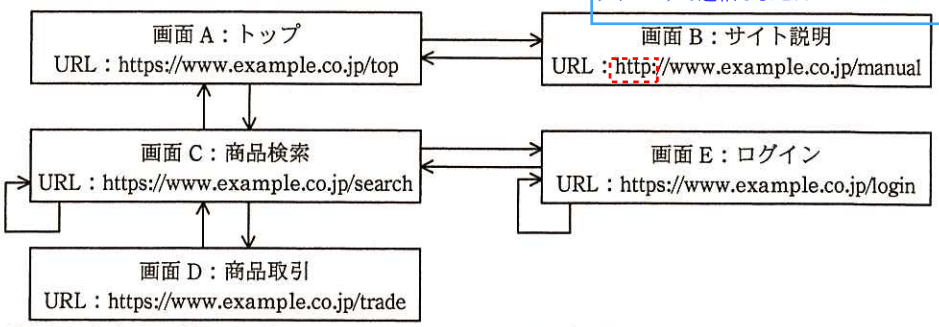
問 1　Web サイトの脆弱性と対策に関する次の記述を読んで，設問 1～3 に答えよ。

S 社は，情報システムの構築，運用，コンサルティングなどのサービスを顧客に提供する従業員数 5,000 名の企業である。S 社にはセキュリティプロフェッショナルグループ（以下，SPG という）という組織があり，Web アプリケーションソフトウェア（以下，Web アプリという）の脆弱性を検査するサービス（以下，脆弱性検査という）を提供している。

SPG では，脆弱性検査に従事できる者を認定するために，技能試験を実施している。技能試験は，Web アプリに脆弱性が作り込まれた，技能試験用の Web サイト（以下，試験用サイトという）を用いて実施される。試験用サイトは，個人間で取引するオークションシステムを想定しており，“http://...” 及び “https://...” の画面がある。図 1 に試験用サイトの画面構成と画面の遷移を，図 2 に画面 C の概要をそれぞれ示す。また，試験用サイトの機能のうち，セッション管理機能の仕組みを図 3 に，検索文字列の引継機能の仕組みを図 4 に示す。

Secureを指定していないので、Cookieが暗号化されないでサーバに送られる (http通信)
Secure設定であれば、https以外は、Cookieをヘッダにつけて送信しません




注記 1　矢印は，ボタン又はリンクのクリックによる画面の遷移を表す。
注記 2　技能試験に関係しない画面は，省略している。

**図 1　試験用サイトの画面構成と画面の遷移**

- 最上部に，画面 E へのリンク，検索フィールド及び検索ボタンがある。
- 検索フィールドに検索文字列を入力した後，検索ボタンをクリックすると，画面 C を再表示し，検索フィールドには入力された検索文字列が，画面の下部には検索された商品一覧が表示される。
- 画面 E へのリンクは，未ログインの状態では “ログイン” と表示されており，クリックすると画面 E に遷移する。画面 E でログインに成功すると遷移前の画面 C が再表示されるが，“ログイン” の表示が “ログイン中” という表示に変わり，クリックはできなくなる。
- ログイン中であって画面下部に商品一覧が表示された状態では，商品一覧中のいずれかの商品をクリックすることによって，画面 D に遷移して商品取引に進むことができる。

**図 2　画面 C の概要**

・Web ブラウザ起動後，試験用サイトの画面の中で最初にアクセスできるのは，画面 A，B だけであり，画面 C～E の URL を指定してアクセスしても，あらかじめ画面 A にアクセスしていないと，画面 A にリダイレクトされる。
・画面 A にアクセスがあると，セッション ID を格納する Cookie（名前を SESSIONID とする）の有無を調べ，ない場合には，試験用サイトが動作するサーバで，セッション ID の値をキーとしたログイン状態を保持するセッションオブジェクトを新規に生成する。その後，サーバは，名前を SESSIONID とする Set-Cookie ヘッダを Web ブラウザに返信する。以後，同一のセッション ID によるアクセスには，同一のセッションオブジェクトが利用され，サーバでセッションが管理される。
・画面 B へのアクセスによってセッション ID が更新されることはない。

**図３　セッション管理機能の仕組み**

・画面 C において，利用者が検索フィールドに文字列を入力した後，又は検索フィールド中の文字列を書き換えた後，検索ボタンをクリックすることによって，検索フィールドの文字列が，クエリ文字列のパラメタとして URL エンコードされた状態でサーバに送信され，サーバで商品検索が実行される。
・商品検索が実行されると，サーバは応答時に，名前が KENSAKU であり，値を検索文字列とする Set-Cookie ヘッダを返す。
・Web ブラウザは，名前が KENSAKU である Set-Cookie ヘッダを受け取ると，当該 Cookie を更新するとともに，以後のリクエストヘッダに含めて毎回当該 Cookie を送信する。
・他の画面から画面 C に戻ったときに，既に商品検索が実行されていた場合には，リクエストヘッダ中の当該 Cookie の値に基づいて，商品検索の最新の実行結果及び検索フィールドが表示される。

**図４　検索文字列の引継機能の仕組み**

〔技能試験〕

SPG に新たに配属された従業員は研修検査員と呼ばれ，2 か月間の研修を終えると，技能試験を受ける。技能試験に合格した者だけが脆弱性検査を実施できる規則になっている。技能試験では，脆弱性検査専用の Web ブラウザを使用する。検査専用の Web ブラウザには，一般的な Web ブラウザの機能に加えて，送信する HTTP リクエストやパラメタの値を意図的に変更する機能がある。

技能試験では，試験官とのディスカッション，検査手順や報告書の作成を通して，力量が判断され，合否が決定される。このたび，研修検査員の T 君が，技能試験を受けることになった。試験官は U 主任である。最初に，U 主任は，技能試験の前提として図１及び図２の内容を T 君に伝えた。

なお，図３及び図４の内容は T 君には伏せられている。

〔検査シナリオと HTTP ヘッダ〕

技能試験の開始に当たって，U 主任は，図５に示す検査シナリオの順番で画面にアクセスするよう T 君に指示した。T 君が検査シナリオを実行した際の HTTP リクエ

ストヘッダ及びHTTP レスポンスヘッダを，図6に示す。

検査専用の Web ブラウザを起動→画面 A→画面 B→画面 A→画面 C→画面 C→画面 E→画面 C→画面 D

**図5　検査シナリオ**

リクエストX

```
01 GET /top HTTP/1.1
02 Accept: text/html,
   application/xhtml+xml, */*
03 Accept-Language: ja-JP
04 Accept-Encoding: gzip, deflate
05 Host: www.example.co.jp
06 Connection: Keep-Alive
```

→

レスポンスX

```
07 HTTP/1.1 200 OK
08 Date: Tue, 10 Jun 2014 05:30:31 GMT
09 Set-Cookie: SESSIONID=134D96E470da240421svr5B019;
   Expires= Wed, 11-Jun-2014 05:30:31 GMT;
   domain=example.co.jp;
10 Expires: Thu, 19 Nov 1981 08:52:00 GMT
11 Cache-Control: no-store, no-cache
```

リクエストY

```
12 GET /manual HTTP/1.1
13 Accept: text/html, application/xhtml+xml, */*
14 Accept-Language: ja-JP
15 Accept-Encoding: gzip, deflate
16 Host: www.example.co.jp
17 Cookie: SESSIONID=134D96E470da240421svr5B019
```

→

レスポンスY

```
18 HTTP/1.1 200 OK
19 Date: Tue, 10 Jun 2014 05:32:16 GMT
20 Expires: Thu, 19 Nov 1981 08:52:00 GMT
21 Cache-Control: no-store, no-cache
```

リクエストZ

```
22 GET /trade?itemid=10 HTTP/1.1
23 Accept: text/html, application/xhtml+xml, */*
24 Accept-Language: ja-JP
25 Accept-Encoding: gzip, deflate
26 Host: www.example.co.jp
27 Cookie: SESSIONID=134D96E470da240421svr5B019
28 Cookie: KENSAKU=jewelry
```

→

レスポンスZ

```
29 HTTP/1.1 200 OK
30 Date: Tue, 10 Jun 2014 05:45:58 GMT
31 Expires: Thu, 19 Nov 1981 08:52:00 GMT
32 Cache-Control: no-store, no-cache
```

**注記1**　リクエストXとレスポンスXは，最初に画面Aを表示した際のHTTP ヘッダである。
**注記2**　リクエストYとレスポンスYは，画面Aから画面Bに遷移した際のHTTP ヘッダである。
**注記3**　リクエストZとレスポンスZは，画面Cから画面Dに遷移した際のHTTP ヘッダである。
**注記4**　リクエスト及びレスポンス中の行番号は，本図中における通し番号である。

**図6　検査シナリオを実行した際のHTTP ヘッダ（抜粋）**

〔脆弱性に関するディスカッション〕

次は，図6に関するU主任とT君のディスカッションである。

U 主任：図 6 中のレスポンス X について，セキュリティ上，気になる点があれば指摘してください。

T 君　：Set-Cookie ヘッダに，①secure 属性が設定されていません。secure 属性を設定しないと，セッション ID を第三者に盗聴されるリスクがあり，セッションハイジャックなどにつながると思います。

U 主任：他に，図 6 全体を通して，気になる点はありますか。

T 君　：HTTP ヘッダインジェクションの脆弱性が存在する可能性があります。図 7 の検査コードをリクエストのクエリ文字列のパラメタの値にセットし，スクリプトの実行ができるかどうか，確認してみます。

CRLF(%0d%0a)

```
a %3chtml%3e%3cbody%3e%3cscript%3ealert%28%221%22%29%3c%2fscript%3e%3c%2fbody%3e%3c
%2fhtml%3e
```

注記 1　図中の文字列は URL エンコード済みの形式である。

注記

HTTPヘッダインジェクション
%Od%Oa」は,エンコードすると「CRLF」(改行コード) になります。
これが2回繰り返されているので1行空白行ができて、その後に(これもエンコードすると) スクリプトタグが続くことになります。
HTTP では, ヘッダとボディを分けるのは “空白行” なので,これをそのまま解釈すると, このスクリプトはレスポンスボディになり、本来は**ヘッダ部なので実行されないスクリプトが, レスポンスボディなので実行されてしまうというわけです**。

このケースの HTTPヘッダインジェクションに対しては,
・ヘッダ出力用の関数や API を使用
・出力文字列に改行コードが含まれる時は,それを無効化してエラー画面を出力
・改行コード以降の文字列を削除するように実装します。

| 文 | | | | | | | |
|---|---|---|---|---|---|---|---|
| NU | | | | | | | |
| SO | | | | | | | |
| STX | 02 | DC2 | 12 | " | 22 | 2 | 32 |
| ETX | 03 | DC3 | 13 | # | 23 | 3 | 33 |
| EOT | 04 | DC4 | 14 | $ | 24 | 4 | 34 |
| ENQ | 05 | NAK | 15 | % | 25 | 5 | 35 |
| ACK | 06 | SYN | 16 | & | 26 | 6 | 36 |
| BEL | 07 | ETB | 17 | ' | 27 | 7 | 37 |
| BS | 08 | CAN | 18 | ( | 28 | 8 | 38 |
| HT | 09 | EM | 19 | ) | 29 | 9 | 39 |
| LF | 0a | SUB | 1a | * | 2a | : | 3a |
| VT | 0b | ESC | 1b | + | 2b | ; | 3b |
| FF | 0c | IS4 | 1c | , | 2c | < | 3c |
| CR | 0d | IS3 | 1d | - | 2d | = | 3d |
| SO | 0e | IS2 | 1e | . | 2e | > | 3e |
| SI | 0f | IS1 | 1f | / | 2f | ? | 3f |

注記　文字コード 00～1f は制御文字である。文字コード 20 は空白文字である。

図 8　ASCII 文字コード一覧（抜粋）

T 君が，図 7 の検査コードを用いて確認したところ，予想どおり，警告ダイアログが表示された。T 君は，HTTP ヘッダインジェクションの脆弱性が存在することを指摘した。

T 君　：サーバ側での HTTP レスポンスヘッダの出力処理に問題があり，HTTP ヘッダインジェクションの脆弱性が存在すると思います。具体的には，入力された検索文字列を適切に処理せずに Set-Cookie ヘッダの値にセットしているものと思われます。この脆弱性を突いた攻撃では，［ b ］攻撃と同様に，攻撃者が指定した任意のスクリプトをクライアント側で実行できます。

U 主任：仮に問題があるとした場合，Set-Cookie ヘッダの値をセットするサーバ側の処理において，どのような対策が考えられますか。

T 君　：幾つかの対策があります。例えば，HTTP レスポンスヘッダを適切に出力するために，Web アプリの実行環境やプログラム言語が用意している，ヘッダ出力用の関数や API を使用する方法が考えられます。それが使用できない場合は，［ c ］するといった処理を開発者が自身で実装する方法も考えられます。

・出力文字列に改行コードがあるとエラー画面を出力
・出力文字列の改行コード以降の文字列を削除

U 主任：その他に気になる点はありますか。

T 君　：はい。図 6 の一連の HTTP ヘッダのうち，例えば，行番号［ d ］と行番号［ e ］を見比べると，サーバ側のセッション管理に問題があり，セッションフィクセーションの脆弱性が存在する可能性があります。攻撃者が Cookie Monster Bug を突く攻撃や，前述した HTTP ヘッダインジェクション攻撃を組み合わせることによって，セッションフィクセーションを成功させる可能性があります。図 9 に攻撃手順の一例を示します。

セッションハイジャックの例

1. 攻撃者 J が，試験用サイトの画面 A にアクセスし，セッション ID を取得する。このときの，セッション ID を“01234”とする。
2. 攻撃者 J が，攻撃対象の利用者 K に HTML メールを送信する。この HTML メールには URL リンクがあり，攻撃用のクエリ文字列を含む画面 C の URL が記載されている。
3. 利用者 K が，試験用サイトの画面 A にアクセスし，セッション ID を取得する。このときの，セッション ID を“56789”とする。その後，HTML メールの URL リンクをクリックする。その際に送信される HTTP リクエストには，攻撃者 J が用意した攻撃用クエリ文字列が含まれており，検索文字列の値は次のとおりである。

   `%0d%0aSet%2dCookie%3a%20SESSIONID%3d` ［ f ］ `%3b%20Expires%3d（省略）domain%3dexample%2eco%2ejp%3b（省略）`
4. 利用者 K がログインする。
5. 攻撃者 J は，利用者 K になりすまし，本来はアクセス権限がない画面にアクセスできるようになる。

**図 9　攻撃手順の一例**

U 主任：セッションフィクセーションの脆弱性について，どのような対策が考えられますか。

T 君　：例えば，サーバ側の処理を変更する方法があります。検査シナリオの画面遷移でいえば，ログイン後の，画面 E から画面 C に遷移する際の，サーバ側の処理において，［ g ］といった対策を行うことによって，この脆弱性を確実に防ぐことができます。

新規セッションIDを発行しセッションを開始する

U 主任は，その後も T 君に対してディスカッションや報告書の審査などを行い，技能試験は終了した。

T 君は見事，技能試験に合格し，SPG の検査員として業務を始めた。

**設問 1**　本文中の下線①について，(1)，(2)に答えよ。

(1) secure 属性が設定されていないと，どの画面に遷移するときにセッション ID を盗聴されるリスクがあるか。遷移直後の画面を，画面 A～E の中から一つ選び，答えよ。　画面 B

(2) secure 属性が設定されていないと，セッション ID を盗聴されるリスクがある理由を，40 字以内で述べよ。

暗号化されないHTTP 通信において，セッションID が送信されるから

**設問 2**　HTTP ヘッダインジェクションの脆弱性について，(1)～(3)に答えよ。

(1) 図 7 中の［ a ］に入れる適切な文字列を URL エンコード済の形式で答えよ。　%0d%0a%0d%0a

(2) 本文中の［ b ］に入れる適切な字句を解答群の中から選び，記号で答えよ。

解答群

ア　SQL インジェクション　　イ　TCP SYN Flood

(ウ) クロスサイトスクリプティング　　エ　ディレクトリトラバーサル

(3) 本文中の［ c ］に入れる適切な処理を，30 字以内で具体的に述べよ。

**設問 3**　セッション管理の脆弱性について，(1)～(4)に答えよ。

(1) 本文中の［ d 17 ］，［ 19 e ］に入れる適切な行番号を答えよ。

(2) 図 9 中の［ f ］に入れる適切な字句を答えよ。 01234

(3) 図 9 中の手順 4 及び 5 について，利用者 K がログインした後，攻撃者 J が

利用者 K になりすますことができるのはなぜか。“セッション ID”という字句を含めて 40 字以内で述べよ。

利用者Kに攻撃者Jが取得したセッションIDでログインさせているから

(4) 本文中の [ g ] に入れる適切な対策を，30 字以内で述べよ。