سوال اول

این نوع حملات Buffer Overflow به این صورت عمل میکند که باعث میشود دیتا بیش از حد Storage بر روی آن نوشته شود، در نتیجه چون دیتا از حد مجاز بزرگتر است Overflow رخ داده و مکان های خارج از محدوده را هم Overwrite میکند. روش هایی برای محافظت از آن وجود دارد. برای مثال:

سخت افزاری: در این روش میتوان سه شیوه را اجرا کرد.

1. Address space randomization: رو این شیوه مموری آدرس را به طور مرتب حرکت میدهیم زیر ویروس نیاز به دانستن کد های اطراف خود دارد و با این کار ویروس نمیتواند درست کار کند.

2. Data execution prevention: در این روش پرچم هایی داریم که اگر بخشی از مموری رفتار غیر معمولی نشان داد جلوی برنامه در حال اجرا را بگیرد.

3. یک ماژول امنیتی بین Cache و Memory اضافه کنیم به طوری که از Return Address در مقابل حملات محافظت بیشتری شود.

نرم افزاری: در این روش باید حین توسعه نرم افزار ماژول های امنیتی بیشتری بنویسیم و امنیت آن را طوری ارتقاء دهیم که موقعیت های Buffer Attack های احتمالی را تشخیص داده مطمئن شوند متغیر هایی که از Stack یا Heap استفاده میکنند رفتار غیر معمولی نمیکنند و اصطلاحا vulnerabilities را کم کنیم.

سوال دوم

در صورت Hit شدن به 210 ثانیه نیاز داریم (100 ثانیه برای دسترسی به TLB و 10 ثانیه رفتن به Frame و همچنین 100 ثانیه هم برای اجرا). در غیر این صورت به 110 ثانیه نیاز داریم.

EAT = (0.05 x 210) + (0.95 x 110) = 115 s

سوال سوم

مزایا:

1. یک لایه امنیتی برای قبل اجرای برنامه ها است و باعث میشود vulnerability ها و آسیب های احتمالی به سیستم گرفته شود.
2. Cost Sensitive است یا به عبارتی اگر برنامه ای از زمان یا منابع بیش از حد مجاز استفاده کرد/خواست استفاده کند آن را متوقف میکند.
3. به طور اتوماتیک کارش را انجام میدهد و نیازی به نظارت ندارد.

معایب:

1. اینکه به اجرا شدن برنامه ها کمی Cost اضافه میکند که مربوط به اجرا شدن خود برنامه Watchdog Program است. به عبارتی اجرا شدن این برنامه یک هزینه/بار اضافی بر دوش سیستم عامل است.

2. چون این برنامه تنها لایه امنیتی برای اجرا شدن برنامه ها است در صورت درست کار نکردن آسیب زیادی به سیستم و برنامه ها میزند مخصوصا وقتی Decision Making درستی نکند. به عبارتی در این مقطع یک Buttle Neck برای سیستم محسوب میشود.

سوال چهارم

Algorithm	Memory	Data
First-Fit		
	10KB	9 KB
	2KB	2KB
	8KB	8 KB
Best-Fit		
	30KB	20KB
	10KB	9KB
	8KB	8KB
Worst-Fit		
	10KB	9KB
	10KB	8KB
	30KB	25KB

سوال پنجم

الف) خیر، افزایش سرعت پردازنده نه تنها کمک نمیکند بلکه میتواند باعث کاهش بهرهوری شود. چون سیستم بر parsing disk محدود است که سرعت قابل توجه پایین تری دارد.

ب) خیر، مشکل از حجم parsing disk نیست، بلکه زمان است. با افزایش حجم این دستگاه هیچ دردی دوا نمیشود (:

ج) خیر، افزایش multiprogramming باعث کاهش locality شده و افزایش page fault. پس parsing disk بیشتر اشغال میشود و بهرهوری کاهش میابد.

د) بله، دقیقا برعکس حرف بالا اینجا صادق است و افزایش بهرهوری را به همراه دارد.

ه) اگر منظور رم باشد بله تاثیر زیادی دارد جون همجواری مکانی افزایش میابد.

و) بله، سرعت بیشتر انتقال اطلاعات رم باعث افزایش بهرهوری میشود طبیعتا.

ز) خیر، افزایش آن باعث کمتر شدن Page Frame ها شده و ارتباط با physical memory سخت تر میشود. پس بهرهوری کاهش میابد.