TEXT S OTÁZKAMI

Kontejnerizace je metoda virtualizace na úrovni operačního systému, která umožňuje spouštět aplikace a jejich závislosti v izolovaných prostředích nazývaných kontejnery. Na rozdíl od virtuálních strojů (VM), které virtualizují celý hardwarový stack včetně vlastního jádra (kernelu) operačního systému, kontejnery sdílejí jádro hostitelského operačního systému. Díky tomu jsou lehké, rychlé při startu a mají minimální režii.

Klíčovou technologií, která toto umožňuje, je Docker Engine — klientsko-serverová aplikace se třemi hlavními částmi:

1. Server (daemon dockerd): Dlouhodobě běžící proces, který naslouchá API požadavkům od klienta a spravuje Docker objekty – obrazy (images), kontejnery (containers), sítě a svazky (volumes).
2. REST API: rozhraní, kterým klient komunikuje s démonem.
3. Klient (příkaz docker): Příkazový řádek (CLI), který uživatel používá k interakci s Dockerem. Klient překládá příkazy (např. docker run) na volání REST API a posílá je démonovi ke zpracování.

Základní stavební jednotkou je obraz (image). Jedná se o neměnnou (immutable) šablonu pouze pro čtení, která obsahuje instrukce pro vytvoření kontejneru. Obraz se skládá z několika vrstev souborového systému (filesystem layers). Každý příkaz v souboru Dockerfile (např. RUN, COPY) vytváří novou vrstvu. Díky tomuto vrstvení jsou obrazy efektivní – pokud se dva obrazy liší pouze v poslední vrstvě, sdílejí všechny předchozí, což šetří místo na disku.

Když se z obrazu spustí instance, vznikne kontejner. Kontejner je v podstatě běžící proces izolovaný od ostatních procesů a hostitelského systému. Klíčové je, že Docker nad vrstvy obrazu (které jsou jen pro čtení) přidá jednu novou, zapisovatelnou vrstvu, tzv. kontejnerovou vrstvu. Veškeré změny provedené za běhu kontejneru (vytváření, mazání, modifikace souborů) se ukládají právě do této tenké vrstvy. Původní obraz zůstává netknutý. Pokud je kontejner smazán, je tato zapisovatelná vrstva ztracena. Pro trvalé uchování dat se používají svazky (volumes), jejichž životní cyklus není vázán na životnost kontejneru.

Otázky:

1. Uživatel stáhne obraz ubuntu:latest (velikost 70 MB) a následně stáhne obraz nginx:latest, který je založen na ubuntu:latest a přidává webový server (celková velikost 130 MB). Jaké místo tyto dva obrazy reálně zaberou na disku hostitelského systému podle principů popsaných v textu?

a) 200 MB, protože se sečte velikost obou obrazů.
b) 70 MB, protože obraz nginx je pouze lehký ukazatel na ubuntu.
c) Přibližně 130 MB, protože nginx obraz znovu využije všechny sdílené vrstvy z již staženého ubuntu obrazu.
d) Místo na disku se nezmění, protože obrazy jsou pouze šablony a nezabírají žádné místo, dokud z nich není spuštěn kontejner.

2.Pokud existují dva obrazy, které sdílejí většinu svých vrstev, co je podle textu hlavní výhodou tohoto stavu?

a) Zrychlí se start kontejnerů, protože nemusí načítat všechny vrstvy.
b) Zvýší se bezpečnost, protože sdílené vrstvy jsou kontrolovány na jednom místě.
c) Ušetří se místo na disku hostitelského systému.
d) Zjednoduší se komunikace mezi kontejnery vytvořenými z těchto obrazů.

3.Uživatel provede změnu v Dockerfile na řádku, který je zhruba uprostřed souboru. Co se stane při následném sestavování (build) nového obrazu?

a) Docker kompletně přestaví všechny vrstvy od začátku až do konce.
b) Docker znovu použije všechny existující vrstvy z mezipaměti (cache), protože změna uprostřed souboru nemá vliv.
c) Docker znovu použije vrstvy z mezipaměti až do místa změny a od změněného řádku dál všechny následující vrstvy přestaví.
d) Docker ignoruje změnu a použije původní obraz, dokud není mezipaměť manuálně vymazána.

4.Jakou roli hraje podle textu příkazový řádek (klient docker) v architektuře Docker Engine?

a) Přímo spravuje a spouští kontejnery, obrazy a svazky na hostitelském systému.
b) Slouží jako rozhraní pro uživatele a jeho příkazy překládá na API volání, která posílá démonovi.
c) Definuje specifikaci, jak spolu mají jednotlivé komponenty Dockeru komunikovat.
d) Je to dlouhodobě běžící proces, který naslouchá požadavkům a vykonává je.

5.Uživatel provede změnu v Dockerfile. Který koncept popsaný v textu vysvětluje, proč by následný build mohl být rychlejší než ten úplně první?

a) Existence zapisovatelné kontejnerové vrstvy.
b) Sdílení jádra hostitelského operačního systému
c) Schopnost obrazů skládat se z vrstev, které mohou být sdíleny.
d) Izolace běžících procesů kontejneru.

LOGICKÉ

Představte si, že máte za úkol seřadit datovou strukturu – v tomto případě pole (seznam) čísel – od nejmenšího po největší. Zvolíte k tomu algoritmus **bubble sort**.

Tento algoritmus pracuje na principu opakovaných průchodů polem. Během jednoho průchodu porovnává každý prvek s jeho bezprostředním sousedem (prvek na indexu **i** s prvkem na indexu **i+1**) a pokud nejsou ve správném pořadí (tj. levý je větší než pravý), prohodí je. Jeden kompletní průchod tedy znamená sérii porovnání a výměn od začátku pole až na jeho konec. Efektem každého úplného průchodu je, že největší prvek z dosud nesetříděné části pole se "probublá" na svou finální pozici.

Tvůj seznam čísel je:
[8,5,9,3,6,2,7,1,4]

6.Jak bude vypadat tento seznam po dokončení přesně dvou úplných průchodů algoritmem bubble sort?

a) [5, 8, 3, 6, 2, 7, 1, 4, 9]
b) [5, 3, 6, 2, 7, 1, 4, 8, 9]
c) [5, 3, 8, 6, 2, 7, 1, 4, 9]
d) [5, 3, 6, 2, 8, 1, 4, 7, 9]

Jsi bezpečnostní analytik a tvým úkolem je prověřit logy z firewallu a systému detekce anomálií (ADS). Máte podezření na pokročilý útok, který se skládá ze dvou fází:

1. Fáze 1: Slow Scan: Útočník se snaží zůstat nenápadný tím, že z jedné IP adresy zkouší různé porty s delšími časovými rozestupy. Většina těchto pokusů je zamítnuta (DENY).
2. Fáze 2: Brute-force: Jakmile útočník objeví otevřený, ale nestandardní port, zahájí na tento jediný port masivní vlnu pokusů o přihlášení. To se v logu projeví jako série rychlých, po sobě jdoucích záznamů ACCESS_DENIED ze stejné IP adresy na stejný port.

Tvým úkolem je identifikovat tento dvoufázový útok v logu a přesně určit, jak dlouho trvala druhá fáze – samotný brute-force útok.

Důležité:

- Brute-force fáze začíná prvním záznamem ACCESS_DENIED a končí posledním záznamem ACCESS_DENIED z dané série.
- Běžný provoz (např. na portech 80 a 443) a jednoduché rychlé skeny od jiných IP adres ignoruj. Hledáš kombinaci pomalého a následného brute-force útoku.

```
[11:21:15] FW:DENY      src_ip=198.51.100.12  dst_port=22
[11:21:16] FW:DENY      src_ip=198.51.100.12  dst_port=23
[11:21:17] FW:DENY      src_ip=198.51.100.12  dst_port=25
[11:22:48] FW:ALLOW     src_ip=203.0.113.55   dst_port=443
[11:24:05] FW:DENY      src_ip=159.89.33.101  dst_port=5900
[11:25:10] FW:ALLOW     src_ip=203.0.113.55   dst_port=80
[11:27:31] FW:DENY      src_ip=159.89.33.101  dst_port=1433
[11:29:55] FW:DENY      src_ip=159.89.33.101  dst_port=8000
[11:30:12] FW:ALLOW     src_ip=159.89.33.101  dst_port=8443
[11:32:04] ADS:ACCESS_DENIED src_ip=159.89.33.101 on_port=8443
[11:32:09] ADS:ACCESS_DENIED src_ip=159.89.33.101 on_port=8443
[11:32:15] ADS:ACCESS_DENIED src_ip=159.89.33.101 on_port=8443
[11:32:18] FW:DENY      src_ip=198.51.100.12  dst_port=8080
[11:32:23] ADS:ACCESS_DENIED src_ip=159.89.33.101 on_port=8443
[11:32:30] ADS:ACCESS_DENIED src_ip=159.89.33.101 on_port=8443
[11:33:01] ADS:ACCESS_DENIED src_ip=159.89.33.101 on_port=8443
[11:33:19] ADS:ACCESS_DENIED src_ip=159.89.33.101 on_port=8443
```

7.Jak dlouho trvala fáze 2 (brute-force útok), kterou provedl útočník?

a) 1 minuta a 15 sekund
b) 9 minut a 14 sekund
c) 1 minuta a 2 sekund
d) 57 sekund

Jako analytik pro dešifrování jste narazil/a na zdokonalenou metodu šifrování, kterou používá nepřátelský agent. Je to **Autokey Cipher** šifra, která eliminuje opakující se vzory v klíči tím, že jako jeho součást použije samotnou zprávu. Díky tomu je mnohem odolnější proti frekvenční analýze.

1. Přiřazení čísel k písmenům: Každému písmenu abecedy přiřadíme číslo podle jeho pořadí (A=0, B=1, ..., Z=25).
2. Generování klíče: Šifrování začíná krátkým klíčovým slovem. Jakmile se písmena klíčového slova vyčerpají, klíč pokračuje písmeny původního otevřeného textu. Klíč je tedy tvořen klíčovým slovem, za kterým následuje začátek samotné zprávy.
    - Příklad:
        - Zpráva k zašifrování: ATTACK AT DAWN (text bez mezer je ATTACKATDAWN)
        - Klíčové slovo: QUEEN
        - Výsledný klíč bude: QUEEN + ATTACKATDA (začátek zprávy) = QUEENATTACKATDA
1. Šifrování: Každé písmeno zprávy se zašifruje sečtením jeho číselné hodnoty s číselnou hodnotou odpovídajícího písmene z vygenerovaného klíče. Pokud součet přesáhne 25 (hodnota Z), odečte se od něj 26.
    - Příklad šifrování zprávy ATTACK s klíčem QUEENATT:

1. A (0) + Q (16) = 16 -> Q
2. T (19) + U (20) = 39. Po přetečení: 39 - 26 = 13 -> N
3. T (19) + E (4) = 23 -> X
4. A (0) + E (4) = 4 -> E
5. C (2) + N (13) = 15 -> P
6. K (10) + A (0) = 10 -> K

Zašifrovaný text ATTACK je QNXEPK.

8.Zašifruj zprávu "*SECRET MESSAGE*" (bez mezer) pomocí klíčového slova "*CIPHER*".

a) UMVHZC EWKQEIO
b) UMVYIKEIUJEZQ
c) UVWHZCEWKQEIO
d) VMWHZC ICGSOIO

Síťový inženýr konfiguruje tři nové bezpečnostní politiky na centrálním firewallu: Politika ALFA (blokuje neznámý provoz), Politika BETA (loguje veškerý provoz) a Politika GAMA (povoluje specifický provoz pro management). Politiky lze aplikovat nebo odebírat pouze po jedné. Inženýr si je vědom nebezpečných dočasných stavů:

- Pokud je aktivní Politika BETA (logování) bez Politiky ALFA (blokování), dojde k okamžitému přehlcení disku logy z internetu.
- Pokud je aktivní Politika ALFA (blokování) bez Politiky GAMA (povolení), inženýr ztratí přístup k firewallu a bude nutný tvrdý restart.

Inženýr začíná ve stavu, kdy žádná z těchto tří politik není aktivní, a chce se dostat do stavu, kdy jsou aktivní všechny tři.

9.Kolik nejméně kroků (aplikování nebo odebrání jedné politiky) musí provést?

a) 3 kroky
b) 5 kroků
c) 7 kroků
d) Není možné bezpečně dosáhnout cílového stavu.

Vývojář integruje službu, která komunikuje přes API s omezeným počtem povolených požadavků. Systém funguje na principu "kreditu", kde má vývojář na začátku 100 kreditů. Pravidla jsou následující:

- Jeden ZAPISUJÍCÍ požadavek (např. odeslání dat) stojí 15 kreditů.
- Jeden ČTECÍ požadavek (např. stažení stavu) stojí 1 kredit.
- Každých 5 sekund se automaticky obnoví 10 kreditů, ale nikdy ne více, než je maximální kapacita 100.

- Bezpečnostní limit: Během jakéhokoliv 10sekundového intervalu nelze provést více než 3 ZAPISUJÍCÍ požadavky.

10.Vývojář potřebuje co nejrychleji odeslat dávku dat. Kolik nejvíce ZAPISUJÍCÍCH požadavků může úspěšně provést během prvních 20 sekund, pokud začíná s plnými 100 kredity?

a) 4 požadavky
b) 6 požadavků
c) 7 požadavků
d) 8 požadavků

POROZUMĚNÍ TEXTU

Virtuální privátní síť (VPN) je služba, která vytváří chráněné a šifrované spojení mezi vaším zařízením a internetem. Když se k internetu připojíte prostřednictvím VPN, vaše data nejdou přímo na cílové webové stránky. Nejprve putují na server poskytovatele VPN, který funguje jako zprostředkovatel komunikace. Tento server následně odešle váš požadavek na cílovou stránku, ovšem pod svou vlastní veřejnou IP adresou – nikoli pod tou vaší.

Díky tomu se zdá, že veškerý provoz pochází z místa, kde se nachází VPN server, nikoli z vašeho skutečného zařízení nebo lokace. Celý přenos mezi vaším zařízením a serverem VPN je navíc šifrovaný, takže váš poskytovatel internetu sice vidí, že jste připojeni k VPN, ale nemůže zjistit, jaké weby navštěvujete ani jaký obsah si prohlížíte. Naopak samotné webové stránky vidí pouze veřejnou IP adresu VPN serveru a neznají vaši původní identitu ani polohu.

11.Jaký konkrétní účinek má podle textu používání VPN na to, jak se vaše online aktivita jeví webovým stránkám?

a) Webové stránky mohou stále zjistit vaši původní veřejnou IP adresu, ale díky šifrování nepoznají, odkud pochází přenášená data.
b) Webové stránky vidí vaši skutečnou veřejnou IP adresu, jenom se jim místo přesné polohy zobrazí umístění VPN serveru.
c) Webové stránky sice nevidí vaši veřejnou IP adresu, ale mohou vaši polohu odhadnout podle šifrovacího klíče použití VPN.
d) Webové stránky dostávají požadavky, které vypadají, že přicházejí ze serveru VPN, čímž je vaše skutečná veřejná IP adresa skryta.

API je soubor pravidel a nástrojů, který umožňuje různým softwarovým aplikacím spolu komunikovat. Představte si API jako číšníka v restauraci. Vy (aplikace A) si chcete objednat jídlo (data) z kuchyně (aplikace B). Nejdete přímo do kuchyně, ale zavoláte číšníka (API), sdělíte mu svou objednávku, on ji předá kuchyni a následně vám přinese hotové jídlo. Číšník zajišťuje, že komunikace proběhne standardizovaným způsobem, aniž byste museli znát složité procesy v kuchyni. Například když aplikace s předpovědí počasí zobrazuje data z meteorologické stanice, používá k tomu její API.

12.Které přirovnání nebo situace nejlépe vystihuje funkci API podle uvedeného textu?

a) API funguje jako tlumočník mezi dvěma lidmi, kteří mluví různými jazyky – překládá požadavky a odpovědi tak, aby si obě strany rozuměly, aniž by museli znát cizí jazyk.
b) API funguje jako šéfkuchař, který zpracovává suroviny podle vlastních pravidel a tvoří z nich nová data pro aplikaci.
c) API funguje jako recepční hotelu, který nejen přeposílá požadavky, ale zároveň rozhoduje, zda je host dost důležitý na to, aby byl obsloužen.
d) API funguje jako jídelní lístek, který jen ukazuje, jaká data existují, ale žádnou komunikaci mezi aplikacemi nezprostředkovává.

Phishing je typ kybernetického útoku, při kterém se útočník vydává za důvěryhodnou instituci (např. banku nebo sociální síť) a snaží se z oběti vylákat citlivé údaje, jako jsou hesla nebo čísla kreditních karet. Tyto útoky jsou obvykle plošné a rozesílají se tisícům uživatelů najednou s obecným oslovením. Oproti tomu Spear Phishing je cílená forma phishingu. Útočník si předem zjistí informace o konkrétní oběti nebo organizaci (jméno, pozice, kolegové) a vytvoří podvodnou zprávu na míru. Taková zpráva působí mnohem důvěryhodněji, protože obsahuje osobní detaily, a je tedy těžší ji odhalit. Čím více osobních informací zpráva obsahuje, tím účinnější bude.

V příkladu jsou k dispozici tyto údaje: Jan Novák, *vedoucí logistiky* ve firmě ALFA Tech s.r.o., právě pracuje na projektu Orion a často komunikuje s kolegyní Mgr. Petrou Dvořákovou.

13.Jaký předmět emailu by byl nejúčinnější pro Spear Phishing útok?

a) Důležité upozornění: Váš bankovní účet byl zablokován, klikněte pro odblokování.
b) RE: Projekt Orion - chybějící podklady od Petry Dvořákové.
c) Gratulujeme! Byl jste vylosován jako výherce nového iPhone 15.
d) Bezpečnostní varování IT oddělení: Změňte si prosím okamžitě heslo.

Software lze dělit podle licenčních podmínek na dvě hlavní kategorie. Proprietární (uzavřený) software je vlastnictvím konkrétní firmy, která drží jeho zdrojový kód v tajnosti. Uživatel si kupuje pouze licenci k jeho používání, ale nemůže ho studovat, upravovat ani dále šířit. Typickým příkladem je Microsoft Windows nebo Adobe Photoshop. Naopak Open-source (otevřený) software má veřejně dostupný zdrojový kód. Kdokoliv se může podívat, jak program funguje, může si ho upravit pro své potřeby, a dokonce se podílet na jeho vývoji. Příkladem je operační systém Linux.

14.Co je podle textu klíčovým rozdílem, který odlišuje open-source software od proprietárního?

a) Open-source software je vždy zdarma, zatímco proprietární je vždy placený.
b) Proprietární software je bezpečnější, protože jeho kód nikdo nevidí.
c) Dostupnost a možnost úpravy zdrojového kódu u open-source softwaru, což u proprietárního není možné.
d) Open-source software vyvíjí pouze komunita dobrovolníků, zatímco proprietární software vyvíjejí pouze firmy.

Blockchain je decentralizovaná a neměnná digitální „účetní kniha“. Skládá se z řetězce bloků, kde každý blok obsahuje sadu transakcí. Jakmile je blok zaplněn a přidán do řetězce, je kryptograficky propojen s blokem předchozím. Tím vzniká velmi bezpečné spojení – změna v jednom bloku by vyžadovala změnu všech následujících bloků, což je výpočetně téměř nemožné. Klíčovou vlastností je decentralizace: kopie této účetní knihy je uložena na mnoha počítačích po celém světě. Díky tomu neexistuje žádný centrální bod (jako banka), který by mohl selhat nebo data zmanipulovat.

15.Co podle textu zajišťuje vysokou odolnost blockchainu proti podvodným změnám v záznamech?

a) Jeden superpočítač, který hlídá správnost všech transakcí a okamžitě opravuje chyby.
b) Kombinace kryptografického propojení bloků a uložení kopií databáze na mnoha nezávislých počítačích.
c)Skutečnost, že všechny transakce jsou anonymní, a proto je nelze zpětně dohledat a změnit.
d) Pravidelné mazání starých bloků, aby se řetězec nestal příliš dlouhým a zranitelným.