Q1

a) Privacy. Privacy is violated as Bob couldn't control who gets to see his data. Eve learns his data (password) through shoulder surfing.
b) Integrity. The integrity property is being violated as Eve did a SIM swap attack, and the phone number is not associated with the correct SIM card anymore.
c) Confidentiality. This is a violation of the confidentiality property since Eve could access the information easily without further authentication rather just some words "Jack told me to contact you…". It is easy for unauthorized parties to access the system.
d) Availability. As Eve and Eva cut other employees from accessing the admin page. The availability property is being violated, since other employees can no longer access the admin page whenever they want anymore.

Q2

We could use access control lists (ACL) with role-based access control (RBAC) to protect the resource allocation on the admin page. Each employee has its roles and objects specify which roles get to access the object. Right now everyone gets to access the admin page, they need to specify least privilege where only people needed can access the admin page.

Q3

a) Modification (changed passwords). The threat is a modification threat. Eve change the email addresses of multiple accounts and she also changed the passwords.
b) Interception (getting new data). The threat is an interception threat. She intercepted the data and downloaded them onto her hard drive.
c) Fabrication (propose a scam). The threat is a fabrication threat. She posed (created) fake data onto the website on behalf of celebrities.
d) Interruption (stopped them). The threat is an interruption threat. She stops the users from using their accounts so they cannot make post to inform others about the scams.

Q2

a) Preventing: Eve could refuse to meet the buyer and requests for electronic transfers to cash out to avoid meeting the FBI agents in person.
b) Deflecting: she could go abroad to complete her actions, it will make it less attractive for the FBI to track her, and FBI may be limited with what they are allowed to do in foreign countries. It may be less attractive for them to track her because there is less responsibility when one is outside of the border. In contrast, Biocoin can easily be exchanged anywhere.
c) Deterring: Instead of linking her bitcoin access to the email address on the forum, and exchange's identity verification. Eve could add in multiple layers to protect her identity, for example, multiple email accounts, different forum accounts, and fake ID registered under other's name. So, it will be more expensive for the FBI to track her information. This deters on how FBI finds out her identity.
d) Detecting. Eve could possibly check for the identity of the buyer prior to the meeting to know that they are FBI agents. She could possibly detect that the FBI are not usual

accounts on the forum and detect that they may not be actual buyers and stops interacting with them.

Programming Question

Sploit2.c

```
// Returns numeric gid of user
static gid_t get_gid() {
  char* dir;
  struct passwd* pw;
  gid_t gid;

  gid = 0;
  dir = getenv("HOME");
  setpwent();
  while ((pw = getpwent()) != NULL) {
    if (strcmp(pw->pw_dir, dir) == 0) {
      gid = pw->pw_gid;
      break;
    }
  }
  endpwent();
  return gid;
}
```

```
// Returns numeric uid of user
static uid_t get_uid() {
  char* dir;
  struct passwd* pw;
  uid_t uid;

  uid = 0;
  dir = getenv("HOME");
  setpwent();
  while ((pw = getpwent()) != NULL) {
    if (strcmp(pw->pw_dir, dir) == 0) {
      uid = pw->pw_uid;
      break;
    }
  }
  endpwent();
  return uid;
}
```

In sploit2.c, it exploits an incomplete mediation vulnerability in the get_uid and get_gid function. The function didn't verify the user input as the environment variable can be changed. So, if we were to change the environment variable of HOME, then get_uid and get_gid will return the id 0, which is the root user. Then pwgen will change root's password. We can exploit it by using the password generated by pwgen to log in into the root.

This can be fixed by avoiding using getenv to check for gid. Use the c getuid() and getgid() function instead.

Sploit3.c

```
261     switch (c) {
262       case 'e':
263         if (optarg) {
264           strcpy(args.filename, optarg);
265         } else {
266           res = check_perms();
267           if (res) {
268             snprintf(buffer, 1024, "WARNING: '%s' could not creat
269             fprintf(stderr, buffer);
270           } else {
271             strcpy(args.filename, FILENAME);
272             fill_entropy();
273           }
274         }
```

In sploit3.c, there is a TOCTTOU vulnerability. We notice that in the check_perms() and fill_entropy() function can be exploits with a time difference. We can run the program with -3 and then the program will check if the file is owned by the user, then, wait for the user input. At this point, if we create a symlink to the password file, we can overwrite the root password. Then we can exploit the program by using su root.

We can fix this by performing the check right before fwrite in the fill_entropy function, after we get the user input. Then, the user cannot exploit the because the time of check is immediate before time of use.

Sploit4.c

```
case 'e':
  if (optarg) {
    strcpy(args.filename, optarg);
  } else {
    res = check_perms();
    if (res) {
      snprintf(buffer, 1024, "WARNING: '%s' could not create '%s' because the file already exists! :(\n", argv[0], FILENAME)
      fprintf(stderr, buffer);
    } else {
      strcpy(args.filename, FILENAME);
      fill_entropy();
    }
  }
  break;
```

In sploit4.c, there is a format string vulnerability. In parse_args function, we can exploit the snprintf with buff of size 1024 here.

We can exploit it by putting the format string into argv[0]. Similar to a buffer overflow string, putting in NOPs, shellcode, and format string into the buffer. It would then change the return address and instruction pointer to the shellcode. Thus, we get root access.

We can fix this vulnerability by checking the format string given by the user before hand to make sure it doesn't contain any format string such as %hn,%u, to avoid the attack. We fix this vulnerability by restricting user input.

**code of sploit4.c is not written