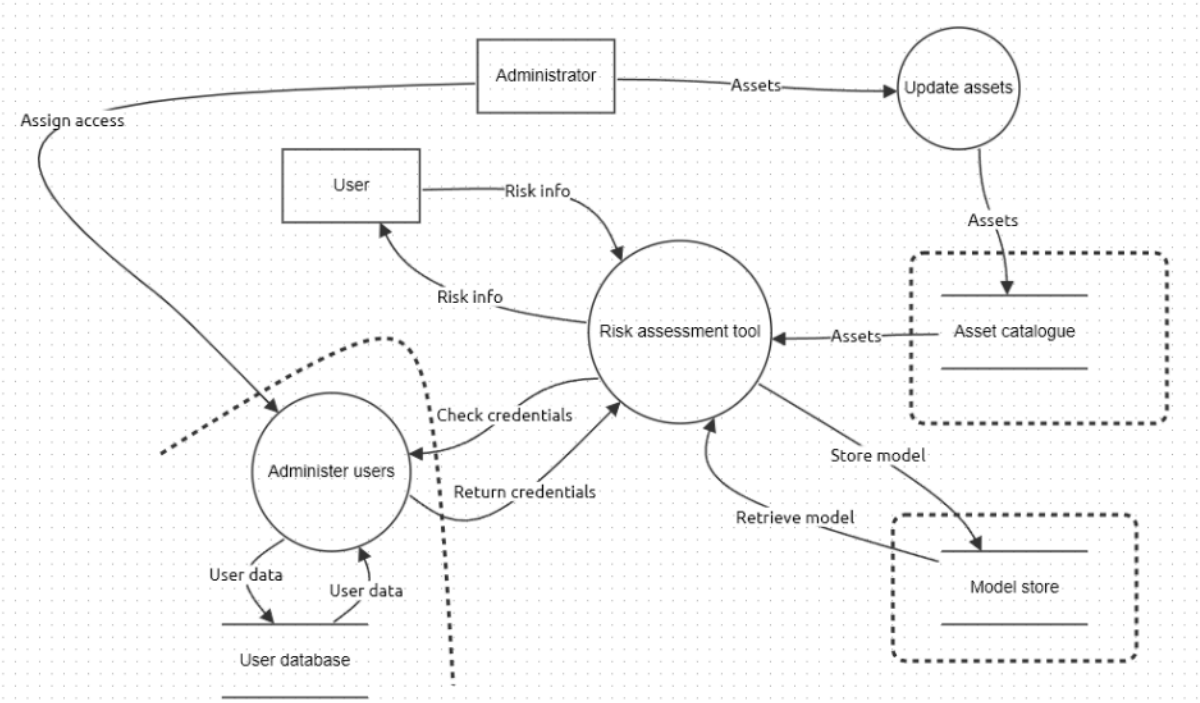
Task 7: The last page shows a DFD. Explain the different types of elements in the diagram. Identify possible attack points in relation to the DFD elements and describe at least 5 threats according to STRIDE. (4 points)

Task 8: Based on the threat models, identify at least 5 technical risks and evaluate them. You should describe necessary assumption related to the technology. (3 points)

Task 9: Select at least 3 technical risks and define one well-formulated security requirement for each. (3 points)

Task 10: Write a short reflection on which threat agents you consider to be significant for this tool. Justify these using principles from attacker-centric threat modelling. (3 points)


UC1: Create risk assessment project for a solution
UC2: Assign access
UC3: Authenticate
U1: ...
U2: ...
U3: ...
U4: ...
U5: ...
UC4: Perform risk assessment
includes
UC5: Define scope
UC6: Identify assets
UC7: Identify threats and vulnerabilities
UC8: Evaluate risks
UC9: Define security requirements
UC10: Retrieve assets
extends
Threat agent


Administrator
Assets
Update assets
Assign access
User
Risk info
Assets
Risk info
Risk assessment tool
Assets
Asset catalogue
Check credentials
Administer users
Return credentials
Store model
Retrieve model
Model store
User data
User data
User database