AZƏRBAYCAN RESPUBLİKASI ELM VƏ TƏHSİL NAZİRLİYİ

AZƏRBAYCAN TEXNİKİ UNİVERSİTETİ

"Kibertəhlükəsizlik" kafedrası

Əlyazması hüququnda

Əlibəyli Orxan Qənbər oğlu

İxtisas: 060632 - "İnformasiya texnologiyaları və sistemləri mühəndisliyi"
İxtisaslaşma: "İnformasiya mühafizəsi və təhlükəsizliyi"
Mövzu: Veb sorğuların modifikasiyası üçün proksi serverin yazılması

MAGİSTRLİK DİSSERTASİYASI

Elmi rəhbər:  **t.f.d. Hüseynov Emin Zakir oğlu**

Kibertəhlükəsizlik kafedrasının
müdiri **t.f.d., dos. Yadigar İmamverdiyev Nəsib oğlu**

Bakı-2023

MÜNDƏRİCAT

İXTİSARLARIN SİYAHISI

API	Application Programming Interface – Tətbiqi Proqramlaşdırma İnterfeysi
CA	Certificate Authority – Sertifikat mərkəzi
CPU	Central Processing Unit – Mərkəzi prosessor
CSRF	Cross Site Request Forgery – Saytlar arası sorğu saxtakarlığı
DLP	Data Loss Prevention – Data sızıntısından qorunma
DNS	Domain Name System – Domain ad sistemi
FTP	File Transfer Protocol – Fayl transferi protokolu
OWASP	The Open World Wide Application Security Project - Açıq Ümumdünya Tətbiq Təhlükəsizliyi Layihəsi
SIEM	Security İncident Event Management – Təhlükəsizlik insidentlərinin və hadisələrinin idarəsi
SSL	Secure Socket Layer – Təhlükəsiz soket təbəqəsi
TLS	Transport Layer Security – Nəqliyyat qatının təhlükəsizliyi
TCP	Transmission Control Protocol – Transmissiya nəzarət protokolu
UDP	User Datagram Protocol – İstifadəçi dataqram protokolu
UI	User Interface – İstifadəçi interfeysi
URI	Uniform Resource İdentifier – Vahid resurs identifikatoru
URL	Uniform Resource Locator – Vahid resurs göstəricisi
UX	User Experience – İstifadəçi təcrübəsi
XML	Extensible Markup Language – Genişləndirilə bilən işarələmə dili
XSS	Cross Site Scripting – Saytlar Arası skriptləşdirmə
WWW	World Wide Web – Dünya miqyaslı şəbəkə

GİRİŞ

Mövzunun aktuallığı. İnternetin qarşılıqlı əlaqə və biznes tərzimizdə inqilab etməsi ilə veb texnologiyalar da müasir dünyamızda olduqca vacib hala gəlmişdir. Hal hazırda internet məlumat almaq, digərləri ilə ünsiyyət qurmaq istəyən hər hansı bir şəxsin ilk müraciət etdiyi vasitədir. İnternetin bu günkü vəziyyətinə gələ bilməsində isə veb texnologiyaların yaranmasının və inkişafının olduqca böyük rolu olmuşdur.

Hazırki informasiya sistemləri mühitində veb texnologiyalar dedikdə kompüterlərin bir-birləri ilə spesifik işarələmə dillərindən istifadə edərək əlaqə qurma mexanizmi nəzərdə tutulur. Bu texnologiyaların tətbiq sahələrinə misal olaraq ən kiçik biznes proqramlarından turmuş çox böyük həcmdə istifadəçiləri olan sosial şəbəkələrə qədər olan çox geniş sahələri misal göstərə bilərik.

Veb sistemlər bu qədər geniş yayıldığından və hazırki həyatımızda olan önəmi bu qədər hiss olunduğundan, bu sistemlərə edilə biləcək hər hansı bir mənfi müdaxilə də müasir cəmiyyətimizə, hər hansı bir dövlətə və ya quruma çox ciddi şəkildə təsir göstərəcəkir. Bu səbəbdən də təhlükəsizlik nöqteyi nəzərdən yanaşdıqda veb texnologiyaların iş prinsipinin başa düşülməsi, onların davamlı olaraq test edilməsi olduqca böyük əhəmiyyətə malikdir.

Veb sistemlərin təhlükəsizliyindən əmin ola bilmək üçün onların "nüfuzetmə testlərinə" məruz qoyulması hal hazırda olduqca geniş yayılmış praktikadır. Bu testləri etmək üçün isə veb sorğuların göndərilməsi və modifikasıyasını həyata keçirmək üçün xüsusi texnologiyalara ehtiyyac vardır. Sırf bu məqam nəzərə alındıqda veb sorğuların modifikasiyasını effektiv şəkildə həyata keçirə bilmək üçün açıq qaynaqlı olan proqram təminatının hazırlanması böyük önəmə malik olacaqdır. Hal hazırda bütün internetin veb sorğular əsanında qurulduğunu və onları test etmək üçün sorğuların mütləq şəkildə modifikasiya edilməli olduğunu nəzərə alsaq bu mövzu hal hazırki internet ekosistemində olduqca böyük önəmə malikdir.

Tətqiqatın obyekti: Tətqiqatın obyekti olaraq veb sorğuların həyata keçirilməsində istifadə edilən ayrı-ayrı sorğu və şifrələmə protokolları və onları

effektiv şəkildə modifikasiya edərək serverə göndərə bilmək üçün Python proqramlaşdırma dilinin bəzi kitabxanaları götürülmüşdür.

İşin məqsədi: Veb sistemlərin təhlükəsizliyini təmin edə bilmək üçün onların ayrı-ayrı situasiyalarda testlərə məruz edilməsi lazımdır. Bu testləri həyata keçirmək üçün göndərilən sorğunun parametrləri dəyişdirilməli, proqramçı tərəfindən nəzərdə tutulmayan şəkildə əlavə məlumatlar daxil edilməli və bunun kimi digər proseslər həyata keçirilməlidir. Bu proseslərin həyata keçirilməsində isə sorğuların modifikasiyası olduqca önəmlidir və bu tətqiatın əsas mahiyyəti də bu prosesi effektiv şəkildə edə biləcək açıq qaynaqlı proqram təminatının hazırlanmasıdır.

Tədqiqatın məsələləri: Tətqiqatın ilkin məsələsi veb sorğuları serverə çatmadan modifikasiya edə bilmək üçün lazım olan proksi serverin elə hazırlanmasıdır ki, göndərilən sorğular effektiv şəkildə qəbul edilə, dəyişdirilə və göndərilə bilsin.

İkinci məsələ isə gələn şifrələnmiş məlumatların emal edilə bilməsi üçün deşifrələnməsini həyata keçirə bilməkdən ibarətdir.

Alınmış nəticələrin praktiki əhəmiyyəti: Əldə edilmiş proqram təminatı vasitəsilə veb sistemlərin nüfuzetmə testləri onlara sərf edilən vaxt və nəticə etibarilə olduqca effektiv olacaqdır. Proqram təminatının açıq kodlu olduğunu nəzərə alsaq, bu proqramdan istənilən qurum və ya kibertəhlükəsizlik mühəndisləri öz sistemlərini test edə bilmək üçün olduqca rahat şəkildə istifadə edə biləcəkər.

I FƏSİL. VEB PROQRAMLARIN ÜMUMİ STRUKTURU

1.1 Müştəri – server arxitekturası

Müştəri-server arxitekturası müştərilər (sorğu edən qurğular) və serverlər (resurslar və ya xidmət təqdim edənlər) arasında tapşırıq və işləri bölüşdürən paylanmış hesablama modelidir. Müasir veb inkişafı kontekstində bu model əlavə komponentləri və texnologiyaları özündə birləşdirərək daha mürəkkəb və təkmilləşmiş bir hala gəlmişdir. İlk öncə bu strukturun müxtəlif aspektlərinə dərindən nəzər salaq.

Müştəri. Müştəri, adətən masaüstü kompüterlər, noutbuklar, smartfonlar və ya planşetlər kimi müxtəlif cihazlarda veb-brauzer ilə işləyən son istifadəçi komponentdir. Müştərilər istifadəçi interfeysini (UI) təqdim edir, istifadəçilərin qarşılıqlı əlaqəsini emal edir və məlumat tələb etmək, qəbul etmək üçün serverlərlə əlaqə saxlayırlar. Arxitekturanın müştəri hissəsinə aid edə biləcəyimiz komponentlər aşağıda sadalanmışdır.

- İstifadəçi interfeysi (UI) və İstifadəçi Təcrübəsi (UX) bu komponentlərdəndir. UI istifadəçilərin qarşılıqlı əlaqədə olduğu qrafik elementlərdən ibarətdir, UX isə ümumi istifadəyə, əlçatanlığa və istifadəçilərin interfeysi necə qəbul etdiyinə diqqət yetirir. Müasir veb tətbiqləri UI-ni müxtəlif ekran ölçülərinə və cihaz növlərinə uyğunlaşdıraraq, həssas dizaynı komponentlərini önə çıxarır.
- Front-end texnologiyalarında HTML (Hypertext Markup Language) məzmunun strukturlaşdırılması üçün, CSS (Cascading Style Sheets) üslub və tərtibat üçün və JavaScript isə interaktivlik və dinamik məzmun manipulyasiyası üçün istifadə olunur.
- Bunlardan başqa müştəri tərəfdə məlumatların saxlanması üçün bir növ bazaya da ehtiyyac vardır. Veb brauzerlər çərəzlər, localStorage, sessionStorage və IndexedDB kimi müxtəlif saxlama seçimlərini dəstəkləyir. Həmin bu texnologiyalradan istifadə etməklə brauzerlər proqramlara keşləmə, fərdiləşdirmə və oflayn giriş kimi məqsədlər üçün müştəri tərəfində məlumatları saxlamağa imkan verir.

- Həmçinin arxitekturanın ayrı-ayrı komponentlərinin əlaqələndirilə bilməsi üçün müəyyən bir texnologiyaya ehtiyyac vardır. Bu ehtiyyac Veb APİ-ların hesabına ödənir. Veb brauzerlər cihazın proqram təminatına, sensorlarına, geolokasiya, kamera, mikrofon və batareya vəziyyəti kimi sistem xüsusiyyətlərinə girişi təmin edən müxtəlif API-lərdən (Application Programming İnterface/Tətbiq Proqramlaşdırma İnterfeysləri) istifadə edir.

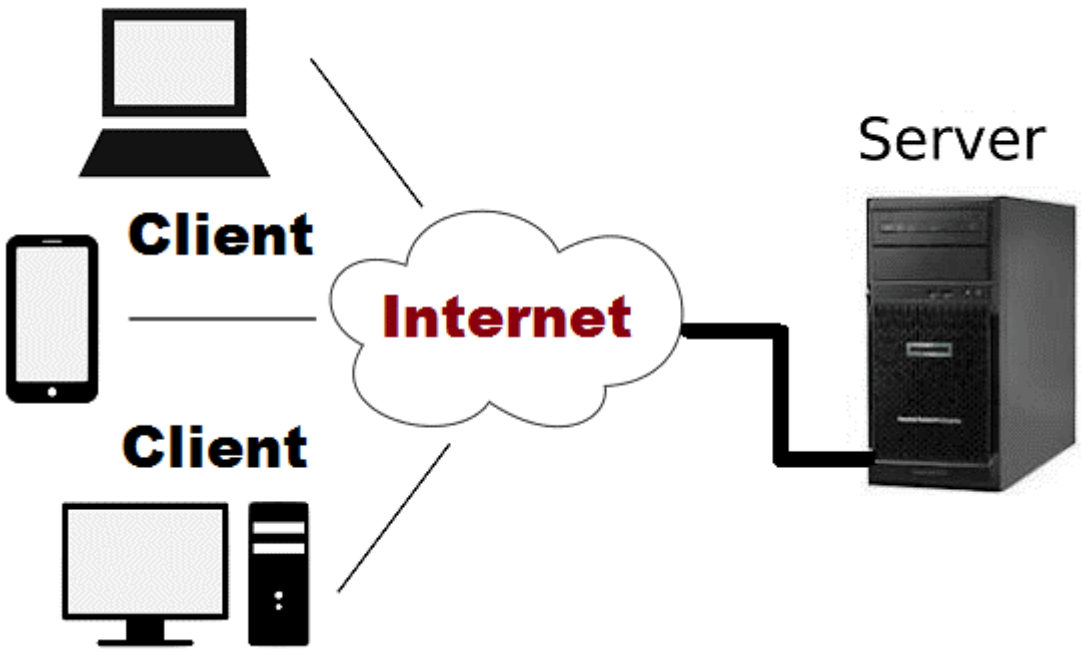
Server. Server müştəri sorğularını emal edir, biznes məntiqini idarə edir, məlumat və ya resursları müştəriyə qaytarır. Serverlər fiziki maşınlarda, virtuallaşdırılmış mühitlərdə və ya bulud infrastrukturunda yerləşdirilə bilər.

Arxitekturanın müştəri tərəfində olduğu kimi, server tərəfi də müəyyən komponentlərdən təşkil olunmuşdur və bu komponentlərin izahı aşağıdakı kimidir.

- Server tərəfindən istifadə olunan proqramlaşdırma dilləri: Proqramçılar əsasən müştəri sorğularını idarə edən, verilənlər bazası ilə qarşılıqlı əlaqədə olan və biznes qaydalarını tətbiq edən server tərəfi proqramlar yaratmaq üçün Python, Java, Ruby, PHP, C# və Node.js (JavaScript) kimi dillərdən istifadə edirlər.
- Server tərəfi proqramlaşdırma mühitləri və kitabxanalar: Django (Python), Express (Node.js), Ruby on Rails (Ruby), Laravel (PHP) və ASP.NET Core (C#) kimi alətlər server tərəfini sadələşdirmək üçün abstraksiyalar və konvensiyalar təmin edirlər ki, bunlar da hər hansı bir server üçün proqramın yazılma prosesini olduqca sürətləndirir. Flask (Python) və Sinatra (Ruby) kimi kitabxanalar yüngül alternativlər təklif edərkən onların qarşılığı olan daha da mürəkkəbləşdirilmiş mühitlərdən də istifadə etmək mümkündür.
- Keşləmə: Performansı yaxşılaşdırmaq üçün serverlər Memcached və ya Redis kimi yaddaşdaxili məlumat anbarlarından, Varnish və ya Nginx kimi əks proksilərdən istifadə edərək HTTP keşləmə əməliyyatlarını həyata keçirirlər.
- Ara proqram: Ara proqram sorğuları və cavabları emal edən, autentifikasiya, avtorizasiya, giriş və məlumatların yoxlanılması kimi tapşırıqları yerinə yetirən komponentlərdir. Ara proqram server tərəfi mühitlərdən və ya müstəqil alətlərdən istifadə etməklə yaradıla bilər.

- API-lər: API-lər müştərilər və serverlər arasında strukturlaşdırılmış əlaqəni təmin edir. Populyar API üslublarına JSON və XML kimi məlumat formatlarını istifadə edən REST (Representational State Transfer), GraphQL və gRPC daxildir.
- Mikroservislər: Bu memarlıq üslubu server tərəfindəki proqramları elə müstəqil hissələrə bölür ki, bu hissələrdən istifadə etməklə dözümlülüyü, proqramın sürətini, yüklənməsini və bunun kimi digər amilləri rahatlıqla artırıb azalda bilərik.

Müştəri server arxitekturasının ümumi sxemi şəkil 1.1-də göstərilmişdir.



Şəkil 1.1. Müştəri-server arxitekturası

1.2 Müştəri-server arxitekturasının alternativləri

Müştəri-server arxitekturasına alternativlər də mövcuddur. Müştəri-server müasir vebin inkişafı zamanı ən çox istifadə olunan model olsa da, digər memarlıq nümunələri də mövcuddur və xüsusi istifadə halları üçün daha məqsədə uyğun ola bilər. Bəzi bunun kimi alternativ modellər aşağıda göstərilmişdir:

- Peer-to-Peer (P2P) arxitekturası mərkəzi serverə ehtiyacı aradan qaldıran, bunun əvəzinə şəbəkə daxilində iştirakçı qovşaqlar arasında birbaşa əlaqə və resurs mübadiləsini təmin edən mərkəzləşdirilməmiş hesablama modelidir. P2P sistemlərində hər bir qovşaq həm müştəri, həm də server kimi fəaliyyət göstərməklə emal gücü, saxlanma və bant genişliyi kimi resurslara töhfə verir. Bu, iş yüklərini paylaya bilən və dəyişən şərtlərə dinamik şəkildə uyğunlaşa bilən, yüksək dayanıqlı, genişlənən və özünü təşkil edən şəbəkə ilə nəticələnir. P2P sistemləri fayl paylaşma şəbəkələri (məsələn, BitTorrent), paylanmış hesablama layihələri (məsələn, SETI@home, Folding@home), blokçeyn sistemləri (məsələn, Bitcoin, Ethereum) və kommunikasiya platformaları (məsələn,) daxil olmaqla müxtəlif proqramlarda istifadə edilmişdir (Skype, WebRTC).

  P2P arxitekturasının əsas üstünlükləri onun qeyri-mərkəzləşdirilmiş təbiəti, xətalara dözümlülüyü və miqyaslılığıdır. Mərkəzsizləşdirmə tək uğursuzluq nöqtələrini aradan qaldırır, məxfiliyi artırır və mərkəzləşdirilmiş orqanlara və ya infrastruktura asılılığı azaldır. Xətaya dözümlülük, sistemin hətta ayrı-ayrı qovşaqlar sıradan çıxdıqda belə işlək qalmasını təmin edir, çünki qalan qovşaqlar əlaqə saxlamağa və resursları paylaşmağa davam edə bilər. Şəbəkəyə qoşulan hər bir yeni qovşaq yeni resurslar əlavə edərək, iş yükünü paylaşır və bununla da miqyaslılıq əldə edilir. P2P sistemləri tez-tez məlumatların səmərəli saxlanması və axtarışı üçün Paylanmış Hash Cədvəlləri (DHT) kimi üsullardan və marşrutlaşdırma və qovşaqların aşkarlanması üçün Chord və ya Kademlia kimi alqoritmlərdən istifadə edir. Bununla belə, P2P arxitekturaları təhlükəsizlik problemləri (məsələn, zərərli qovşaqlar, Sybil hücumları), potensial hüquqi problemlər (məsələn, fayl paylaşımında müəllif hüquqlarının pozulması) və resursların idarə edilməsi (məsələn, pulsuz istifadə, qeyri-bərabər resurs töhfəsi) daxil olmaqla digər növ problemlərlə də üzləşir. Bu problemləri həll etmək üçün P2P sistemləri qovşaqlar arasında əməkdaşlığa və resursların ədalətli bölüşdürülməsinə təşviq edən reputasiya sistemləri, kriptoqrafik üsullar, təşviq mexanizmlərini tətbiq etməyə çalışırlar. Çatışmazlıqlarına baxmayaraq, P2P

arxitekturası möhkəm, miqyaslana bilən və mərkəzləşdirilməmiş tətbiqlər yaratmaq üçün unikal üstünlüklər və imkanlar təklif edərək ənənəvi müştəri-server modellərinə cəlbedici alternativ olaraq qalır.

- Nəşr et-Abunə ol (Pub/Sub) Arxitekturası: Pub/Sub arxitekturasında iştirakçılar (naşirlər və abunəçilər) mesaj brokeri və ya hadisə avtobusu adlanan vasitəçidən istifadə etməklə dolayı olaraq əlaqə qururlar. Nəşriyyatçılar hadisələr və ya mesajlar yaradır, abunəçilər isə xüsusi hadisə növlərinə maraq göstərərək bu mesajlardan istifadə edirlər. Mesaj brokeri mesajları nəşriyyatçılardan onlar ilə maraqlanan abunəçilərə yönləndirir, ayırma və çeviklik təmin edirlər. Pub/Sub arxitekturaları hadisəyə əsaslanan sistemlərdə, mesaj yönümlü ara proqramda və real vaxt məlumat emalında istifadə olunur (məsələn, Apache Kafka, RabbitMQ, Google Cloud Pub/Sub).
- Üç səviyyəli memarlıq: Üç səviyyəli arxitektura müştəri-server modelinin genişləndirilməsi hesab olunsa da, o, server tərəfi komponentini iki fərqli təbəqəyə ayırır: proqram serveri və verilənlər bazası serveri. Bu ayrılıq hər bir səviyyənin müstəqil olaraq inkişaf etməsinə imkan verməklə modulluğu, miqyaslılığı və davamlılığı təşviq edir. Veb kontekstində müştəri səviyyəsi istifadəçi interfeysini idarə edir, proqram serveri biznes məntiqini emal edir, verilənlər bazası serveri isə məlumatların saxlanması və axtarışını idarə edir.

  Üç səviyyəli arxitektura tətbiqi üç fərqli təbəqəyə ayıraraq modulluğu, davamlılığı və miqyaslılığı artıran geniş şəkildə qəbul edilmiş proqram dizayn nümunəsidir. Bu dizayn pillələri bunlardır: təqdimat, tətbiq və məlumat səviyyələri. Təqdimat səviyyəsi dedikdə son istifadəçinin məlumatları hansı şəkildə gördüyü hansı sorğuları həyata keçirdiyi və.s kimi əməliyyatların həyata keçirildiyi səviyyə nəzərdə tutulur. Tətbiq səviyyəsi və ya biznes məntiqi səviyyəsi təqdimat və məlumat səviyyələri arasında vasitəçi kimi xidmət edir, istifadəçi sorğularını idarə edir, biznes qaydalarını tətbiq edir və məlumat axınını idarə edir. Digər səviyyələr arasında qüsursuz əlaqə yaratmaq üçün server tərəfi dillərindən, çərçivələrdən, ara proqramlardan və API-lərdən istifadə edir. Nəhayət, məlumat səviyyəsi məlumatların bütövlüyünü, ardıcıllığını və

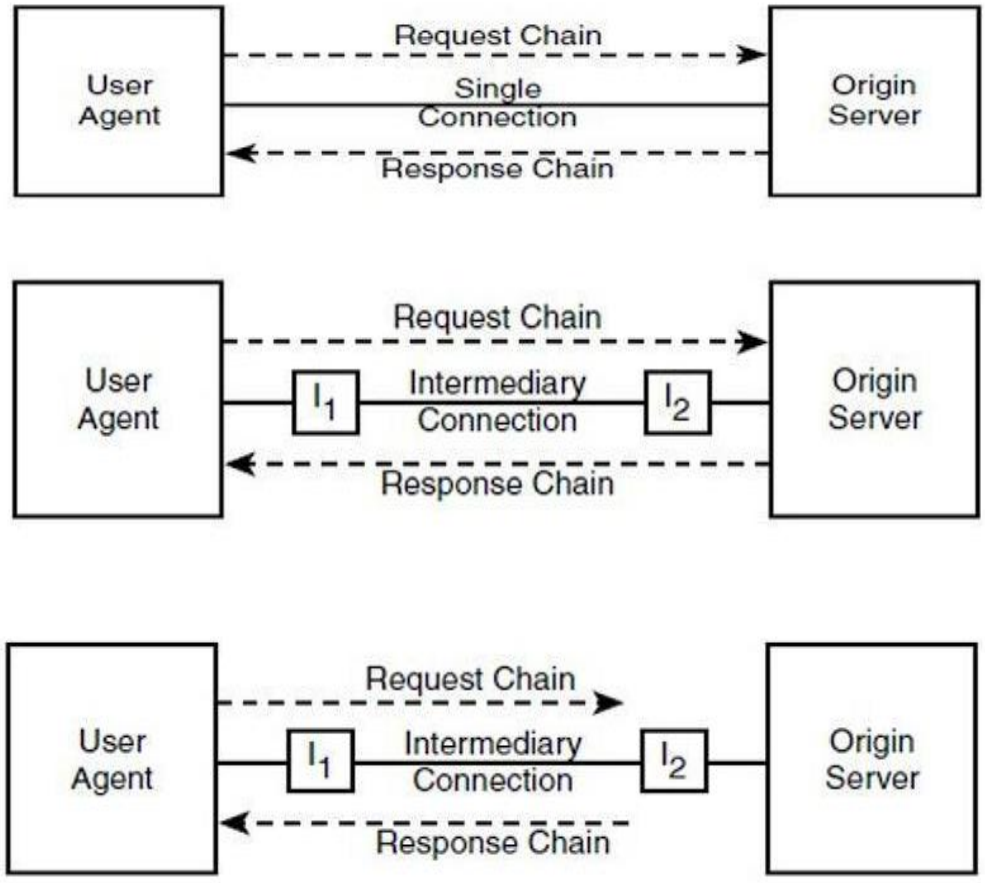
davamlılığını təmin etmək üçün müxtəlif əlaqəli və NoSQL verilənlər bazalarından, məlumat əldə etmək üçün isə kitabxanalardan və ORM alətlərindən istifadə edərək məlumatların saxlanmasına, idarə edilməsinə və axtarışına diqqət yetirir. Bu arxitekturada problemlərin ayrılması hər bir səviyyəyə müstəqil şəkildə təkamül etməyə, resursların səmərəli bölüşdürülməsinə imkan verir və dəyişən tələblərə, texnologiyalara uyğunlaşa bilən möhkəm, genişlənə bilən və saxlanıla bilən proqramların işlənib hazırlanmasını asanlaşdırır. Üç səviyyəli arxitekturadan istifadə etməklə tərtibatçılar müxtəlif istifadə hallarına və biznes ehtiyaclarına cavab verən çevik, yüksək performanslı proqramlar yarada, eyni zamanda uzunmüddətli uyğunlaşma və dayanıqlılığı təmin edə bilərlər.

- Xidmət yönümlü Arxitektura (SOA): Xidmət yönümlü arxitekturada tətbiqlər standart protokollar (məsələn, HTTP, SOAP və ya REST) üzərindən əlaqə saxlayan, sərbəst bağlanmış, müstəqil xidmətlərin inteqrasiyası ilə qurulur. Xidmətlər müxtəlif texnologiyalardan istifadə etməklə həyata keçirilə bilər və bir neçə tətbiqdə təkrar istifadə oluna bilər. SOA modulluğu, təkrar istifadəni və qarşılıqlı əlaqəni təşviq edir. Mikroservislər SOA-nı xüsusi tətbiqi olaraq sayıla bilsə də, SOA ümumiyyətlə əvvəlcədən mövcud olan, çox vaxt heterojen sistemlərin inteqrasiyasına diqqət yetirir.
- Serversiz Arxitektura: Serversiz arxitekturada tərtibatçılar əsas server infrastrukturunu idarə etmədən proqramlar qururlar. Əvəzində, bulud provayderi tələbatla avtomatik olaraq miqyaslanaraq, funksiyaları yerinə yetirmək üçün resursları dinamik şəkildə bölüşdürür. Bu yanaşma server idarəçiliyini mücərrədləşdirir və proqramçılara proqram kodunu yazmağa fokuslanmağa imkan verir. Serversiz arxitekturalarda tez-tez AWS Lambda, Azure Functions və ya Google Cloud Functions kimi Function-as-a-Service (FaaS) platformalarından istifadə edilir.

Hər bir memarlıq nümunəsinin güclü və zəif tərəfləri var və seçim tətbiqin tələbləri, mövcud resurslar və inkişaf məhdudiyyətləri kimi amillərdən asılıdır.

1.3 HTTP

Hypertext Transfer Protocol (HTTP) yarandığı gündən müasir internetdə istifadə olunan fundamental protokoldur. Bu, Ümumdünya Şəbəkəsində (WWW) məlumat mübadiləsi, xəbərləşmə, canlı kommunikasiya qurma, sənəd mübadiləsi, bank tranzaksiyaları və digər önəmli əməliyyatlar üçün tətbiq səviyyəli protokoldur. HTTP, müxtəlif proqramları, mürəkkəb arxitekturaları və kütləvi istifadəçi bazasını əhatə etmək üçün böyüyən müasir internetin dəyişən ehtiyaclarına və tələblərinə uyğunlaşaraq illər ərzində əhəmiyyətli dərəcədə inkişaf etmişdir. HTTP/1.1 və HTTP/2 geniş yayılmış protokolun iki əsas versiyasıdır və HTTP/3 növbəti nəsil olaraq hal hazırda inkişaf etdirilməkdədir.



Şəkil 1.2. HTTP protokolunun iş prinsipi

Geniş yayılmış ilk versiya olan HTTP/1.1, müştərilərə serverlərdən müvafiq cavab vermək üçün resurslar tələb etməyə imkan verən veb kommunikasiyası üçün sadə

sorğu-cavab modelini təqdim etmişdir. Veb proqramları daha da mürəkkəbləşdikcə, HTTP/1.1-in performansında məhdudiyyətlər yarandı. Məsələn, o, hər bir əlaqə üçün yalnız bir gözlənilməz sorğunu dəstəkləyir və bu, xəttin bloklanması kimi gecikmə problemlərinə səbəb olur. Bunu azaltmaq üçün veb proqramçılar tez-tez çoxlu bağlantılardan istifadə edirdilər. Bu isə nəticədə əlavə xərclər və resurs istehlakının artmasına gətirib çıxarırdı. Bu məhdudiyyətlərə cavab olaraq, əhəmiyyətli performans təkmilləşdirmələri təklif edən HTTP/2 protokolu 2015-ci ildə təqdim edildi. HTTP/2, çoxsaylı sorğuların və cavabların eyni vaxtda bir əlaqədə ötürülməsinə imkan verən multipleksləşdirmə funksiyasını yerinə yetirir və bununla da gecikmə və resurs istifadəsini azaldır. Bundan əlavə, o, veb rabitəsini daha da optimallaşdıraraq başlığın sıxılması və server tərəfindən ilkin sorğu göndərilməsi kimi xüsusiyyətlərini də dəstəkləyir.

Müasir internet inkişaf etməyə davam etdikcə HTTP də inkişaf edir. Protokolun ən son versiyası olan HTTP/3 hal-hazırda HTTP/2-də qalan bəzi performans problemlərini, xüsusən də yüksək gecikmə şəbəkələri və ya etibarsız bağlantılar üzərində gecikmə ilə bağlı problemləri həll etmək məqsədi ilə hazırlanır. HTTP/3 sələfləri tərəfindən qoyulmuş təməl üzərində qurulur, lakin əsas nəqliyyat protokolu olan TCP-ni daha müasir və səmərəli QUIC protokolu ilə əvəz edir. QUIC, TCP və İstifadəçi Datagram Protokolunun (UDP) ən yaxşı xüsusiyyətlərini özündə birləşdirir, daha sürətli əlaqə qurulması, təkmilləşdirilmiş sıxlığa nəzarət və səhvlərin düzəldilməsi təklif edir ki, bu da nəticədə daha həssas və etibarlı veb təcrübəsinin yaranması deməkdir. HTTP/3 daha geniş şəkildə qəbul olunduqca, onun müasir internetin imkanlarını və performansını daha da inkişaf etdirməsi, onlayn kommunikasiya və əməkdaşlığın gələcəyini formalaşdırması gözlənilir.

İndi isə HTTP protokolunun iş prinsipinə biraz daha yaxından nəzər salaq.

Hypertext Transfer Protocol (HTTP) HTML faylları, şəkillər və digər resurslar kimi hipermedia sənədlərini internet üzərindən ötürmək üçün istifadə olunan əsas proqram səviyyəli protokoldur. O, müştərilər (məsələn, veb-brauzerlər) və serverlər arasında əlaqəni asanlaşdıran Ümumdünya Şəbəkəsinin əsasını təşkil edir. Protokol sorğu-cavab modelində işləyir, burada müştərilər resurslar üçün sorğulara başlayır və serverlər

həmin sorğuları emal edir və müvafiq cavabları qaytarırlar. HTTP davamsız protokoldur, yəni hər bir sorğu və cavab cütü əvvəlki qarşılıqlı əlaqə haqqında məlumatı olmayan müstəqil əməliyyat kimi qəbul edilir. Bu xüsusiyyətin həm üstünlükləri, həm də çatışmazlıqları var və bu, çərəzlər və sessiya saxlama kimi davamlılığın idarə edilməsi üçün müxtəlif üsulların inkişafına gətirib çıxarır.

HTTP mesajları iki əsas komponentdən ibarətdir: başlanğıc xətti və bir sıra başlıq sahələri. Sorğu mesajı üçün başlanğıc xətti HTTP metodunu (məsələn, GET, POST, PUT, DELETE), sorğu hədəfini (URL kimi) və HTTP versiyasını ehtiva edir. Cavab mesajları üçün başlanğıc xəttinə HTTP versiyası, sorğunun nəticəsini göstərən status kodu və status kodunu təsvir edən səbəb ifadəsi daxildir. Başlanğıc xəttindən sonra məzmun növü, məzmun uzunluğu və ya keşləmə direktivləri kimi sorğu və ya cavab haqqında əlavə məlumat verən açar-dəyər cütləri gəlir. Başlıqlardan sonra, forma məlumatları, fayl yükləmələri və ya tələb olunan HTML sənədi kimi sorğunun və ya cavabın yükünü ehtiva edən əlavə mesaj orqanı daxil edilə bilər.

HTTP/1.0 və HTTP/1.1, protokolun ilk versiyaları, bağlantıların qurulması və məlumatların etibarlı, sifarişli çatdırılmasını təmin etmək üçün Transmissiya İdarəetmə Protokoluna (TCP) etibar edirdi. HTTP/1.0 hər bir sorğu-cavab cütü üçün yeni TCP bağlantısından istifadə etdi və bu, əhəmiyyətli əlavə yükə və gecikməyə səbəb oldu. Bu problemi həll etmək üçün HTTP/1.1 tək TCP bağlantısı üzərindən çoxsaylı sorğuların və cavabların ötürülməsinə imkan verən davamlı bağlantılar konsepsiyasını təqdim etdi. Bu, bağlantıların qurulması və bağlanması, performansın yaxşılaşdırılması ilə bağlı əlavə xərcləri azaldır. Bununla belə, HTTP/1.1 hələ də sıra blokundan əziyyət çəkir, bu məhdudiyyət, onları idarə etmək üçün resursların mövcudluğuna baxmayaraq, tək gözlənilməz sorğu digər sorğuların işlənməsini blokladığı zaman baş verir.

Daha öncə də deyildiyi kimi, HTTP/1.1 məhdudiyyətlərini aradan qaldırmaq üçün HTTP/2 2015-ci ildə təqdim edilib. O, ikili protokoldur və mətn əsaslı HTTP/1.x protokolları ilə müqayisədə onu daha səmərəli və daha az səhvə meyilli edir. HTTP/2 çoxlu sorğu və cavabların bir TCP bağlantısı üzərindən eyni vaxtda ötürülməsinə imkan verən multipleksləşdirmə funksiyasını yerinə yetirir və beləliklə, xəttin bloklanması problemini azaldır. Bundan əlavə, ötürülməli olan məlumatların miqdarını

azaltmaq üçün başlığın sıxılmasını və serverlərin müştərilərə resursları proaktiv şəkildə göndərməsinə imkan verən bir mexanizm olan server təkanını təqdim edir və veb səhifəni yükləmək üçün lazım olan sorğuların sayını potensial olaraq azaldır. Bu təkmilləşdirmələr daha sürətli və daha səmərəli veb təcrübəsinə kömək edir.

İnternet inkişaf etməyə davam etdikcə HTTP də müasir internetin ehtiyaclarını ödəmək üçün təkmilləşdirilir. Protokolun ən son versiyası olan HTTP/3 aktiv inkişaf mərhələsindədir və xüsusilə yüksək gecikmə və etibarsız bağlantılar üçün əhəmiyyətli performans təkmilləşdirmələri vəd edir. HTTP/3 HTTP/2-nin irəliləyişləri əsasında qurulur, lakin nəqliyyat təbəqəsi kimi TCP-ni Quick UDP İnternet Bağlantıları (QUIC) protokolu ilə əvəz edir. QUIC, TCP-nin etibarlılıq və sıxlığa nəzarət xüsusiyyətlərini UDP-nin aşağı gecikmə və əlaqəsiz təbiəti ilə birləşdirir, daha sürətli əlaqə qurulması, təkmilləşdirilmiş sıxlığa nəzarət və xətaların düzəldilməsi təklif edir. QUIC-dən istifadə etməklə HTTP/3 Ümumdünya Şəbəkəsinin performansını, cavabdehliyini və etibarlılığını daha da artırmaq, onlayn ünsiyyət və əməkdaşlığın gələcəyini formalaşdırmaq məqsədi daşıyır.

Hypertext Transfer Protocol (HTTP) internet üzərindən müştərilər (adətən veb-brauzerlər) və serverlər arasında əlaqə yaratmağa imkan verən müştəri-server protokoludur. HTTP-nin necə işlədiyini başa düşmək üçün onun müxtəlif komponentləri və prosesləri, o cümlədən sorğu-cavab dövrü, mesaj strukturu, metodlar, status kodları və başlıqları öyrənmək vacibdir.

- Sorğu-Cavab Dairəsi: HTTP-nin təməli sorğu-cavab dövrünə əsaslanır. Müştəri serverdən resurs (məsələn, veb səhifə və ya şəkil) üçün sorğu başladır. Server sorğunu emal edir, müvafiq cavab yaradır və onu yenidən müştəriyə göndərir. Müştəri daha sonra cavabı şərh edir və lazım gələrsə, istifadəçi üçün resursu təqdim edir (məsələn, brauzerdə veb səhifəni göstərmək).
- Mesaj Strukturu: Həm HTTP sorğuları, həm də cavablar başlanğıc xətti, başlıqlar və əlavə mesaj gövdəsindən ibarət oxşar strukturu paylaşır. Sorğudakı başlanğıc xətti HTTP metodunu (GET, POST və s.), hədəf URI-ni və HTTP versiyasını ehtiva edir. Cavabda başlanğıc xətti HTTP versiyasını, status kodunu və statusu təsvir edən səbəb ifadəsini ehtiva edir. Başlıqlar məzmun növü,

məzmun uzunluğu və keşləmə direktivləri kimi sorğu və ya cavab haqqında metadata ötürən açar-dəyər cütləridir. Mesaj gövdəsi, əgər varsa, forma məlumatları, fayl yükləmələri və ya tələb olunan resursun özü kimi faydalı yükü ehtiva edir.

- HTTP Metodları: Protokol müəyyən edilmiş resursda yerinə yetiriləcək istənilən hərəkəti göstərmək üçün müxtəlif üsullardan istifadə edir. Ən çox yayılmış üsullar GET (resursu əldə etmək), POST (emal edilmək üçün verilənləri təqdim etmək), PUT (mövcud resursu yeniləmək), DELETE (resursu silmək) və HEAD (faktiki məzmunu olmayan resurs haqqında metadata əldə etmək) üsullarıdır. Bu üsullar sorğunun semantikasını müəyyənləşdirməyə kömək edir və serverlərə onları müvafiq şəkildə emal etməyə imkan verir.
- Status kodları: HTTP cavab mesajlarına sorğunun nəticəsini göstərən üç rəqəmli rəqəmlər olan status kodları daxildir. Vəziyyət kodları ilk rəqəminə görə beş sinifdə qruplaşdırılır: 1xx (Məlumat), 2xx (Uğurlu), 3xx (Yönləndirmə), 4xx (Müştəri xətası) və 5xx (Server xətası). Bəzi ümumi status kodlarına misal olaraq 200 OK (sorğu uğurlu), 404 Tapılmadı (tələb olunan resurs tapılmadı) və 500 Daxili Server Xətası (sorğunu emal edərkən server xəta ilə qarşılaşdı) kimi status kodları misal göstərə bilərik.

  HTTP status kodları serverlərin müştəri sorğusunun nəticəsini bildirmək üçün istifadə etdiyi üç rəqəmli nömrələrdir. Onlar HTTP cavab mesajının ayrılmaz hissəsidir və müştərilərə sorğularının nəticəsi haqqında vacib məlumat verir. Vəziyyət kodları ilk rəqəmə əsasən beş sinifə qruplaşdırılıb və hər bir sinif fərqli nəticələr kateqoriyasını təmsil edir. Status kodlarını bu şəkildə təsnif etməklə, HTTP protokolu sorğunun nəticəsini çatdırmaq üçün standartlaşdırılmış üsul təklif edir və müştərilərə müvafiq reaksiya verməyə imkan verir.

  Birinci sinif, 1xx (Məlumat) müvəqqəti cavabı göstərir və sorğu prosesi zamanı ilk növbədə müştəri ilə server arasında ünsiyyət üçün istifadə olunur. Bu sinifdəki kodlar praktikada nadir hallarda rast gəlinir, 100 Davam və 101 Kommutasiya Protokolu ən diqqətəlayiq nümunələrdir. 100 Davam et status kodu müştəriyə serverin sorğu başlıAqlarını qəbul etdiyini və sorğunun əsas

hissəsini gözlədiyini bildirir, 101 Kommutasiya Protokolları kodu isə serverin adətən WebSocket-ə təkmilləşdirmə zamanı müştəri tərəfindən müəyyən edilmiş protokola dəyişdiyini bildirir.

Ən çox rast gəlinən status kodları 2xx (Uğurlu), 3xx (Yönləndirmə), 4xx (Client Error) və 5xx (Server Error) siniflərinə aiddir. 2xx sinfi sorğunun uğurla emal edildiyini göstərir, 200 OK ən ümumi koddur, uğurlu GET və ya POST sorğusunu bildirir. 3xx sinfi yönləndirmə üçün istifadə olunur və müştərilərə sorğunu tamamlamaq üçün əlavə tədbirlər görmələri lazım olduğunu bildirir. Bəzi tanış 3xx kodlarına 301 Daimi Köçürdü və 302 Tapıldı daxildir ki, bu da müştəriyə tələb olunan resursun müvafiq olaraq daimi və ya müvəqqəti olaraq yeni yerə köçürülməsi barədə məlumat verir. 4xx sinfi müştəri səhvlərini təmsil edir və müştərinin sorğuda səhv və ya natamam məlumat verdiyini göstərir. Ümumi 4xx kodlarına 400 Bad Sorğu, 401 İcazəsiz, 403 Qadağan edilmiş və 404 Tapılmadı daxildir. Nəhayət, 5xx sinfi server etibarlı sorğunu yerinə yetirmədikdə baş verən server xətalarını təmsil edir. Bəzi tipik 5xx kodları 500 Daxili Server xətası, 502 Bad Gateway və 504 Gateway Timeout-dur.

Yekun olaraq, HTTP status kodları müştəri sorğularının nəticəsinin çatdırılmasında mühüm rol oynayır, müştərilər və serverlər üçün əməliyyatın nəticəsini bildirmək üçün standartlaşdırılmış metod təmin edir. Uğurlu nəticələrə, yönləndirmələrə, müştəri xətalarına və server xətalarına qədər məlumat cavablarından tutmuş status kodları sorğunun vəziyyətini ardıcıl şəkildə təmsil etməyə imkan verir və tərtibatçılara müxtəlif ssenariləri idarə edə bilən möhkəm veb proqramları yaratmağa imkan verir. Veb inkişaf etməyə davam etdikcə, hamar ünsiyyət və optimal istifadəçi təcrübəsini təmin etmək üçün bu status kodlarını anlamaq və səmərəli istifadə etmək vacibdir.

- Başlıqlar: Başlıqlar sorğu və ya cavab haqqında əlavə məlumatın ötürülməsində mühüm rol oynayır. Bəzi ümumi sorğu başlıqlarına "İstifadəçi-Agent" (müştəri proqram təminatı haqqında məlumat verir), "Qəbul edirəm" (müştərinin idarə edə biləcəyi media növlərini müəyyənləşdirir) və "Avtorizasiya" (müştərinin autentifikasiyası üçün etimadnamələri ehtiva edir) daxildir. Ümumi cavab

başlıqlarına "Məzmun növü" (cavab orqanının media növünü göstərir), "Məzmun uzunluğu" (cavab orqanının ölçüsünü müəyyən edir) və "Keş-nəzarət" (müştərilərə və etibarnamələrə keşləmə direktivlərini təqdim edir) daxildir.

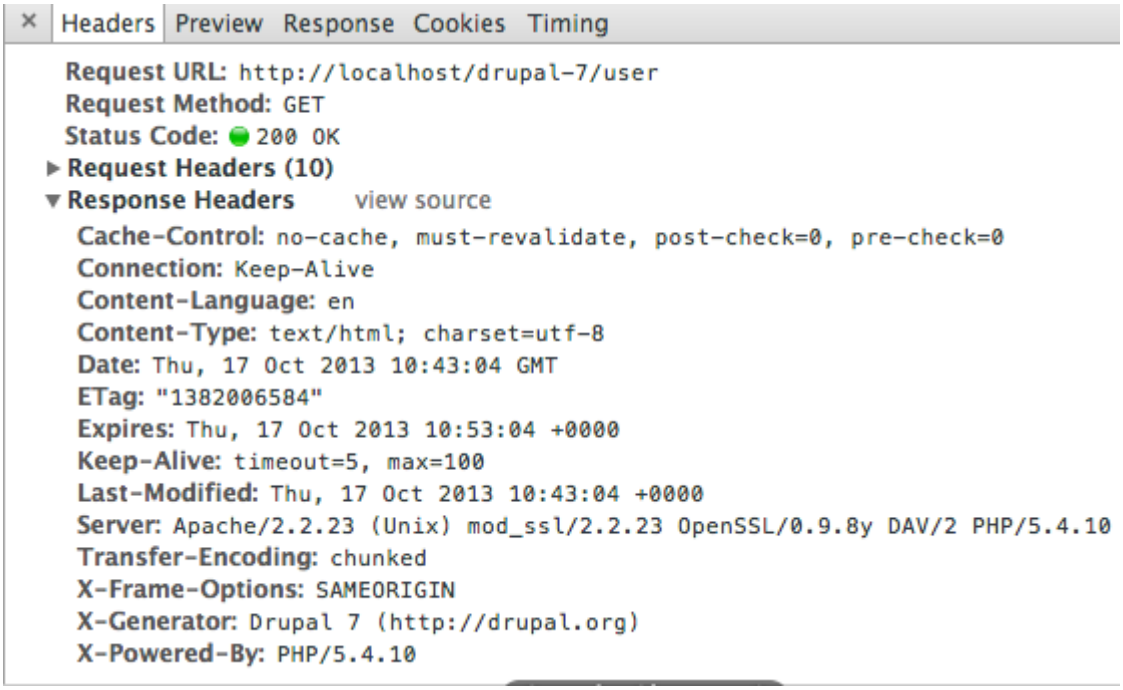
HTTP başlıqları HTTP mesajlarının vacib hissəsidir, çünki onlar müştəri və server arasında sorğu və ya cavab haqqında metadata ötürürlər. Başlıqlar açar-dəyər cütlərindən ibarətdir və məzmun növü, məzmun uzunluğu və keşləmə direktivləri kimi mesaj haqqında əlavə məlumat vermək üçün istifadə olunur. Başlıqları geniş şəkildə iki kateqoriyaya bölmək olar: sorğu başlıqları və cavab başlıqları. Sorğu başlıqları sorğu və müştərinin özü haqqında məlumat vermək üçün müştərilər tərəfindən göndərilir, cavab başlıqları isə cavab və server haqqında məlumat vermək üçün serverlər tərəfindən göndərilir. Müxtəlif növ başlıqları və onların rollarını başa düşmək veb proqramların düzgün işləməsi üçün çox vacibdir.

Sorğu başlıqları müştəri və edilən sorğu haqqında məlumat verməkdə mühüm rol oynayır. Bəzi tez-tez istifadə edilən sorğu başlıqlarına "Host", "İstifadəçi-Agent", "Qəbul", "Qəbul-Enkodlaşdırma", "Qəbul-Dil", "Avtorizasiya", "Keş-nəzarət", "Bağlantı" və "İstinad" daxildir. " "Host" başlığı domen adını və istəkdən asılı olaraq serverin port nömrəsini təyin edir, bu, server birdən çox veb-sayta sahib olduqda xüsusilə faydalıdır. "İstifadəçi-Agent" başlığı müştəri proqramı haqqında məlumat verir (məsələn, brauzerin növü və versiyası) və tez-tez analitika, sazlama və brauzerə məxsus məzmuna xidmət göstərmək üçün istifadə olunur. "Qəbul et" başlığı müştərinin idarə edə biləcəyi media növlərini, "Qəbul et-kodlaşdırma" başlığı isə müştərinin dəstəklədiyi kodlaşdırma sxemlərini (məsələn, gzip və ya deflate) göstərir. "Qəbul-Dil" başlığı serverlərə lokallaşdırılmış məzmuna xidmət etməyə imkan verən cavab üçün müştərinin üstünlük verdiyi dili çatdırır. "Authorization" başlığı müştərinin autentifikasiyası üçün etimadnamələrini təmin etmək üçün istifadə olunur, "Cache-Control" başlığı isə sorğu üçün keşləmə direktivlərini təyin etmək üçün istifadə edilə bilər. "Bağlantı" başlığı cari sorğudan sonra şəbəkə bağlantısının

açıq və ya qapalı olub-olmamasına nəzarət edir və "İstifadəçi" başlığı müştərini sorğu verməyə vadar edən istinad səhifəsinin URL-ni göstərir.

Cavab başlıqları isə server və yaradılan cavab haqqında məlumat verir. Ümumi cavab başlıqlarına "Məzmun növü", "Məzmun uzunluğu", "Məzmun-kodlaşdırma", "Keş-nəzarət", "Bağlantı", "Tarix", "ETag", "Müddəti bitir", "Son modifikasiya," daxildir. "Məkan", "Server", "Set-Cookie", "Dəyişiklik" və "WWW-Authenticate" başlıqlarını misal göstərə bilərik. "Məzmun növü" başlığı cavab orqanının media növünü göstərir (məsələn, mətn/html və ya proqram/json), "Məzmun-uzunluq" başlığı isə cavab orqanının ölçüsünü baytla müəyyən edir. "Content-Encoding" başlığı gzip və ya deflate kimi tətbiq olunan məzmun kodlaşdırmalarını göstərir. "Cache-Control" başlığı müştərilər və proksilər üçün keşləmə direktivlərini təmin edir, serverlərə onların məzmununun keşləşdirilməsinə nəzarət etməyə imkan verir. "Bağlantı" başlığı, sorğu analoqu kimi, şəbəkə bağlantısının vəziyyətinə nəzarət edir. "Tarix" başlığı cavabın yaradıldığı tarix və vaxtı göstərir, "ETag" başlığı isə sorğulanan resursun versiyası üçün unikal identifikator təqdim edir, keşin yoxlanılmasına və şərti sorğulara imkan verir. "Müddəti bitdi" başlığı cavabın köhnə hesab edilməli olduğu tarix və vaxtı təyin edir, "Son modifikasiya" başlığı isə resursun sonuncu dəfə dəyişdirildiyi tarix və vaxtı göstərir. "Yer" başlığı tələb olunan resursun yeni yerini təyin etmək üçün 3xx status kodları ilə birlikdə istifadə olunur. "Server" başlığı server proqram təminatı haqqında məlumat verir, "Set-Cookie" başlığı müştəridə kukilər təyin edir, "Dəyişdir" başlığı cavabı keşləmə zamanı hansı sorğu başlıqlarının nəzərə alınmalı olduğunu göstərir və "WWW-Authenticate" başlığı qorunan resursa daxil olmaq üçün tələb olunan autentifikasiya metodunu müəyyən edir.

HTTP başlıqlarının strukturuna aid bir nümunə şəkil 1.3-də göstərilmişdir.



Şəkil 1.3. HTTP başlıqlarının nümunəsi

Bağlantılar qurmaq və etibarlı məlumatların çatdırılmasını təmin etmək üçün HTTP əsas nəqliyyat qatında, adətən Transmissiya İdarəetmə Protokoluna (TCP) əsaslanır. Protokol HTTP/1.1, HTTP/2 və yeni yaranan HTTP/3 kimi versiyalar vasitəsilə inkişaf etdikcə, performansı yaxşılaşdırmaq, gecikməni azaltmaq və resursu minimuma endirmək üçün davamlı bağlantılar, multipleksləşdirmə, başlığın sıxılması və server təkanları kimi optimallaşdırmaları təqdim etdir. Bu elementləri birləşdirərək və onları daimi surətdə təkminləşdirərək HTTP internetin əsas protokolu olmaq ünvanını qorumaqda davam edir. Buna görə də bu protokolun araşdırılması və öyrənilməsi olduqca böyük əhəmiyyətə malikdir.

HTTP-in üstünlükləri və çatışmamazlıqları

Hypertext Transfer Protocol (HTTP) yarandığı gündən Ümumdünya Şəbəkəsinin onurğa sütunu olub, müştərilər və serverlər arasında əlaqə yaratmaqda mühüm rol oynayır. Bununla belə, hər hansı bir texnologiya kimi, onun da müsbət və mənfi cəhətləri var. HTTP-nin üstünlüklərini və çatışmazlıqlarını başa düşmək onun müasir internetə təsirini qiymətləndirmək və təkmilləşdirmə üçün potensial sahələri araşdırmaq vacibdir.

İlk öncə üstünlüklərdən danışaq:

- Davamsızlıq: HTTP davamlılığı olmayan protokoldur, yəni hər bir sorğu və cavab əvvəlki qarşılıqlı əlaqə haqqında məlumatı olmayan müstəqil əməliyyat kimi qəbul edilir. Bu dizayn seçimi server yükünü və mürəkkəbliyini azaldır, çünki serverlər sorğular arasında hər bir müştərinin vəziyyəti haqqında məlumat saxlamağa ehtiyac duymurlar. Davamsızlıq həmçinin veb proqramların miqyasını artırmağa kömək edir, çünki yük çoxsaylı serverlər arasında daha asan paylana bilər.
- Genişlənmə qabiliyyəti: HTTP genişlənmək üçün nəzərdə tutulmuşdur, lazım olduqda yeni metodlar, başlıqlar və status kodları əlavə etməyə imkan verir. Bu genişlənmə protokolun internetin dəyişən tələblərinə uyğunlaşaraq zamanla təkamül etməyə imkan verir. Məsələn, fərdi başlıqların və OPTIONS və PATCH kimi yeni metodların tətbiqi tərtibatçılara qabaqcıl funksionallığı tətbiq etməyə və tətbiqlərinin performansını optimallaşdırmağa imkan verdi.

Http protokolunun çatışmazlıqları isə bunlardan ibarətdir.

- Performans məhdudiyyətləri: Bir çox üstünlüklərinə baxmayaraq, HTTP bəzi xas performans məhdudiyyətlərinə malikdir. Protokolun ilkin versiyaları (HTTP/1.0 və HTTP/1.1) çoxsaylı TCP bağlantılarının istifadəsi və multipleksləşdirmənin olmaması səbəbindən gecikmə problemlərindən əziyyət çəkirdi ki, bu da xəttin bloklanmasına səbəb olur. HTTP/2 bu narahatlıqların bəzilərini multipleksləşdirmə və performansı artıran digər funksiyaları təqdim etməklə həll etsə də, xüsusilə yüksək gecikmə və ya etibarsız şəbəkə mühitlərində təkmilləşdirmə üçün hələ də yer var.
- Təhlükəsizlik problemləri: HTTP mahiyyət etibarilə təhlükəsiz deyil, çünki o, məlumatları şifrələmədən açıq mətnlə ötürür. Bu, onu dinləmə, ortada olan hücumlar və digər təhlükəsizlik təhdidlərinə qarşı həssas edir. Nəqliyyat Layer Təhlükəsizliyi (TLS) və ya Təhlükəsiz Sockets Layer (SSL) vasitəsilə şifrələmə əlavə edən HTTPS (HTTP Secure) tətbiqi bu narahatlıqları aradan qaldırmağa kömək etdi, lakin təhlükəsiz olmayan HTTP bağlantılarının davamlı yayılması istifadəçilər üçün risk olaraq qalır.

- Davamlılığın idarə edilməsi: Davamlılıq bəzi üstünlüklər təqdim etsə də, çoxlu sorğular üzrə davamlılığın idarə edilməsində çətinliklər yaradır. Bu məhdudiyyəti aradan qaldırmaq üçün tərtibatçılar tez-tez istifadəçi məlumatlarını saxlamaq üçün çərəzlər, sessiya saxlama və digər müştəri tərəfi mexanizmləri kimi üsullara etibar edirlər. Bununla belə, bu yanaşmalar təhlükəsizlik riskləri, məxfiliklə bağlı narahatlıqlar və tətbiqin dizaynında və tətbiqində artan mürəkkəblik yarada bilər.

Ümumilikdə, HTTP sadəliyi və genişlənmə qabiliyyəti sayəsində Ümumdünya Şəbəkəsinin böyüməsində və təkamülündə mühüm rol oynamışdır. Buna baxmayaraq, veb inkişaf etməyə davam etdikcə istifadəçilərin, sistemlərin yeni tələbləri yarandığından bu tələbləri qarşılaya biləcək olan HTTP3-ü təkminləşdirmək olduqca vacibdir.

1.4 Nüfuzetmə testləri və HTTP

Penetrasiya testi, həmçinin nüfuzetmə sınağı və ya etik hakerlik, kibertəhlükəsizlik mütəxəssislərinin zəiflikləri müəyyən etmək və onların təhlükəsizlik vəziyyətini qiymətləndirmək üçün təşkilatın sistemlərinə, şəbəkələrinə və ya proqramlarına kiberhücumları simulyasiya etməsi prosesidir. Penetrasiya testi təşkilatlara üzləşdikləri potensial riskləri anlamağa kömək edir, zəif tərəfləri aradan qaldırmağa və fəal şəkildə müdafiələrini gücləndirməyə imkan verir. HTTP Ümumdünya Şəbəkəsinin onurğa sütunu olduğundan təşkilatlar üçün potensial təhlükələrdən qorunmalarını təmin etmək üçün HTTP protokoluna əsaslanan veb proqramlarını və API-lərini hərtərəfli sınaqdan keçirmələri çox vacibdir.

HTTP kontekstində nüfuz sınağı adətən veb proqramların, serverlərin və əlaqəli infrastrukturun təhlükəsizliyini qiymətləndirmək üçün bir sıra metodologiyaları, alətləri və texnikaları əhatə edir. Proses tez-tez kəşfiyyatla başlayır, bu müddət ərzində tester domen adları, IP ünvanları və mövcud xidmətlər kimi hədəf tətbiq haqqında məlumat toplayır. Bu məlumat testerə tətbiqin hücum səthinin xəritəsini tərtib etməyə və potensial giriş nöqtələrini müəyyən etməyə kömək edir. Kəşfiyyatdan sonra testçilər proqram kodundakı boşluqlardan, server parametrlərində yanlış konfiqurasiyalardan

və ya HTTP protokolunun özündəki zəifliklərdən istifadə etmək kimi müxtəlif hücum üsullarından istifadə edirlər. Məsələn, testerlər icazəsiz giriş əldə etmək, həssas məlumatları oğurlamaq və ya proqram funksionallığını manipulyasiya etmək üçün SQL inyeksiyası, saytlar arası skript (XSS) və ya saytlararası sorğu saxtakarlığı (CSRF) kimi boşluqlardan istifadə etməyə cəhd edə bilər. Bundan əlavə, testçilər HTTP başlıqlarının, kukilərin və protokolun digər elementlərinin təhlükəsizliyini qiymətləndirə bilər.

Nüfuz testi prosesi zamanı testçilər hədəf tətbiqdəki zəiflikləri müəyyən etmək üçün tez-tez sadə və avtomatlaşdırılmış üsulların birləşməsindən istifadə edirlər. Veb tətbiqi zəiflik skanerləri və proksilər kimi avtomatlaşdırılmış alətlər məlum təhlükəsizlik problemlərini müəyyən etmək və tətbiqin ən yaxşı təcrübələrə uyğunluğunu qiymətləndirmək üçün istifadə edilə bilər. Bununla belə, birbaşa testi də vacibdir, çünki o, tətbiqin təhlükəsizlik vəziyyətinin daha dərin və nüanslı təhlilinə imkan verir. Manual test üsullarına kodun nəzərdən keçirilməsi, xüsusi zəifliklərin yönəldilmiş əl testi və avtomatlaşdırılmış alətlərin müəyyən edə bilməyəcəyi xüsusi və ya mürəkkəb funksionallıqdan istifadə etmək üçün yaradıcı yanaşmalar daxil ola bilər. Nüfüzetmənin yoxlanılması prosesi başa çatdıqdan sonra sınaqçı müəyyən edilmiş zəiflikləri, onların potensial təsirlərini və aradan qaldırılması üçün tövsiyələri əks etdirən ətraflı hesabat hazırlayır. Bu hesabat təşkilata aşkar edilmiş zəif cəhətləri prioritetləşdirməyə və həll etməyə imkan verir, HTTP protokoluna əsaslanan veb proqramlarının və infrastrukturunun ümumi təhlükəsizliyini yaxşılaşdırır.

Nəticə etibarilə, nüfuz sınağı HTTP protokolundan istifadə edən veb proqramların və API-lərin təhlükəsizliyinin təmin edilməsində mühüm aspektdir. Kiberhücumları simulyasiya edərək və zəiflikləri müəyyən etməklə təşkilatlar zəif tərəfləri aktiv şəkildə aradan qaldıra və potensial təhlükələrə qarşı müdafiələrini gücləndirə bilərlər. Avtomatlaşdırılmış alətləri birbaşa sınaq üsulları ilə birləşdirmək proqramın təhlükəsizlik vəziyyətinin hərtərəfli qiymətləndirilməsini təmin edir, təşkilatlara öz həssas məlumatlarını daha yaxşı qorumağa və getdikcə daha çox əlaqələnən rəqəmsal dünyada istifadəçilərinin etibarını qorumağa imkan verir.

Headers during penetration testing

HTTP başlıqları sorğu və cavab haqqında vacib metadata təmin edərək, müştərilər və serverlər arasında ünsiyyətdə mühüm rol oynayır. Beləliklə, onlar veb tətbiqləri və API-lərin nüfuz sınağı zamanı qiymətləndirmək üçün vacib komponentdir. HTTP başlıqlarının tədqiqi və manipulyasiyası təhlükəsizlik pozuntularına səbəb ola biləcək potensial zəiflikləri və yanlış konfiqurasiyaları aşkar edə bilər. Nüfuz testi kontekstində HTTP başlıqlarına diqqəti üç əsas sahəyə bölmək olar: etibarsız başlıqların müəyyən edilməsi, təhlükəsizlik başlıqlarının yanlış konfiqurasiyalarının sınaqdan keçirilməsi və icazəsiz giriş əldə etmək və ya imtiyazları artırmaq üçün başlıqlardan istifadə etmək. Nüfuz testində diqqət mərkəzində olan ilk sahə etibarsız başlıqların müəyyən edilməsidir. Bir çox veb proqram və serverlərə server proqram təminatının versiyası, istifadə olunan əsas texnologiyalar və ya sessiya identifikatorları kimi həssas məlumatı təsadüfən sızdıra bilən standart başlıqlar daxildir. Bu məlumat təcavüzkarlar üçün dəyərli ola bilər, onlara açıqlanmış texnologiyalardakı məlum zəifliklərdən və ya zəifliklərdən istifadə etmək üçün potensial imkanlar təqdim edə bilər. Bundan əlavə, testçilər mühafizəyə kömək edə biləcək Məzmun-Təhlükəsizlik-Siyasəti, Ciddi-Nəqliyyat-Təhlükəsizlik, X-Content-Type-Options, X-Frame-Options və X-XSS-Protection kimi təhlükəsizlik başlıqlarının istifadəsini qiymətləndirməlidirlər. XSS, klikləmə və məzmun inyeksiyası kimi ümumi hücumlar da bu qiymətləndirmə prosesinə daxildir.

Nüfuz testində diqqətin ikinci sahəsi təhlükəsizlik başlığının yanlış konfiqurasiyaları üçün sınaqdan keçirilməsindən ibarətdir. Yanlış konfiqurasiya edilmiş başlıqlar gözlənilməz nəticələrə səbəb ola bilər, məsələn, veb tətbiqinin təhlükəsizliyini zəiflətmək və ya təcavüzkarların giriş nəzarətlərini keçməsinə icazə vermək. Məsələn, yanlış konfiqurasiya edilmiş CORS (Mənbələrarası Resurs Paylaşımı) siyasəti təcavüzkarın məhdudlaşdırılmalı olan sorğular etməyinə imkan verə bilər. Eynilə, düzgün qurulmamış Cache-Control başlıqları həssas məlumatın keşləşdirilməsi və potensial olaraq ifşa edilməsi ilə nəticələnə bilər. Penetrasiya testçiləri təhlükəsizlik başlıqlarının konfiqurasiyasını hərtərəfli qiymətləndirməli və veb tətbiqini və onun

istifadəçilərini potensial təhlükələrdən qorumaq üçün onların düzgün şəkildə həyata keçirilməsini təmin etməlidirlər.

Nüfuz testində diqqətin üçüncü sahəsi icazəsiz giriş əldə etmək və ya imtiyazları artırmaq üçün olan başlıqların test edilməsindən ibarətdir. Təcavüzkarlar tez-tez "İstifadəçi", "İstifadəçi-Agent", "X-Yönləndirilmiş-For" kimi HTTP başlıqlarını və xüsusi başlıqları manipulyasiya edir, giriş nəzarətindən yan keçmək, imtiyazların artırılması və server tərəfindən istifadə etmək kimi hücumlar həyata keçirə bilirlər. Məsələn, təcavüzkar IP-əsaslı giriş nəzarətlərini keçərək öz IP ünvanlarını saxtalaşdırmaq üçün "X-Forwarded-For" başlığını manipulyasiya edə bilər. Başqa bir misal, təcavüzkarın başlıq dəyərlərinə zərərli məzmun yeritdiyi başlıq inyeksiyasıdır ki, bu da potensial olaraq server tərəfində kod icrasına, cavabın parçalanmasına və ya XSS hücumlarına səbəb olur. Penetrasiya testçiləri veb tətbiqinin HTTP başlıqları ilə işləməsini yoxlamalı və başlıq dəyərlərinin manipulyasiyası ilə istifadə edilə bilən potensial zəiflikləri sınamalıdır.

Yekun olaraq, HTTP başlıqları veb proqramlar və API-lər üçün nüfuz testinin mühüm aspektidir, çünki onlar təhlükəsizlik pozuntularına səbəb ola biləcək zəiflikləri və yanlış konfiqurasiyaları aşkar edə bilər. Təhlükəsiz başlıqların müəyyən edilməsinə, təhlükəsizlik başlıqlarının yanlış konfiqurasiyalarının sınanmasına və icazəsiz giriş əldə etmək və ya imtiyazları artırmaq üçün başlıqlardan istifadə etməyə diqqət yetirməklə, nüfuzetmə testçiləri tətbiqin təhlükəsizlik vəziyyətinin hərtərəfli qiymətləndirilməsini təmin edə və təşkilatlara potensial təhlükələri istismar edilməzdən əvvəl həll etməyə kömək edə bilər.

Başlıqların nüfuzetmə testləri zamanı modifikasiya edilməsi

HTTP proksiləri müştərilər və serverlər arasında yerləşən, HTTP sorğularının və cavablarının tutulmasına, yoxlanılmasına və dəyişdirilməsinə imkan verən vasitəçi serverlərdir. Onlar məzmunu keşləmə, yük balansı və veb trafikinin filtrasiyası kimi müxtəlif məqsədlərə xidmət edə bilər. Təhlükəsizlik mütəxəssisləri və nüfuzetmə testçiləri üçün HTTP proksiləri HTTP başlıqlarını təhlil etmək və manipulyasiya etmək, müştəri-server qarşılıqlı əlaqəsi üzərində daha çox nəzarəti təmin etmək və veb

proqramlar və API-lərdə potensial zəifliklərin aşkar edilməsini asanlaşdırmaq üçün əvəzsiz alətlərdir.

Təhlükəsizlik testində HTTP proksilərinin əsas istifadələrindən biri həm sorğularda, həm də cavablarda HTTP başlıqlarını dəyişdirməkdir ki, bu da testerlərə bu dəyişikliklərin hədəf tətbiqə təsirini müşahidə etməyə və təhlil etməyə imkan verir. Başlıqları dəyişdirməklə, sınaqçılar müxtəlif müştəri mühitlərini simulyasiya edə, giriş nəzarətlərini keçə və potensial olaraq server tərəfindəki zəifliklərdən istifadə edə bilərlər. Məsələn, nüfuzetmə testçisi fərqli brauzer və ya cihazı təqlid etmək üçün "İstifadəçi-Agent" başlığını dəyişə bilər və ya IP-əsaslı giriş nəzarətlərini potensial olaraq yan keçməklə IP ünvanlarını saxtalaşdırmaq üçün "X-Forwarded-For" başlığını dəyişdirə bilər. Bundan əlavə, testçilər giriş nəzarəti, məlumat sızması və ya daxiletmənin yoxlanılması ilə bağlı zəiflikləri araşdırmaq üçün "İstinad", "Mənşə" kimi başlıqları və fərdi başlıqları manipulyasiya edə bilər. Burp Suite, OWASP ZAP və Fiddler kimi alətlər təhlükəsizlik testi zamanı HTTP başlıqlarını tutmaq və dəyişdirmək üçün geniş funksiyalar təklif edən məşhur HTTP proksiləridir.

Mövcud başlıqları manipulyasiya etməkdən başqa, HTTP proksiləri yeni başlıqları daxil etmək və ya mövcud olanları HTTP mesajlarından silmək üçün də istifadə edilə bilər. Bu qabiliyyət test edənlərə xüsusi başlıqların mövcudluğu və ya olmamasının gözlənilməz davranışa və ya təhlükəsizlik zəifliyinə səbəb ola biləcəyi ssenariləri araşdırmaq imkanı verir. Məsələn, sınaqçı potensial başlıq inyeksiya zəifliklərini araşdırmaq üçün zərərli məzmundan ibarət fərdi başlıq əlavə edə və ya onun olub-olmamasının tətbiqin təhlükəsizlik vəziyyətinə təsirini yoxlamaq üçün "Məzmun-Təhlükəsizlik-Siyasət" kimi təhlükəsizlik başlığını silə bilər. Başlıqları yeritməklə və ya silməklə sınaqçılar tətbiqin müxtəlif hücum vektorlarına qarşı dayanıqlığını daha dərindən başa düşə və müvafiq əks tədbirlərin görülməsini təmin edə bilərlər.

Nəticə olaraq, HTTP proksiləri təhlükəsizlik testi üçün güclü alətlərdir və testerlərə həm sorğularda, həm də cavablarda HTTP başlıqlarını tutmaq, yoxlamaq və dəyişdirmək imkanı verir. Proksilərin istifadəsi vasitəsilə testçilər müxtəlif müştəri mühitlərini simulyasiya edə, giriş nəzarətlərini yan keçə və veb proqramlar və API-

lərdə potensial zəiflikləri aşkar edə bilərlər. Bu imkanlardan istifadə etməklə təhlükəsizlik mütəxəssisləri proqramın təhlükəsizlik vəziyyətini daha yaxşı qiymətləndirə və təşkilatlara potensial təhlükələri proaktiv şəkildə həll etməyə kömək edə bilərlər.

II FƏSİL. VEB PROKSİLƏR

2.1 Proksi serverlərin ümumi arxitekturası

Proksi server istifadəçi ilə İnternet arasında şlüz rolunu oynayan, müştəriyə internetdən məlumat tələb etmək və almaq imkanı verən vasitəçi serverdir. Proksi serverlər tez-tez veb trafikə nəzarət etmək və optimallaşdırmaqla təhlükəsizlik, məxfilik və performansı artırmaq məqsədilə istifadə olunur. Onlar müxtəlif məqsədlər üçün şirkətlər, təhsil müəssisələri və fərdi istifadəçilər tərəfindən quraşdırıla bilər. Bu fəsildə biz proksi serverlərdən istifadənin funksiyalarını, növlərini, üstünlüklərini və çatışmazlıqlarını, həmçinin onların necə qurulacağını müzakirə edəcəyik.

Proksi serverin əsas funksiyası müştəri ilə internet arasında vasitəçi kimi çıxış etməkdir. İstifadəçi veb səhifə və ya resurs tələb etdikdə, sorğu əvvəlcə proksi serverə göndərilir, sonra isə sorğu təyinat serverinə yönləndirilir. Təyinat serveri tələb olunan məlumatları yenidən proksi serverə göndərir, sonra isə məlumatı müştəriyə ötürür. Bu proses proksi serverə əlavə təhlükəsizlik, məxfilik və performans üstünlükləri təmin etməklə müştərinin İnternet trafikini idarə etməyə, filtrləməyə və izləməyə imkan verir.

Proksi serverlərin bir neçə növü var və hər biri öz xüsusi istifadə halları və üstünlükləri ilə seçilir. Məsələn, HTTP proksiləri; Çoxlu protokolları dəstəkləyən və daha yüksək səviyyədə anonimliyi təmin edən SOCKS proksiləri; Veb serverləri potensial təhlükələrdən qoruyan və daxil olan trafiki paylayan əks proksilər. Şəffaf proksilər istifadəçi sorğularını dəyişdirməyən və ya hər hansı səviyyədə anonimlik təmin etməyən başqa bir növdür. Onlar tez-tez təşkilatlar tərəfindən məzmunun filtrasiyasını tətbiq etmək və ya təkmilləşdirilmiş performans üçün məzmunu keşləmək məqsədilə istifadə olunur.

Proksi serverlərdən istifadə etməyin bir çox üstünlükləri var. Birincisi, onlar müştəri ilə internet arasında qorunma qatını təmin etməklə təhlükəsizliyi artırır və zərərli aktorların müştərinin sistemini pozmasını çətinləşdirir. İkincisi, onlar müştərinin IP ünvanını maskalamaqla məxfiliyi təkmilləşdirə, üçüncü tərəflərin onlayn fəaliyyətlərini izləməsini və ya monitorinqini çətinləşdirir. Üçüncüsü, proksi serverlər tez-tez daxil olan məzmunu keşləməklə, təyinat serverindəki yükü azaltmaqla və

müştəri üçün cavab müddətini sürətləndirməklə performansı artıra bilər. Nəhayət, onlar istifadəçilərə regionda kilidlənmiş veb-saytlara və xidmətlərə daxil olmağa imkan verən məzmuna dair coğrafi məhdudiyyətləri keçmək üçün istifadə oluna bilər.

Üstünlüklərinə baxmayaraq, proxy serverlərin bəzi çatışmazlıqları da var. Əhəmiyyətli bir mənfi cəhət ondan ibarətdir ki, onlar gecikmə tətbiq edə bilərlər, çünki sorğu təyinat yerinə çatmazdan əvvəl əlavə serverdən keçməlidir. Bundan əlavə, proksi serverlər oflayn olduqda və ya texniki çətinliklərlə üzləşdikdə potensial tək uğursuzluq nöqtəsinə çevrilə bilər. Bundan əlavə, bütün proksi serverlər eyni səviyyədə təhlükəsizlik, məxfilik və performans təmin etmir, yəni istifadəçilər etibarlı proksi provayderi diqqətlə seçməlidirlər. Nəhayət, bəzi veb-saytlar və xidmətlər məlum proksi serverlərdən gələn trafiki bloklaya bilər və onların coğrafi məhdudiyyətləri keçmək üçün faydalılığını məhdudlaşdıra bilər.

Proksi server qurmaq üçün istifadəçilərin bir neçə variantı var, məsələn, öz brauzerini və ya əməliyyat sistemini proksidən istifadə etmək üçün konfiqurasiya etmək, üçüncü tərəf proqramlarından istifadə etmək və ya hətta öz proxy serverini qurmaq bu variantlara daxildir. Proksi server seçərkən provayderin reputasiyası, təqdim edilən anonimlik səviyyəsi və proksi serverin yeri kimi amilləri nəzərə almaq vacibdir. Bizneslər və qabaqcıl istifadəçilər üçün xüsusi proxy serverin qurulması əlavə nəzarət və fərdiləşdirmə seçimləri təklif edə bilər. Bununla belə, bu seçim daha yüksək səviyyəli texniki təcrübə tələb edir, həmçinin server avadanlığı və texniki xidmət üçün əlavə xərclər tələb edə bilər.

Nəticə olaraq, proxy serverlər təhlükəsizlik, məxfilik və performans baxımından çoxsaylı üstünlüklər təklif edən dəyərli alətlərdir. Bununla belə, istifadəçilər mənfi cəhətləri diqqətlə nəzərdən keçirməli və müsbət təcrübə təmin etmək üçün etibarlı provayder və ya həll yolu seçməlidirlər. Fərdlər və təşkilatlar proksi serverlərin müxtəlif növlərini və onlardan istifadə hallarını başa düşərək məlumatlı qərarlar qəbul edə və onlayn təcrübələrini artırmaq üçün proksi serverlərin gücündən istifadə edə bilərlər.

Veb proksi xidmətləri HTTP və HTTPS trafikinin idarə edilməsinə və işlənməsinə diqqət yetirən xüsusi bir proksi server növüdür ki, bu da ilk növbədə internetə baxış

fəaliyyətlərini əhatə edir. Bu xidmətlər istifadəçilər İnternetdə gəzərkən əlavə məxfilik, təhlükəsizlik və funksionallıq səviyyəsini təmin etmək üçün nəzərdə tutulub.

Veb proksi xidmətləri istifadəçinin veb sorğularını nəzərdə tutulan təyinat serverinə yönləndirməzdən əvvəl onları tutaraq və emal etməklə fəaliyyət göstərir. Bu proses veb proksiyə sorğunu dəyişməyə və ya filtrləməyə, istifadəçinin IP ünvanını maskalamaqla, məzmunun filtrasiyasını tətbiq etməklə xüsusi veb-saytlara girişi bloklamaqla məxfiliyi gücləndirməyə imkan verir. Təyinat serveri cavab verdikdə, veb proksi xidməti tələb olunan məzmunu əldə edir və onu istifadəçiyə qaytarır, ümumi baxış təcrübəsini yaxşılaşdırmaq üçün tez-tez keşləmə və digər optimallaşdırma üsullarını tətbiq edir.

Veb proxy xidmətindən istifadənin əsas üstünlüklərindən biri onun təklif etdiyi təkmilləşdirilmiş məxfilikdir. İstifadəçi və təyinat serveri arasında vasitəçi kimi çıxış edərək, veb proksi xidmətləri istifadəçinin IP ünvanını effektiv şəkildə gizlədə bilər, bu da üçüncü tərəflərin onlayn fəaliyyətlərini izləmələrini və ya monitorinqlərini çətinləşdirir. Bu xüsusiyyət həssas məlumatlara daxil olarkən və ya arzuolunmaz diqqəti cəlb edə biləcək fəaliyyətlərlə məşğul olarkən şəxsiyyətlərini qorumaq istəyən istifadəçilər üçün xüsusilə dəyərli ola bilər. Bundan əlavə, istifadəçilərə dünyanın istənilən yerindən bölgə ilə bağlı saytlara və xidmətlərə daxil olmaq imkanı verən məzmuna dair coğrafi məhdudiyyətləri aşmaq üçün veb proksi xidmətlərindən istifadə edilə bilər.

Veb proksi xidmətlərinin digər əhəmiyyətli üstünlüyü onların təmin edə biləcəyi artan təhlükəsizlikdir. Veb trafiki filtrləmək və izləməklə bu xidmətlər istifadəçiləri zərərli proqram, fişinq hücumları və digər zərərli fəaliyyətlər kimi müxtəlif onlayn təhlükələrdən qorumağa kömək edə bilər. Bundan əlavə, veb proksi xidmətləri məzmunun filtrasiyası siyasətlərini tətbiq edə bilər. Proksilər həmçinin təşkilatlara və qurumlara xüsusi veb-saytlara və ya məzmun kateqoriyalarına girişi məhdudlaşdırmağa imkan verir, daha təhlükəsiz və daha idarə olunan baxış mühitini yaratmağı təşviq edir.

Çoxsaylı üstünlüklərinə baxmayaraq, veb proxy xidmətlərinin də bəzi məhdudiyyətləri var. Əsas çatışmazlıqlardan biri gecikmənin artması potensialıdır, çünki veb sorğuları

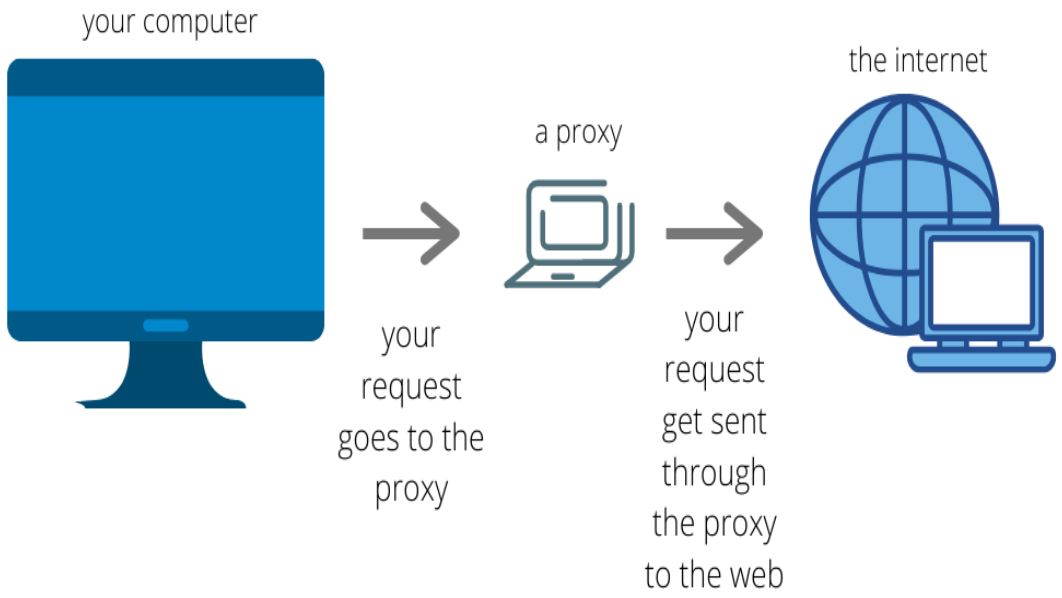
təyinat yerinə çatmazdan əvvəl əlavə serverdən keçməlidir. Bu problem xüsusilə veb-proxy xidməti istifadəçilərdə təyinat serverindən uzaqda yerləşdiyi halda özünü göstərə bilər. Bundan əlavə, veb proksi xidmətləri tərəfindən təmin edilən məxfilik və təhlükəsizlik səviyyəsi provayderdən asılı olaraq əhəmiyyətli dərəcədə dəyişə bilər, bəzi xidmətlər istifadəçi məlumatlarını qeyd edir və ya qeyri-adekvat şifrələmə təklif edir. İstifadəçilər provayderi seçməzdən əvvəl veb proksi xidmətinin məxfilik siyasətini və təhlükəsizlik xüsusiyyətlərini diqqətlə qiymətləndirməlidirlər.

Veb proxy xidmətini seçərkən bir neçə amili nəzərə almaq lazımdır. İstifadəçilər provayderin reputasiyasını, təklif olunan məxfilik və təhlükəsizlik səviyyəsini, xidmətin yerini və brauzerin performansına potensial təsirini qiymətləndirməlidirlər. Reklamın bloklanması, zərərli proqramların skan edilməsi və ya məzmunun filtrasiyası seçimləri kimi xidmət tərəfindən təmin edilən hər hansı əlavə funksiyaları və ya alətləri nəzərə almaq da vacibdir. Müəssisələr və təşkilatlar üçün qabaqcıl funksiyalar, xüsusi dəstək və daha yaxşı performans təklif edən ödənişli veb proksi xidmətini nəzərdən keçirmək lazım ola bilər.

Nəticə olaraq, veb proksi xidmətləri İnternetə baxarkən məxfiliyi, təhlükəsizliyi və performansı artırmaq üçün dəyərli vasitələrdir. Bu xidmətlərin üstünlüklərini, məhdudiyyətlərini və istifadə hallarını başa düşməklə, istifadəçilər məlumatlı qərarlar qəbul edə və xüsusi ehtiyacları üçün düzgün veb proksi xidmətini seçə bilərlər. Düzgün seçildikdə və konfiqurasiya edildikdə, veb proksi xidmətləri ümumi baxış təcrübəsini əhəmiyyətli dərəcədə yaxşılaşdıra bilər, istifadəçilərə onlayn dünyada naviqasiya zamanı daha çox nəzarət və rahatlıq təmin edə bilər.

Proksilərin ümumi arxitekturası

Proksi, kompüter şəbəkələri və internet kontekstində müştəri ilə hədəf server və ya resurs arasında oturan vasitəçi serverdir. Proksilər təhlükəsizliyin təkmilləşdirilməsi, məxfiliyin artırılması və ya şəbəkə performansının optimallaşdırılması kimi müxtəlif vəzifələri yerinə yetirmək üçün nəzərdə tutulub. Proksilərin arxitekturası onların xüsusi məqsədindən və həyata keçirilməsindən asılı olaraq olduqca müxtəlif ola bilər.



Şəkil 2.1. Proksi serverin ümumi arxitekturası

Proksi serverlərin növləri

Hər biri özünəməxsus arxitekturası və məqsədi olan bir neçə növ proksilər mövcuddur:

- İrəli Proksi: İrəli proksi müştəri cihazları (məsələn, kompüterlər və ya mobil cihazlar) və hədəf server arasında yerləşdirilir. İstifadəçilər sorğularını proksi serverə göndərir, sonra isə onları nəzərdə tutulan təyinat yerinə yönləndirir. İrəli proksilər keşləmə, məzmunun filtrlənməsi və ya müştəri sorğularının anonimləşdirilməsi üçün istifadə edilə bilər.
- Tərs Proksi: Tərs proksi hədəf server və müştəri cihazları arasında yerləşdirilir. O, daxil olan sorğular üçün vasitəçi kimi xidmət edir, onları müvafiq backend serverlərinə paylayır. Əks proksilər yük balansını, təhlükəsizliyi və tətbiq performansını yaxşılaşdırmağa kömək edə bilər.
- Şəffaf Proksi: Şəffaf bir proksi müştəri sorğularını və ya cavablarını dəyişdirməyən yönləndirici proksi növüdür. O, müştərinin xəbəri olmadan işləyir, tez-tez keşləmə və ya filtrləmə məqsədləri üçün istifadə olunur.

- Anonim Proksi: Anonim proksi müştərinin IP ünvanını gizlədir, hədəf serverin müştərini birbaşa müəyyən etməsinə mane olmaqla məxfilik və təhlükəsizliyin qorunmasına kömək edir.
- Yüksək Anonimlik təmin edən proksilər: Yüksək anonimlik proksiləri, hədəf serverə təqdim etdiyi IP ünvanını müntəzəm olaraq dəyişdirərək, anonim proksilərə nisbətən daha yüksək səviyyədə məxfilik mühafizəsini təmin edir.
- Datacenter Proksi: Datacenter proksiləri məlumat mərkəzlərində yerləşdirilir və sürətli və sabit əlaqələri təmin edir. Bu proksilər yüksək sürətli əlaqə tələb edən tapşırıqlar üçün istifadə olunur.

Proksi protokolları

Proksilər müştərilər və hədəf serverlərlə necə əlaqə saxladıqlarını müəyyən edən müxtəlif protokollardan istifadə etməklə işləyə bilər:

- HTTP Proksi: HTTP proksiləri veb trafiki, xüsusilə HTTP protokolu ilə işləyir. Onlar veb məzmununu keşləyə, sorğuları süzə və ya istifadəçiləri anonimləşdirə bilər.
- HTTPS Proksi: HTTPS proksiləri HTTP proksilərinə bənzəyir, lakin şifrələnmiş HTTPS trafiki ilə işləyir. Onlar müştərilər və hədəf serverlər arasında məlumat ötürülməsini təmin etməyə kömək edir.
- SOCKS Proksi: SOCKS proksiləri HTTP/HTTPS proksilərindən daha çox yönlüdür, çünki onlar təkcə veb trafiki deyil, müxtəlif protokollardan gələn trafiki idarə edə bilirlər. Onlar tez-tez torrenting və ya geo məhdudiyyətləri keçmək kimi vəzifələr üçün istifadə olunur.

Proksi serverlərin quraşdırılma arxitekturaları

Proksilər verilmiş şəbəkə və ya tətbiqin xüsusi tələblərindən asılı olaraq müxtəlif arxitekturalardan istifadə etməklə yerləşdirilə bilər:

- Tək Proksi: Tək bir proksi bütün müştəri sorğularını idarə edən bir proxy server ilə ən sadə şəkildə yerləşdirilməlidir. Proksi server yükün öhdəsindən gələ bilmirsə, bu arxitektura əlaqənin olduqca zəifləməsi ilə nəticələnə bilər.

- Proksi Zəncirləməsi: Proksi zəncirləməsi, hər bir proksi yönləndirmə sorğusu ilə bir-birinin ardınca bir neçə proksi serveri birləşdirməyi nəzərdə tutur. Bu arxitektura çox sayda əlavə qatlar əlavə etməklə məxfiliyi və təhlükəsizliyi yaxşılaşdıra bilər.
- Balanslaşdırılmış Proksi: Balanslaşdırılmış proksi, daxil olan sorğuları yaymaq üçün çoxlu proksi serverlərdən istifadə edir, performans və etibarlılığı yaxşılaşdırmağa kömək edir. Bu arxitekturaya DNS yük balansı, dəyirmi sistemli alqoritmlər və ya məzmuna əsaslanan marşrutlaşdırma kimi müxtəlif texnologiyalar aid ola bilər.

Yekun olaraq, proksilərin arxitekturası müxtəlifdir, müxtəlif növlər, protokollar və yerləşdirmə variantları mövcuddur. Proksilər təhlükəsizlik, məxfilik və şəbəkə performansının yaxşılaşdırılması da daxil olmaqla geniş məqsədlərə xidmət edə bilər. Proksinin xüsusi arxitekturası onun nəzərdə tutulan istifadə vəziyyətindən və onu yerləşdirən istifadəçilərin və ya təşkilatların ehtiyaclarından asılı olaraq dəyişir.

Təhlükəsizlik cəhətdən proksilər

Veb proksiləri internetdən istifadə və veb resurslara çıxış kontekstində təhlükəsizlik üçün əhəmiyyətli təsirlərə malikdir. Onlar istifadəçilərin və təşkilatların ümumi təhlükəsizliyini artıra biləcək çoxsaylı üstünlüklər və imkanlar təklif edirlər.

Təhlükəsizlik nöqteyi-nəzərindən veb proksilər internetə baxarkən istifadəçi məxfiliyini və anonimliyini qorumağa kömək edə bilər. Müştərinin IP ünvanını maskalamaq və sorğu başlıqlarını potensial olaraq dəyişdirməklə, veb proksilər hədəf serverlər və ya üçüncü tərəflər üçün fərdi istifadəçiləri müəyyən etməyi və ya izləməyi çətinləşdirir. Bu, ciddi internet senzurası olan ölkələrdə və ya hökumət nəzarətindən narahat olan istifadəçilər üçün xüsusilə faydalıdır. Veb proksiləri həmçinin müştərinin şəxsiyyətini daha da aydınlaşdıran "İstifadəçi-Agent" və ya "İstifadəçi" başlıqları kimi potensial müəyyən edilə bilən məlumatları ehtiva edən başlıqları çıxarmaq və ya dəyişdirmək üçün konfiqurasiya edilə bilər.

Veb proksiləri zərərli məzmuna və vebsaytlara girişin qarşısını almaq üçün vacib vasitə ola bilər. Administratorlar məlum zərərli URL-ləri bloklamaq üçün veb proksilərini

konfiqurasiya edə bilər və ya potensial zərərli məzmunu müəyyən etmək və bloklamaq üçün dərin paket yoxlaması kimi məzmun filtrləmə üsullarından istifadə edə bilərlər. İstifadəçilərin zərərli saytlara daxil olmasının qarşısını almaqla veb proksilər müştəriləri zərərli proqramlardan, fişinq hücumlarından və digər təhlükəsizlik təhdidlərindən qorumağa kömək edir. Bundan əlavə, veb proksilərin təhlükə kəşfiyyatı lentləri ilə inteqrasiyası olaraq real vaxtda ən son təhlükələri müəyyən etmək və bloklamaq qabiliyyətlərini artıra bilərlər.

Təhlükəsizlik baxımından veb proksilərin digər mühüm cəhəti onların müştərilər və hədəf serverlər arasında təhlükəsiz ünsiyyəti təmin etmək qabiliyyətidir. HTTPS bağlantılarının istifadəsini tələb etməklə, veb proksilər müştəri ilə hədəf server arasında ötürülən məlumatların şifrlənməsini təmin edərək, onları dinləmədən və ya ortadakı adam hücumlarından qoruyur. Bəzi veb proksi-serverlər hətta SSL/TLS yoxlamasını həyata keçirə, şifrələnmiş trafiki yenidən şifrələmədən və hədəf serverə yönləndirmədən əvvəl potensial təhlükəsizlik təhdidləri üçün şifrəni aça və təhlil edə bilər. Bu texnika məxfiliklə bağlı narahatlıqları artırsa da, o, şəbəkələrini şifrələnmiş trafikdə gizlənmiş təhlükəsizlik təhdidlərindən izləmək və qorumaq istəyən təşkilatlar üçün vacib bir vasitə ola bilər.

Veb proksiləri həmçinin təşkilatın təhlükəsizlik infrastrukturunun ayrılmaz hissəsi ola bilər. Həmçinin proksilər təhlükəsizlik siyasətlərini həyata keçirmək və şəbəkə monitorinqini asanlaşdırmaq üçün nəzarət nöqtəsi kimi çıxış edirlər. Bütün veb trafikini mərkəzləşdirilmiş proxy vasitəsilə yönləndirməklə, təşkilatlar girişə nəzarət, məzmunun filtrasiyası və şəbəkə istifadəsi kimi siyasətləri həyata keçirə və tətbiq edə bilərik. Bu, idarəçilərə internetdən istifadəni daha effektiv idarə etməyə və təhlükəsizliyini təmin etməyə imkan verir, işçilərin məqbul istifadə siyasətlərinə əməl etməsini və təhlükəsizlik təhdidlərindən qorunmasını təmin edir. Əlavə olaraq, veb proksi-nüsxələri hərtərəfli, çoxtərəfli təmin etmək üçün müdaxilənin aşkarlanması/profilaktikası sistemləri (IDS/IPS), təhlükəsizlik məlumatı və hadisələrin idarə edilməsi (SIEM) həlləri və yeni nəsil təhlükəsizlik divarları kimi digər təhlükəsizlik alətləri və sistemləri ilə inteqrasiya oluna bilər

Nəticə etibarilə, veb proksi-serverlər istifadəçi məxfiliyini qorumaq, zərərli məzmuna girişi bloklamaq, təhlükəsiz ünsiyyəti təmin etmək və təşkilatın təhlükəsizlik infrastrukturunun mühüm komponenti kimi xidmət etməklə internet istifadəçilərinin və təşkilatların təhlükəsizliyinin artırılmasında mühüm rol oynayır. Veb proksilərinin müxtəlif təhlükəsizlik imkanlarını başa düşmək və istifadə etməklə, təşkilatlar ümumi təhlükəsizlik vəziyyətini əhəmiyyətli dərəcədə yaxşılaşdıra və özlərini kibertəhlükələrin inkişaf edən mənzərəsindən daha yaxşı qoruya bilərlər.

2.2 Brauzerlər və proksi serverlər

İnternet gündəlik həyatımızın ayrılmaz hissəsinə çevrilib, bizə ünsiyyət, araşdırma, əyləncə və daha çox şeylər üçün sonsuz imkanlar təqdim edir. Bizim internetə çıxışımızı və onunla qarşılıqlı əlaqəmizi asanlaşdıran iki əsas komponent veb brauzerlər və proxy serverlərdir.

Veb brauzer istifadəçilərə veb səhifələr, şəkillər, videolar və digər rəqəmsal resurslar da daxil olmaqla internetdən əldə etmək və göstərmək imkanı verən proqram təminatıdır. Veb-brauzerlər veb məzmunun əsas dili olan HTML-ni (Hypertext Markup Language) şərh edir və onu vizual olaraq cəlbedici və naviqasiya oluna bilən interfeysə çevirir.

Bu gün ən populyar veb brauzerlərdən bəziləri Google Chrome, Mozilla Firefox, Apple Safari, Microsoft Edge və Operadır. Hər bir brauzerin özünəməxsus xüsusiyyətləri, performans imkanları və fərdiləşdirmə seçimləri var və onlar müxtəlif istifadəçi seçimləri və ehtiyaclarına cavab verir. Veb brauzerlərin ardıcıl inkişafı və təkmilləşdirilməsi müasir internet təcrübəsinin formalaşmasında mühüm rol oynamışdır.

Proksi server istifadəçinin kompüteri ilə internet arasında vasitəçi kimi çıxış edir, mahiyyətcə istifadəçi adından sorğuları emal edən “aralıq” kimi fəaliyyət göstərir. Brauzerlər proksi serverlər ilə inteqrasiya onlunduqda aşağıdakı təhlükəsizlik prinsiplərinə nail olmaq olar.

- Təkminləşdirilmiş təhlükəsizlik. Proksi serverdən istifadə edərkən sizin IP ünvanınız (cihazınız üçün unikal identifikator) maskalanır və üçüncü tərəflərin

onlayn fəaliyyətlərinizi izləməsini çətinləşdirir. Bu, maliyyə və ya tibbi qeydlər kimi həssas məlumatlara daxil olarkən və ya ictimai Wi-Fi şəbəkələrindən istifadə edərkən məxfiliyinizi qorumaq üçün xüsusilə faydalı ola bilər.

- Təhlükəsizlik səviyyəsinin artırılması. Proksi serverləri viruslar və zərərli proqramlar kimi zərərli məzmunu cihazınıza çatmazdan əvvəl filtrləmək üçün konfiqurasiya edilə bilər. Bundan əlavə, onlar sisteminizi potensial təhlükələrdən qoruyaraq məlum zərərli veb-saytlara girişi blok etməyə kömək edə bilər.
- Təkminləşdirilmiş performans. Bəzi hallarda, proksi serverlər tez-tez daxil olan veb səhifələri keşləmək və istifadəçilərə daha tez xidmət göstərməklə baxış performansını yaxşılaşdırmağa kömək edə bilər. Bu, məhdud şəbəkə genişliyi və ya yavaş internet bağlantısı olan istifadəçilər üçün xüsusilə faydalı ola bilər.

2.3 Sistem səviyyəli və proqram səviyyəli proksilər

Sistem miqyaslı və tətbiqetmə miqyaslı proksilər proksi server vasitəsilə internet trafikini yönləndirmək üçün iki ümumi yanaşmadır. Proksi serverlər müştəri ilə internet arasında vasitəçi rolunu oynayır, performansı, təhlükəsizliyi və məxfiliyi yaxşılaşdırmağa kömək edir. Bu iki proksi növü arasındakı əsas fərqləri başa düşmək şəbəkə konfiqurasiyaları haqqında qərarlar qəbul etmək, optimal və təhlükəsiz əlaqə təmin edə bilmək üçün vacibdir.

Qlobal proksilər kimi də tanınan sistem səviyyəli proksilər əməliyyat sistemi səviyyəsində işləyir və cihazda işləyən bütün proqramlara və proseslərə təsir göstərir. Bu o deməkdir ki, tətbiqdən asılı olmayaraq internet sorğusu verildikdə, nəzərdə tutulan ünvana çatmazdan əvvəl trafik proxy server vasitəsilə yönləndirilir. Sistem geniş proksinin əsas üstünlüyü ondan ibarətdir ki, o, internet trafikini idarə etmək üçün ardıcıl və mərkəzləşdirilmiş üsul təqdim edir. O, bütün məlumatların ardıcıl olaraq proksi vasitəsilə yönləndirilməsini təmin edir, məlumat sızması riskini azaldır. Əlavə olaraq, bütün proqramlardan gələn trafiki idarə etməklə, sistem geniş proksiləri cihazda proksi parametrlərinin konfiqurasiyası və saxlanması prosesini sadələşdirir.

Proqram səviyyəli proksilər isə xüsusi olaraq fərdi proqramlara konfiqurasiya edilir və tətbiq edilir. Bu o deməkdir ki, yalnız təyin edilmiş proqramdan gələn trafik proxy server vasitəsilə yönləndirilir, digər proqramlar isə birbaşa internetə qoşulur. Tətbiq səviyyəli proksilər daha çox çeviklik təklif edir, çünki onlar istifadəçilərə proksi parametrlərini hər bir proqram üçün müstəqil şəkildə uyğunlaşdırmağa imkan verir. Bu, xüsusilə müxtəlif proqramlar üçün müxtəlif təhlükəsizlik siyasətlərini tətbiq etmək istəyən təşkilatlar və ya başqaları ilə birbaşa əlaqə saxlayaraq müəyyən tətbiqlərdə regiona aid məzmuna daxil olmaq məcburiyyətində olan istifadəçilər üçün faydalı ola bilər.

Bununla belə, proqram səviyyəli proksilərdən istifadə etməyin bəzi potensial çatışmazlıqları var. Birincisi, hər bir proqram üçün proksi parametrlərinin saxlanması çox vaxt apara bilər və səhvlərə meylli ola bilər, xüsusən də istifadəçi hər bir proqram üçün tələb olunan xüsusi konfiqurasiya parametrləri ilə tanış deyilsə. Bu, uyğun olmayan təhlükəsizlik siyasətlərinə və şəbəkə davranışına, həmçinin yanlış konfiqurasiya ehtimalının artmasına səbəb ola bilər. Bundan əlavə, bəzi proqramlar proksidən istifadəni dəstəkləməyə bilər və ya məhdud proksi dəstəyinə malik ola bilər ki, bu da bütün proqramlarda ardıcıl proxy siyasətini tətbiq etməyi çətinləşdirə bilər.

Həmçinin, sistem geniş proksilərin də öz məhdudiyyətləri ola bilər. Məsələn, onlar fərdi proqramlar üçün trafikə nəzarətdə eyni dərəcədə asanlaşdırmanı təmin edə bilməzlər. Müəyyən proqramlar üçün xüsusi proksi parametrləri lazımdırsa, qlobal proksi konfiqurasiyası lazımi çevikliyi təklif etməyə bilər. Bundan əlavə, sistem səviyyəli proksiləri müəyyən proqramlar və ya xidmətlər, xüsusən optimal performans və ya funksionallıq üçün birbaşa bağlantılara əsaslanan proqramlar ilə uyğunluq problemlərinə səbəb ola bilər. Belə hallarda istifadəçilər sistem səviyyəli proksisini müvəqqəti olaraq söndürməli və ya həmin xüsusi tətbiqlər üçün istisnaları konfiqurasiya etməlidirlər.

Həm sistem səviyyəli, həm də proqram səviyyəli proksilər fərqli üstünlüklər və çatışmazlıqlar təklif edir. Sistem səviyyəli proksilər bütün tətbiqlər üzrə internet trafikini idarə etmək üçün mərkəzləşdirilmiş və ardıcıl üsul təqdim edir, lakin fərdi proqramlar üçün eyni səviyyədə nəzarət təklif etməyə bilər. Tətbiq səviyyəli proksilər

isə proksi parametrlərinə daha ətraflı nəzarət etməyə imkan verir, lakin bir çox proqramda idarə etmək və saxlamaq daha çətin ola bilər. Hansı yanaşmanın ən yaxşı olduğuna qərar verərkən, istənilən nəzarət səviyyəsi, konfiqurasiya və texniki xidmətin asanlığı və istifadə olunan proqramların xüsusi tələbləri kimi amilləri nəzərə almaq vacibdir.

Sistem səviyyəli proksi OS səviyyəsində işləyir, yəni sorğu yaradan proqramdan və ya prosesdən asılı olmayaraq cihazdan gələn bütün internet trafikini kəsir. Bu, əməliyyat sisteminin şəbəkə parametrlərini müəyyən edilmiş proksi server vasitəsilə bütün trafiki yönləndirmək üçün konfiqurasiya etməklə əldə edilir. Proksi serverin və müştəri cihazının tələb və imkanlarından asılı olaraq HTTP, HTTPS, SOCKS və FTP daxil olmaqla, sistem səviyyəli proksilər üçün müxtəlif protokollardan istifadə edilə bilər.

Sistem səviyyəli proksilərin əhəmiyyətli faydalarından biri də onların bütün tətbiqlərdə ardıcıl təhlükəsizlik siyasətini tətbiq etmək qabiliyyətidir. Bütün trafiki bir proksi server vasitəsilə yönləndirməklə, istifadəçilər məzmunun filtrlənməsi, zərərli proqramların skan edilməsi və məlumatların şifrələnməsi kimi təhlükəsizlik tədbirlərini vahid şəkildə tətbiq edə bilərlər. Bu, cihazları fişinq hücumları, zərərli proqram təminatı və icazəsiz məlumat girişi kimi müxtəlif təhlükələrdən qorumağa kömək edə bilər. Bundan əlavə, sistem səviyyəli proksi-nüsxələri şəbəkə fəaliyyətini izləmək və qeyd etmək üçün də istifadə oluna bilər ki, bu da inzibatçılara potensial təhlükəsizlik problemlərini proaktiv şəkildə müəyyən etməyə və həll etməyə imkan verir.

Sistem geniş proksilərin digər üstünlüyü onların məxfiliyin artırılmasındakı roludur. Bütün internet trafiki proksi serverdən keçdiyi üçün müştərinin IP ünvanı proksi serverin IP ünvanı ilə əvəz olunur və müştərinin həqiqi yerini və şəxsiyyətini effektiv şəkildə maskalayır. Bu, geo-məhdud məzmuna daxil olmaq və ya müəyyən hökumətlər tərəfindən tətbiq edilən senzura tədbirlərindən yan keçmək istəyən istifadəçilər üçün xüsusilə faydalı ola bilər. Bundan əlavə, bəzi proksi serverlər məxfiliyi daha da artırmaq və üçüncü tərəflərin dinləmələrinin qarşısını almaq üçün SSL/TLS şifrələməsi və məlumatların çaşdırılması kimi üsullardan istifadə edir.

Sistemin geniş proksiləri, xüsusən də birdən çox cihazın eyni şəbəkəyə qoşulduğu korporativ və ya təhsil parametrlərində şəbəkə performansını yaxşılaşdırmağa kömək edə bilər. Tez-tez daxil olan məzmunu keşləməklə, proksi serverlər bant genişliyi istehlakını azalda və şəbəkəyə yükü azalda bilər, nəticədə son istifadəçilər üçün daha sürətli gəzinti sürəti olur. Bu, məhdud bant genişliyi olan və ya xüsusi tapşırıqlar və ya tətbiqlər üçün şəbəkə resurslarını prioritetləşdirmək istəyən təşkilatlar üçün xüsusilə faydalıdır.

Sistem geniş proksilərin çoxsaylı üstünlüklərinə baxmayaraq, nəzərə alınmalı bəzi potensial mənfi cəhətlər var. Problemlərdən biri optimal performans üçün birbaşa bağlantılar tələb edən müəyyən proqramlar və ya xidmətlərlə uyğunluq problemlərinin yaranmasıdır. Belə hallarda, düzgün funksionallığı təmin etmək üçün administratorlar istisnaları konfiqurasiya etməli və ya proksi parametrləri daxilində qaydaları müəyyən etməlidirlər. Bundan əlavə, sistem geniş proksidən istifadə bir uğursuzluq nöqtəsi ilə nəticələnə bilər. Proksi serverdə dayanma vaxtı və ya performans problemləri olarsa, cihaz üçün bütün internet bağlantısı təsirlənə bilər. Bu riski azaltmaq üçün təşkilatlar trafiki paylamaq və ardıcıl şəbəkə performansını təmin etmək üçün lazımsız proksi serverlərdən və ya yük balanslaşdırma üsullarından istifadə edə bilərlər.

Yekun olaraq, sistem geniş proksiləri bütün cihazda internet trafikini, təhlükəsizliyi və məxfiliyi idarə etmək üçün hərtərəfli həll yolu təqdim edir. ƏS səviyyəsində fəaliyyət göstərərək və bütün şəbəkə bağlantılarını təyin edilmiş proxy server vasitəsilə yönləndirməklə, bu proksilər təhlükəsizlik siyasətlərini tətbiq etmək, məxfiliyi artırmaq və şəbəkə performansını optimallaşdırmaq üçün mərkəzləşdirilmiş yanaşma təklif edir. Bununla belə, sistem səviyyəli proksi həllini həyata keçirərkən potensial uyğunluq problemlərini və gələcək ehtiyyacları da nəzərə almaq vacibdir. Nəhayət, sistem geniş proksilərin nüanslarını başa düşmək istifadəçilərə öz şəbəkə konfiqurasiyaları haqqında əsaslandırılmış qərarlar qəbul etməyə və onların əlaqə və təhlükəsizlik ehtiyaclarının effektiv şəkildə ödənilməsini təmin etməyə kömək edə bilər.

Proqram səviyyəli proksilər

Tətbiq miqyasında və ya proqram səviyyəli proksi kimi də tanınan proqram əsaslı proksilər istifadəçilərə hər bir proqram əsasında internet trafikinin idarə edilməsini fərdiləşdirməyə imkan verməkdə mühüm rol oynayır. Bu proksilər istifadəçilərə xüsusi proqramları təyin edilmiş proksi serverlər vasitəsilə yönləndirməyə imkan verir, digər proqramlara isə birbaşa internetə qoşulmağa imkan verir. Bu nəzarət səviyyəsi uyğunlaşdırılmış təhlükəsizlik siyasəti və geo məhdudlaşdırılmış məzmuna seçmə giriş kimi bir sıra üstünlüklər təklif edir. Tətbiq əsaslı proksilərin üstünlüklərini və mənfi cəhətlərini tam qiymətləndirmək üçün onların funksionallığını, istifadə hallarını və potensial problemləri araşdırmaq vacibdir.

Tətbiq əsaslı proksilər fərdi proqramlar daxilində ya daxili parametrlər vasitəsilə, ya da üçüncü tərəf proqram təminatından istifadə etməklə konfiqurasiya edilir. Cihazdakı bütün tətbiqlərə tətbiq olunan geniş sistemli proksilərdən fərqli olaraq, proqram əsaslı proksilər yalnız göstərilən tətbiqin internet trafikinə təsir göstərir. Bu seçmə yanaşma istifadəçilərə internet trafikini idarə etmək üçün daha çevik və fərdiləşdirilə bilən həll təqdim edərək, hər bir proqram üçün müxtəlif proksi parametrləri tətbiq etməyə imkan verir.

Tətbiq əsaslı proksilərin əsas üstünlüklərindən biri müxtəlif tətbiqlər üçün uyğunlaşdırılmış təhlükəsizlik siyasətlərini tətbiq etmək bacarığıdır. Məsələn, bir təşkilat təhlükəsizlik təhdidlərinə daha çox meylli olan bəzi proqramlar üçün ciddi məzmun filtrasiyası və monitorinqi tətbiq etmək istəyə bilər, eyni zamanda onlarla əlaqəli daha az riski olan digər proqramlar üçün məhdudiyyətsiz girişə icazə verə bilər. Bu, istifadəçinin məhsuldarlığına mane olmadan lazımi qorunmaların olmasını təmin edərək təhlükəsizlik və istifadə imkanları arasında tarazlıq yaratmağa kömək edə bilər.

Tətbiq əsaslı proksilərin digər əhəmiyyətli üstünlüyü onların geo məhdudiyyətləri keçmək və regiona aid məzmuna daxil olmaq qabiliyyətidir. Müəyyən bir tətbiqin trafikini başqa ölkədə yerləşən proxy server vasitəsilə yönləndirməklə, istifadəçilər öz regionlarında başqa cür əlçatmaz ola biləcək məzmuna daxil ola bilərlər. Bu, xüsusilə axın xidmətləri, xəbər saytları və ya istifadəçinin yerləşdiyi yerə görə fərqli məzmun təklif edən digər platformalar üçün faydalı ola bilər. Bundan əlavə, proqram əsaslı proksilər istifadəçilərə performansı optimallaşdırmağa və gecikməni minimuma

endirməyə kömək edə biləcək digər tətbiqlər üçün birbaşa əlaqə saxlamağa imkan verir.

Tətbiq əsaslı proksilər həmçinin müəyyən edilmiş proqram üçün istifadəçinin IP ünvanını maskalamaqla məxfiliyin qorunması səviyyəsini təklif edir. Trafikin proksi server vasitəsilə yönləndirilməsi ilə istifadəçinin həqiqi yeri və şəxsiyyəti üçüncü şəxslərdən gizlənə bilər ki, bu da izləmə və ya müşahidə riskini azaldır. Bu, kommunikasiya alətləri və ya sosial media platformaları kimi müəyyən proqramlardan istifadə edərkən anonimliyi qorumaq istəyən istifadəçilər üçün xüsusilə faydalı ola bilər.

Tətbiq əsaslı proksilərin çoxsaylı üstünlüklərinə baxmayaraq, nəzərə alınmalı potensial çatışmazlıqlar və problemlər var. Əsas narahatlıqlardan biri çoxsaylı proqramlar üçün proksi parametrlərini idarə etməyin mürəkkəbliyidir. Bu, vaxt apara bilər və konfiqurasiya xətalarına və ya təhlükəsizlik siyasətlərində uyğunsuzluqlara səbəb ola bilər. Bundan əlavə, bütün proqramlar proksilərin istifadəsini dəstəkləmir və ya onların məhdud proksi funksionallığı ola bilər ki, bu da bütün proqramlarda ardıcıl proxy siyasətinin həyata keçirilməsini çətinləşdirir.

Bundan əlavə, proqrama əsaslanan proksilərdən istifadə, proksi serverin özü sıxlaşarsa və ya fasilələrlə üzləşərsə, performans çətinlikləri yarada bilər. Bu, proksidən istifadə edən xüsusi tətbiq üçün istifadəçi təcrübəsinə təsir göstərərək xidmətdə potensial gecikmələrə və ya pozulmalara səbəb ola bilər. Bu problemi həll etmək üçün istifadəçilər trafiki paylamaq və performans problemləri riskini minimuma endirmək üçün çoxsaylı proksi serverlərdən və ya yük balanslaşdırma üsullarından istifadə etməyi düşünə bilərlər.

Nəticə etibarilə, proqram əsaslı proksilər hər bir proqram əsasında internet trafikini idarə etmək üçün yüksək dərəcədə fərdiləşdirilə bilən və çevik həll yolu təqdim edir. Onların uyğunlaşdırılmış təhlükəsizlik siyasətlərini tətbiq etmək, geo məhdudiyyətləri keçmək və məxfiliyin qorunmasını təklif etmək qabiliyyəti onları müasir şəbəkə konfiqurasiyalarında dəyərli alətə çevirir. Bununla belə, istifadəçilər çoxsaylı proqramlar üçün proksi parametrlərinin idarə edilməsinin mürəkkəbliyini, həmçinin potensial performans darboğazlarını və uyğunluq problemlərini diqqətlə nəzərdən

keçirməlidirlər. Tətbiq əsaslı proksilərin incəliklərini başa düşməklə, istifadəçilər öz şəbəkə konfiqurasiyaları haqqında məntiqli qərarlar qəbul edə və onların əlaqə və təhlükəsizlik ehtiyaclarının effektiv şəkildə qarşılanmasını təmin edə bilərlər.
Həm tətbiq əsaslı (tətbiq səviyyəli), həm də sistem əsaslı (sistem əasaslı) proksilərin özünəməxsus üstünlükləri və çatışmazlıqları var. Hər birinin müsbət və mənfi tərəflərini başa düşmək istifadəçilərə şəbəkə konfiqurasiyaları haqqında məlumatlı qərarlar qəbul etməyə və təhlükəsizlik, məxfilik və performansı optimallaşdırmağa kömək edə bilər.

Tətbiq və sistem səviyyəli proksilərin mənfi cəhətləri
Tətbiq əsaslı proksilər və sistem əsaslı proksilər internet trafikini, təhlükəsizlik və məxfiliyi idarə etmək üçün iki fərqli yanaşmadır. Hər bir metodun müəyyən bir istifadəçi və ya təşkilat üçün ən uyğun həllini təyin edərkən diqqətlə nəzərə alınmalı olan öz üstünlükləri və çatışmazlıqları var. Hər birinin müsbət və mənfi tərəflərini daha ətraflı araşdırmaqla, istifadəçilər hər bir yanaşma ilə bağlı təsirləri və potensial faydaları daha yaxşı başa düşə bilərlər.

Tətbiq əsaslı proksilər və ya geniş tətbiqetmə proksiləri fərdi proqramlar üçün proksi parametrləri üzərində ətraflı nəzarətin faydasını təklif edir. Bu çeviklik səviyyəsi istifadəçilərə hər bir proqram əsasında internet trafikini və təhlükəsizlik siyasətlərini idarə etməyə imkan verir, xüsusi proqramların unikal tələblərə cavab verməsi və ya regiona aid məzmuna daxil olması üçün uyğunlaşdırılmasını təmin edir.
Bununla belə, tətbiqə əsaslanan proksilər tərəfindən təmin edilən artan nəzarət öz çətinlikləri ilə gəlir. Əsas çatışmazlıqlardan biri çoxsaylı proqramlar üçün proksi parametrlərinin idarə edilməsində mürəkkəblikdir. Bu, vaxt apara bilər və konfiqurasiya xətaları və ya təhlükəsizlik siyasətlərində uyğunsuzluq riski hər əlavə tətbiq ilə arta bilər. Bundan əlavə, bütün proqramlar proksilərin istifadəsini dəstəkləmir. Həm də onların məhdud proksi funksionallığı ola bilər ki, bu da bütün proqramlarda ardıcıl proksi siyasətinin həyata keçirilməsini çətinləşdirə bilər.

Sistem miqyasında və ya qlobal proksilər kimi də tanınan sistem əsaslı proksilər internet trafikini və təhlükəsizliyi idarə etmək üçün daha mərkəzləşdirilmiş yanaşma təmin edir. ƏS səviyyəsində işləmək və bütün şəbəkə əlaqələrini təyin edilmiş proksi server vasitəsilə yönləndirməklə, sistem əasalı proksiləri məzmunun filtrasiyası, zərərli proqramların skan edilməsi və məlumatların şifrələnməsi kimi təhlükəsizlik tədbirlərinin bütün tətbiqlərdə vahid şəkildə tətbiq olunmasını təmin edir. Təhlükəsizlik siyasətlərindəki bu ardıcıllıq cihazları müxtəlif təhlükələrdən qorumağa və proksi parametrlərinin konfiqurasiyası və saxlanması prosesini sadələşdirməyə kömək edir.

Bundan əlavə, sistem səviyyəli proksiləri bütün tətbiqlər üçün istifadəçinin IP ünvanını maskalamaqla, izləmə və ya nəzarət riskini azaltmaqla məxfiliyin qorunmasını gücləndirə bilər. Üstəlik, tez-tez daxil olan məzmunu keş etməklə sistem geniş proksiləri, xüsusən də birdən çox cihazın eyni şəbəkəyə qoşulduğu mühitlərdə şəbəkə performansını yaxşılaşdıra bilər.

Bu üstünlüklərə baxmayaraq, sistem səviyyəli proksilərin öz məhdudiyyətləri var. Bir diqqətəlayiq çatışmazlıq fərdi tətbiqlər üçün trafikə nəzarətdə azaldılmış konfiqurasiya qabiliyyətidir. Müəyyən proqramlar üçün xüsusi proksi parametrləri lazımdırsa, qlobal proksi konfiqurasiyası lazımi çevikliyi təklif etməyə bilər. Bundan əlavə sistem səviyyəli proksilər bəzi proqramlar ilə susmaya görə çalışan zaman problem yarada bilər. Belə hallarda istifadəçilər sistem səviyyəli proksisini müvəqqəti olaraq söndürməli və ya həmin xüsusi tətbiqlər üçün istisnaları konfiqurasiya etməlidirlər.

Nəticə olaraq, həm proqram əsaslı, həm də sistem əsaslı proksilər unikal üstünlüklər və problemlər təklif edir. Tətbiq əsaslı proksilər fərdi proqramlar üzərində daha çox çeviklik və nəzarət təmin edir, lakin bir çox tətbiqlərdə idarə etmək və saxlamaq daha çətin ola bilər. Sistem əsaslı proksilər ardıcıl təhlükəsizlik siyasətlərini və internet trafikinin mərkəzləşdirilmiş idarə edilməsini təmin edir, lakin fərdi proqramlar üçün eyni səviyyədə nəzarət olmaya bilər. Hansı yanaşmanın qəbul ediləcəyinə qərar verərkən istifadəçilər onların xüsusi tələbləri, istənilən nəzarət səviyyəsi, konfiqurasiya və texniki xidmətin asanlığı kimi amilləri nəzərə almalıdırlar..

2.4 Proksi serverdə veb sorğuların modifikasiyasının həyata keçirilməsi

Daha öncə də qeyd edildiyi kimi, proksi server, müştəri ilə server arasında yerləşən, müştəriyə proksi vasitəsilə serverə sorğu göndərməyə imkan verən vasitəçidir. Proksi serverdən istifadə etməklə müştərilər əlavə təhlükəsizlik, məxfilik və nəzarət qatını təmin edərək veb sorğularını dəyişdirə, süzgəcdən keçirə və ya idarə edə bilərlər. Dissertasiyanın bu hissəsi proksilərin veb sorğularını dəyişdirə biləcəyi müxtəlif yolları, onlardan istifadənin faydalarını və bəzi potensial çatışmazlıqları əhatə edəcək. Proksilərin veb sorğularını dəyişdirə bilməsinin bir yolu HTTP sorğusunun başlıqlarını dəyişdirməkdir. Başlıqlar müştərinin IP ünvanı, tələb olunan resurs və istifadəçi agenti kimi sorğu haqqında metadata təmin edir. Başlıqları dəyişdirməklə, proksi müştərinin həqiqi IP ünvanını gizlətmək, sorğunun başqa coğrafi yerdən gəldiyini göstərmək və ya hətta veb-sayt məhdudiyyətlərini keçmək üçün fərqli istifadəçi agentini təqlid etmək kimi müxtəlif məqsədlərə nail ola bilər. Bu, coğrafi cəhətdən məhdudlaşdırılmış məzmuna daxil olmaq, istifadəçi məxfiliyini qorumaq və ya brauzerin izini keçmək üçün faydalı ola bilər.

Proksilərin dəyişdirə biləcəyi veb sorğularının başqa bir aspekti məzmunun özüdür. Proksilər reklamlar, izləmə skriptləri və ya zərərli kod kimi veb məzmunun xüsusi elementlərini filtrləmək və ya bloklamaq üçün konfiqurasiya edilə bilər. Bu, istifadəçinin baxış təcrübəsini artıra, yükləmə sürətini yaxşılaşdıra və potensial təhlükəsizlik təhdidlərindən qoruya bilər. Bundan əlavə, proksilər yeni funksiyaları sınaqdan keçirərkən və ya sazlama problemlərini həll edərkən proqramçının əvvəlcədən hesablamadığı xüsusi konfiqurasiyaları da test etmək üçün istifadə edilə bilər. Burada proqramçının əvvəlcədən hesablamadığı konfiqurasiya dedikdə proqramın gözləmədiyi hər hansı bir əlavə məlumat (başlıq, protokol, böyük həcmli dəyişən və.s) nəzərdə tutulur

Proksilər həmçinin veb trafiki idarə etmək və optimallaşdırmaq üçün istifadə edilə bilər. Keşləmə, hədəf serverdəki yükü azaltmaq və məzmunun çatdırılmasını sürətləndirmək üçün proxy serverlər tərəfindən istifadə edilən ümumi bir texnikadır. Tez-tez daxil olan resursların nüsxələrini saxlamaqla, proksi bu resursları birbaşa müştərilərə xidmət edə bilər və bu, orijinal serverə təkrar sorğulara ehtiyacı azaldır.

Bu, şəbəkə sıxlığını azaltmağa, bant genişliyinə qənaət etməyə və istifadəçilər üçün cavab müddətini yaxşılaşdırmağa kömək edə bilər. Proksilər tərəfindən istifadə edilən başqa bir texnika olan yük balansı, heç bir serverin yüklənməməsini təmin etmək üçün daxil olan veb-trafikin bir çox server arasında paylanmasını nəzərdə tutur. Bu, xüsusilə yüksək trafik dövrlərində veb saytın performansını və etibarlılığını artıra bilər.

Bundan əlavə, xüsusi məqsədli proksilər təhlükəsizliyin və məxfiliyin artırılmasında mühüm rol oynaya bilər. Müştəri və proksi arasında veb trafiki şifrələməklə, dinləyicilərin həssas məlumatları ələ keçirməsinin qarşısını alan təhlükəsiz əlaqə yaradıla bilər. Bu, çox vaxt təhlükəsiz olmayan və hücumlara meylli olan ictimai Wi-Fi şəbəkələrindən istifadə edərkən xüsusilə vacibdir. Proksilər həmçinin şəbəkə administratorlarına istifadəçi etimadnamələri və ya IP ünvanları əsasında xüsusi vebsaytlara və ya məzmun növlərinə girişi məhdudlaşdırmağa imkan verən girişə nəzarət siyasətlərini həyata keçirmək üçün istifadə edilə bilər. Bu, xüsusilə təşkilati parametrlərdə təhlükəsiz və idarə olunan baxış mühitini qorumağa kömək edə bilər.

Veb sorğularını dəyişdirmək üçün proksilərdən istifadə etməyin bir çox üstünlüklərinə baxmayaraq, nəzərə alınmalı potensial çatışmazlıqlar var. Məsələn, üçüncü tərəfin proksi serverinə güvənmək yeni potensial uğursuzluq nöqtəsini təqdim edir və əgər proksi server təhlükə altına düşərsə və ya fasilələrlə üzləşərsə, istifadəçinin əlaqəsi pozula bilər. Bundan əlavə, proksidən istifadə bəzən vasitəçi server vasitəsilə trafikin marşrutlaşdırılmasının əlavə gecikməsi səbəbindən daha yavaş əlaqə sürəti ilə nəticələnə bilər. Nəhayət, etibarlı proksi provayder seçmək vacibdir, çünki zərərli proksilər məxfiliyinizə və təhlükəsizliyinizə əhəmiyyətli risklər yaradaraq veb trafikinizi ələ keçirə, daxil edə və ya hətta manipulyasiya edə bilər.

Nəticə olaraq, proksilər veb sorğularını dəyişdirmək üçün geniş imkanlar təklif edir, təkmilləşdirilmiş məxfilik, təhlükəsizlik və veb məzmununa nəzarət kimi üstünlükləri təmin edir. Bununla belə, istifadəçilər potensial çatışmazlıqları diqqətlə nəzərdən keçirməli və təhlükəsiz və etibarlı baxış təcrübəsini təmin etmək üçün nüfuzlu proksi provayder seçməlidirlər.

Nüfuz testi zamanı proksi təhlükəsizlik mütəxəssisləri üçün əvəzolunmaz alət ola bilər və hədəf sistemdəki potensial zəiflikləri müəyyən etmək üçün veb sorğularının

müxtəlif aspektlərini dəyişdirməyə imkan verir. Nüfuz testi zamanı proksidən istifadə etməklə edilə bilən bəzi ümumi modifikasiya növləri bunlardır:

- HTTP başlıqlarının dəyişdirilməsi: Başlıqlar müştərinin IP ünvanı, istifadəçi agenti və tələb olunan resurs kimi sorğu haqqında metadata təmin edir. Başlıqları dəyişdirməklə, nüfuz test cihazı müxtəlif ssenariləri simulyasiya edə, girişə nəzarət mexanizmlərini sınaqdan keçirə və ya təhlükəsizlik məhdudiyyətlərini keçə bilər. Ümumi dəyişikliklərə istifadəçi agentinin dəyişdirilməsi, IP ünvanlarının saxtalaşdırılması və xüsusi başlıq sahələrinin əlavə edilməsi və ya silinməsi daxildir.
- URL manipulyasiyası: Tələb olunan URL-nin dəyişdirilməsi hədəf tətbiqdə potensial zəiflikləri aşkar edə bilər. Nümunələrə yolun dəyişdirilməsi, zərərli sorğu parametrlərinin əlavə edilməsi və ya URL ilə kodlaşdırılmış məlumatların manipulyasiyası ilə kataloq keçidinin sınaqdan keçirilməsi daxildir.
- HTTP metodlarının dəyişdirilməsi: HTTP metodunun dəyişdirilməsi (məsələn, GET-dən POST, PUT və ya DELETE-ə) müxtəlif sorğu növlərinin düzgün idarə olunmamasını müəyyən etməyə və ya müəyyən təhlükəsizlik nəzarətlərini keçməyə kömək edə bilər.
- Parametrlərin dəyişdirilməsi: Forma sahələri və ya sorğu parametrləri kimi daxiletmə məlumatlarının manipulyasiyası SQL inyeksiyası, saytlar arası skript (XSS) və ya uzaqdan kod icrası kimi zəiflikləri aşkar etməyə kömək edə bilər. Tipik dəyişikliklərə zərərli kodların yeridilməsi, məlumat növlərinin dəyişdirilməsi və ya kodlaşdırılmış məlumatların manipulyasiyası daxildir.
- Əsas hissənin dəyişdirilməsi: POST və PUT sorğuları üçün sorğunun əsas hissəsi hədəf tətbiqin gözlənilməz və ya zərərli daxiletməni necə idarə etdiyini yoxlamaq üçün manipulyasiya edilə bilər. Bu, xüsusi simvolların yeridilməsi, JSON və ya XML məlumatlarının dəyişdirilməsi və ya fayl yükləmələrinin dəyişdirilməsini əhatə edə bilər.
- Çərəz manipulyasiyası: Çərəz dəyərlərinin dəyişdirilməsi sessiyanın idarə edilməsi, autentifikasiya və ya girişə nəzarət ilə bağlı potensial təhlükəsizlik problemlərini müəyyən etməyə kömək edə bilər. Ümumi dəyişikliklərə seans

nişanlarının dəyişdirilməsi, son istifadə tarixlərinin dəyişdirilməsi və ya zərərli məlumatların yeridilməsi daxildir.

- SSL/TLS yoxlaması: SSL/TLS ilə şifrələnmiş hər hansı bir traffiki proksidən keçirdikdə proksi server həmin şifrələməni açmaqla trafikin kontentini yoxlamaq və lazimi dəyişiklikləri etmək qabiliyyətinə malikdir.
- Cavab müdaxiləsi: Server cavablarının tutulması və dəyişdirilməsi xətaların idarə edilməsi, məlumatın açılması və ya etibarlı olmayan müştəri tərəfi emal ilə bağlı potensial problemləri müəyyən etməyə kömək edə bilər.

Proksi serverlərin məhsuldarlığa təsiri

Proksilər veb proqramların işinə həm müsbət, həm də mənfi təsir göstərə bilər. Onların təsiri proksinin növü, konfiqurasiyası, şəbəkə şərtləri və tətbiqin özünün təbiəti kimi müxtəlif amillərdən asılıdır. Gəlin gecikmə, keşləmə, yük balansı, təhlükəsizlik və məzmunun filtrasiyası daxil olmaqla, proksilərin performansa təsir edə biləcəyi müxtəlif yollara baxaq.

Birincisi, proksidən istifadənin proqram performansına ən aydın təsirlərindən biri gecikmənin tətbiqidir. Müştəri bir proxy server vasitəsilə sorğu göndərdikdə, hədəf serverə yönləndirilməzdən əvvəl sorğu proksi tərəfindən işlənməlidir. Bu əlavə addım istər-istəməz ünsiyyət prosesinə müəyyən dərəcədə gecikmə əlavə edir. Təqdim olunan gecikmənin faktiki miqdarı proksinin coğrafi yerindən, xüsusi proksi funksiyaları üçün tələb olunan emal xərclərindən və ümumi şəbəkə şərtlərindən asılıdır. Proksi server həm müştəridən, həm də hədəf serverdən uzaqdırsa və ya yüksək trafiə məruz qalırsa, gecikmə əhəmiyyətli ola bilər. Belə ki, nəticədə daha yavaş cavab vaxtları yaranır ki, bu da tətbiqin performansı azalır.

Digər tərəfdən, proksilər də həmçinin keşləmə vasitəsilə tətbiq işini yaxşılaşdıra bilər. Keşləmə proksi serverdə şəkillər, skriptlər və ya üslub cədvəlləri kimi tez-tez istifadə olunan resursların nüsxələrinin saxlanmasını nəzərdə tutur. Müştəri keşlənmiş resurs tələb etdikdə, proksi sorğunu hədəf serverə yönləndirmədən ona birbaşa xidmət göstərə bilər. Bu, keşlənmiş resurslar üçün cavab müddətini, həmçinin hədəf serverə yükü əhəmiyyətli dərəcədə azalda bilər. Keşləmənin performansa ümumi təsiri keşin yığılma

sürətindən asılıdırki bu da keşdən xidmət ala bilən sorğuların nisbəti deməkdir. Daha yüksək keşləmə dərəcəsi adətən daha yaxşı performans qazancı ilə nəticələnir.

Proksilərin tətbiq performansını artırmağın başqa bir yolu yük balansını asanlaşdırmaqdır. Yük balansı, heç bir serverin yüklənməməsini təmin etmək üçün daxil olan veb trafiki bir neçə server arasında paylamağı əhatə edir. Bu, yüksək trafik həcminə malik irimiqyaslı proqramlar üçün xüsusilə vacib ola bilər, çünki o, ardıcıl performansı qorumağa və server darboğazlarının qarşısını almağa kömək edir. Proksilər trafiki səmərəli şəkildə paylamaq və optimal server istifadəsini saxlamaq üçün round-robin, ən az bağlantılar və ya resurs əsaslı alqoritmlər kimi müxtəlif yük balanslaşdırma alqoritmlərini tətbiq edə bilər. İş yükünü serverlər arasında bərabər paylamaqla, yük balansı tətbiqin performansını, etibarlılığını və miqyasını yaxşılaşdıra bilər.

Proksilər həmçinin təhlükəsizlik təkmilləşdirmələri vasitəsilə proqram performansına dolayı təsir göstərə bilər. Məsələn, proksilər hədəf serverə çatmazdan əvvəl Paylanmış Xidmətdən imtina (DDoS) hücumları kimi zərərli trafiki süzgəcdən keçirə bilər. Bu təhlükələri azaldaraq, proksilər hətta əlverişsiz şəraitdə belə tətbiqin əlçatanlığını və performansını qorumağa kömək edə bilər. Bundan əlavə, proksilər SSL/TLS şifrələmə tapşırıqlarını hədəf serverdən yükləyə, tətbiqə aid tapşırıqların işlənməsi üçün qiymətli resursları yarada bilərlər. Bu, xüsusilə yüksək səviyyəli şifrələnmiş trafikə malik tətbiqlər üçün təkmilləşdirilmiş performansa səbəb ola bilər.

Nəhayət, proksilər məzmunu filtrləmək və ya dəyişdirməklə tətbiqin performansına təsir göstərə bilər. Məsələn, proksilər reklamlar, izləmə skriptləri və ya multimedia faylları kimi xüsusi elementləri bloklamaq və ya silmək üçün konfiqurasiya edilə bilər. Bununla, müştəri ilə server arasında ötürülən məlumatların ümumi miqdarı azalır ki, bu da daha sürətli yükləmə müddətinə və bant genişliyi istehlakına səbəb ola bilər. Bununla belə, məzmunun süzülməsi bəzi hallarda performansa mənfi təsir göstərə bilən əlavə emal xərcləri də təqdim edə bilər. Performansa ümumi təsir filtrləmə qaydalarının mürəkkəbliyindən və səmərəliliyindən və tətbiq trafikinin xüsusiyyətlərindən asılı olaraq dəyişir.

Mahiyyət etibarilə, proksilər veb proqramların işinə mürəkkəb və çoxşaxəli təsir göstərə bilər. Onlar əlavə gecikməyə səbəb ola bilsələr də, keşləmə, yük balansı, təhlükəsizlik və məzmunun filtrasiyası kimi performansı artıran funksiyaları da təklif edə bilərlər. Proksinin proqram performansına ümumi təsiri müxtəlif amillərdən, o cümlədən proxy-nin konfiqurasiyası, şəbəkə şərtləri və tətbiqin xüsusi tələblərindən asılı olacaq.

Proksi serverlərin məhsuldarlığını necə artırmaq olar

Müştərilər və serverlər arasında sürətli və səmərəli əlaqəni təmin etmək üçün proksi serverlərin işinin yaxşılaşdırılması vacibdir. Proksi serveri optimallaşdırmaqla siz gecikməni azalda, ötürmə qabiliyyətini artıra və ümumi istifadəçi təcrübəsini artıra bilərsiniz. Proksi serverlərin işini yaxşılaşdırmaq üçün bəzi strategiyalar bunlardır:

- Avadanlığın optimallaşdırılması: Proksi serverin CPU, yaddaş və şəbəkə bant genişliyi kimi kifayət qədər aparat resurslarına malik olmasını təmin etmək optimal performans üçün çox vacibdir. Proksi serverin resurs istifadəsinə mütəmadi olaraq nəzarət etmək və artan trafik yüklərinə uyğunlaşmaq və darboğazların qarşısını almaq üçün lazım olduqda aparatı təkmilləşdirmək lazımdır.
- Şəbəkə optimallaşdırması: Gecikməni minimuma endirmək üçün coğrafi baxımdan həm müştərilərə, həm də hədəf serverlərə yaxın olan proxy server yerini seçmək lazımdır. Bundan əlavə, aşağı gecikmə və yüksək ötürmə qabiliyyətini təmin etmək üçün proksi serverin yüksək sürətli və etibarlı şəbəkə infrastrukturuna qoşulduğundan əmin olmaq lazımdır.
- Keşləmə strategiyaları: Tez-tez əldə edilən resursları proxy serverdə saxlamaq üçün ağıllı keşləmə strategiyalarını həyata keçirilməlidir. Bu, mənbə serverinə təkrar sorğulara ehtiyacı azaldır və cavab müddətini əhəmiyyətli dərəcədə yaxşılaşdırır. Keşin effektiv və aktual qalmasını təmin etmək üçün keşləmə siyasətlərini mütəmadi olaraq qiymətləndirmək və yeniləmək lazımdır. Keş hit dərəcələrini artırmaq və resurs istifadəsini

optimallaşdırmaq üçün keş iyerarxiyası, keş bölməsi və ya keşin çıxarılması alqoritmlərindən istifadə edilməsi də xüsusi əhəmiyyətə malikdir.

- Yük balansı: Daxil olan trafiki çoxsaylı proxy serverlər və ya arxa serverlər arasında paylamaq üçün yük balanslaşdırma üsullarından istifadə edilməlidir. Bu, ardıcıl performansı qorumağa və server darboğazlarının qarşısını almağa kömək edə bilər. Tətbiqin xüsusi tələblərindən asılı olaraq round-robin, ən az əlaqə və ya resurs əsaslı alqoritmlər kimi müxtəlif yük balanslaşdırma alqoritmləri həyata keçirilməlidir.
- Qoşulmanın idarə edilməsi: Bağlantıların qurulması və bağlanması ilə bağlı əlavə xərcləri azaltmaq üçün canlı saxlama müddətləri və əlaqənin birləşdirilməsi kimi əlaqə idarəetmə parametrlərini optimallaşdırmaq lazımdır. Bağlantıları təkrar istifadə etməklə əlaqənin qurulması və sökülməsi ilə bağlı gecikməni minimuma endirərək daha sürətli cavab vaxtlarına nail olmaq mümkündür.
- Sıxılma: Proksi server və müştərilər həmçinin proksi serverlər və hədəf serverlər arasında ötürülən məlumatların ölçüsünü azaltmaq üçün sıxılma üsullarından istifadə edilməlidir. Daha kiçik məlumat yükləri daha sürətli ötürmə vaxtlarına və daha az bant genişliyinə səbəb ola bilər. Mətn əsaslı məzmun üçün gzip və ya Brotli kimi ümumi sıxılma alqoritmlərini tətbiq etmək və müştərilərin və serverlərin seçilmiş sıxılma üsullarını dəstəklədiyinə əmin olmaq vacib məqamlardandır.
- Məzmunun filtrasiyasının optimallaşdırılması: Əgər proksi server məzmunun filtrasiyasını həyata keçirirsə, filtrləmə qaydaları və alqoritmlərinin səmərəli və yaxşı optimallaşdırıldığına əmin olmaq lazımdır. Səmərəli filtrləmə əhəmiyyətli emal yükü təqdim edə və proksi serverin yavaş işləməsinə səbəb ola bilər. Optimal performansı qorumaq üçün filtrləmə qaydalarını mütəmadi olaraq nəzərdən keçirmək və yeniləmək lazımdır.
- SSL/TLS yüklənməsi: SSL/TLS şifrələmə və deşifrə tapşırıqlarını arxa serverlərdən proxy serverə və ya xüsusi SSL/TLS yükləmə cihazına keçirmək lazımdır. Bu, backend serverlərində resursları boşalda bilər ki, bu da onlara

proqrama aid tapşırıqların işlənməsinə diqqət yetirməyə və nəticədə performansı yaxşılaşdırmağa imkan verir.

Bu strategiyaları həyata keçirməklə proksi serverlərin işini əhəmiyyətli dərəcədə yaxşılaşdıra bilmək, nəticədə daha sürətli cavab vaxtları, artan ötürmə qabiliyyəti və təkmilləşdirilmiş istifadəçi təcrübəsi əldə etmək mümkündür.

2.5 SSL deşirləməsi və proksi serverlər

Proksi serverlərdə SSL şifrəsinin açılması şəbəkədə şifrələnmiş trafikin yoxlanılmasına və monitorinqinə imkan verən mühüm prosesdir. Şifrənin açılması prosesi təhlükəsizlik cihazlarına şifrələnmiş trafikin məzmununu yoxlamağa, potensial təhlükəsizlik təhdidlərini müəyyən etməyə və məlumat itkisinin qarşısının alınması (DLP) siyasətlərini tətbiq etməyə imkan verir. Bu hissədə biz beş əsas aspektə diqqət yetirərək, proxy serverlərdə SSL şifrəsinin açılması prosesinin incəliklərini araşdıracağıq. Beş əsas aspekt bunlardır: SSL/TLS əl sıxması, proxy server rolu, şifrənin açılması prosesi, yenidən şifrələmə və potensial məxfilik problemləri.

1. SSL/TLS əl sıxma prosesi: SSL (Təhlükəsiz Socket Layer) və ya TLS (Nəqliyyat Layeri Təhlükəsizliyi) əl sıxma prosesi müştəri ilə server arasında təhlükəsiz əlaqə yaratmaq üçün ilkin addımdır. Bu proses şifrləmə alqoritmlərinin müzakirəsini, açıq açarların mübadiləsini və ortaq məxfi açarın yaradılmasını nəzərdə tutur. Əl sıxma əlaqənin təhlükəsiz, şəxsi və orijinal olmasını təmin edir. Müştəri serverlə şifrələnmiş əlaqə yaratmaq istədikdə, digər məlumatlar arasında müştərinin SSL/TLS versiyasını, dəstəklənən şifrə dəstlərini və təsadüfi nömrəni özündə birləşdirən "ClientHello" mesajı göndərməklə əl sıxma prosesinə başlayır. Server "ServerHello" mesajı ilə cavab verir, ən uyğun şifrə dəstini seçir, onun sertifikatını və açıq açarını və başqa təsadüfi nömrəni təqdim edir.
2. Proksi server rolu: SSL deşifrəsi kontekstində proksi server müştəri və server arasında şifrələnmiş trafiki ələ keçirən və deşifrə edərək, ortada adam adlandırılan bir obyekt (MITM) kimi çıxış edir. Bunu həyata keçirmək üçün proksi server SSL/TLS trafikinin şifrəsini açmaq üçün konfiqurasiya edilməlidir və müştərilər proxy serverin sertifikat orqanına (CA) etibar etməlidirlər. Proksi

server şəxsi açarından istifadə edərək hər bir təyinat serveri üçün yeni sertifikat yaradır və imzalayır. Bu yolla, müştərilər şifrələnmiş rabitəni idarə etmək üçün proksiyə etibar edərək, proksinin sertifikatını orijinal kimi təsdiq edə bilər.

3. Şifrənin açılması prosesi: Proksi server SSL/TLS trafikini ələ keçirdikdən sonra təyinat serveri üçün yaratdığı sertifikatla əlaqəli şəxsi açardan istifadə edərək məlumatın şifrəsini açır. Bu, proksi serverə şifrələnmiş trafikin məzmununu yoxlamağa, təhlükəsizlik siyasətlərini tətbiq etməyə və potensial təhlükələri və ya məlumat sızmalarını aşkar etməyə imkan verir. Şifrənin açılması prosesi təşkilatlar üçün şəbəkə təhlükəsizliyini və uyğunluğu təmin etmək üçün çox vacibdir, çünki o, başqa şəkildə həssas məlumatları çıxarmaq və ya zərərli fəaliyyətləri gizlətmək üçün istifadə oluna bilən şifrələnmiş kommunikasiyaların görünməsini təmin edir.
4. Yenidən şifrələmə: Proksi server şifrəsi açılmış trafiki yoxladıqdan və təhlükəsizlik siyasətlərini tətbiq etdikdən sonra məlumatları nəzərdə tutulan təyinat serverinə yönləndirməzdən əvvəl onu yenidən şifrələyir. Proksi server verilənləri şifrləmək üçün təyinat serverinin açıq açarından istifadə edir və yalnız nəzərdə tutulan alıcının məlumatı deşifrə edə və məlumatı əldə edə bilməsini təmin edir. Bu təkrar şifrələmə prosesi müştəri ilə server arasında şifrələnmiş rabitənin məxfiliyini və bütövlüyünü qoruyur, eyni zamanda proksi serverə öz təhlükəsizlik funksiyalarını yerinə yetirməyə imkan verir.
5. Potensial məxfiliklə bağlı narahatlıqlar: Proksi serverlərdə SSL şifrəsinin açılması şəbəkə təhlükəsizliyini və uyğunluğu qorumaq üçün vacib olsa da, şifrənin açılması prosesi şifrələnmiş kommunikasiyaların ələ keçirilməsini və yoxlanılmasını nəzərdə tutduğundan məxfiliklə bağlı narahatlıqları da artıra bilər. Təşkilatlar SSL şifrəsinin açılmasını həyata keçirərkən təhlükəsizlik və məxfilik arasındakı tarazlığı diqqətlə nəzərdən keçirməli və şifrənin açılması prosesinin yalnız qanuni təhlükəsizlik məqsədləri üçün istifadə olunmasını təmin etmək üçün aydın siyasətlər yaratmalıdır. Bundan əlavə, təşkilatlar öz işçilərini SSL deşifrəsinin istifadəsi və potensial məxfiliyə təsirləri barədə məlumatlandırmalı və icazəsiz girişin və ya sui-istifadənin qarşısını almaq üçün

proxy serverin CA sertifikatının düzgün idarə olunmasını və təhlükəsizliyini təmin etməlidir.

SSL şifrəsinin açılmasının həyata keçirilməsi bir neçə addımı əhatə edir. Məsələn, proksi serverin konfiqurasiyası, sertifikat orqanının (CA) yaradılması və paylanması, müştərilərə CA sertifikatının quraşdırılması, tələb olunan qayda və siyasətlərin qurulması və təhlükəsizlik təhdidləri üçün şifrəsi açılmış trafikin monitorinqi. Aşağıda bu addımların hər biri, həmçinin SSL şifrəsinin açılmasını həyata keçirərkən məxfiliyin və təhlükəsizliyin qorunması üçün ən yaxşı təcrübələr müzakirə ediləcəkdir.

Proksi serverin konfiqurasiyası: SSL deşifrəsini həyata keçirmək üçün ilk addım proksi serveri SSL/TLS trafikinə müdaxilə etmək və deşifrə etmək üçün konfiqurasiya etməkdir. Bu, istifadə olunan xüsusi proksi server proqram təminatından asılı olaraq dəyişə bilən proxy server parametrlərində SSL şifrəsinin açılması funksiyalarını aktivləşdirməklə edilə bilər. Konfiqurasiya prosesi adətən monitorinq ediləcək şəbəkə interfeyslərinin və ya IP ünvanlarının təyin edilməsini, həmçinin şifrəsi açılacaq protokolların və şifrə dəstlərinin müəyyən edilməsini əhatə edir.

Sertifikat orqanının (CA) yaradılması və paylanması: Proksi server SSL/TLS əl sıxma prosesi zamanı hər bir təyinat serveri üçün sertifikatları imzalamaq üçün istifadə olunacaq öz-özünə imzalanmış CA sertifikatı və şəxsi açar yaratmalıdır. Bu CA sertifikatı şəbəkədəki müştərilər tərəfindən etibar edilməlidir, ona görə də CA sertifikatını bütün müvafiq müştərilərə paylamaq vacibdir. CA sertifikatı Windows Active Directory mühitində Qrup Siyasəti Obyektləri (GPO) və ya mobil cihazlar üçün Mobil Cihaz İdarəetmə (MDM) həlləri kimi müxtəlif üsullardan istifadə etməklə paylana bilər.

Müştərilərə CA sertifikatının quraşdırılması: Müştərilər və proksi server arasında etimadın yaradılması üçün, proksi server tərəfindən yaradılan CA sertifikatı hər bir müştəri cihazında quraşdırılmalıdır. Bu proses adətən CA sertifikatının əməliyyat sisteminin sertifikat mağazasına və ya veb-brauzerlər üçün brauzerin sertifikat mağazasına idxalını əhatə edir. Bu, müştərilərə proksi serverin sertifikatlarının həqiqi olduğunu təsdiq etməyə və şifrələnmiş kommunikasiyaları idarə etmək üçün proxy

serverə etibar etməyə imkan verir. CA sertifikatının bütün müvafiq cihazlarda təhlükəsiz şəkildə quraşdırılmasını təmin etmək vacibdir, çünki pozulmuş CA sertifikatı təhlükəsizlik risklərinə səbəb ola bilər.

Qaydaların və siyasətlərin qurulması: SSL şifrəsinin açılması üçün konfiqurasiya edilmiş proksi server və müştəri cihazlarında quraşdırılmış CA sertifikatı ilə növbəti addım şifrənin açılması prosesini tənzimləyən qayda və siyasətləri müəyyən etməkdir. Bu, adətən mənbə və təyinat IP ünvanları, domen adları və ya tətbiq protokolları kimi faktorlara əsaslanaraq hansı trafik növlərinin deşifrə edilib-edilməməsini nəzərdə tutur. Əlavə olaraq, təşkilatlar deşifrə edilmiş trafikdə potensial təhlükəsizlik təhdidləri və ya məlumat sızması aşkar edildikdə hansı tədbirlərin görülməli olduğunu müəyyən etmək üçün siyasətlər yaratmalıdır. Bu, trafiki bloklamaq, hadisəni qeyd etmək və ya inzibatçıya bildiriş vermək kimi hərəkətləri əhatə edə bilər.

Şifrədən çıxarılan trafikin monitorinqi: SSL şifrəsinin açılması həyata keçirildikdən və müvafiq qaydalar və siyasətlər mövcud olduqdan sonra, proksi server şifrələnmiş trafikin şifrəsini açmağa və yoxlamağa başlaya bilər. Təhlükəsizlik administratorları zərərli fəaliyyət əlamətləri, məlumatların çıxarılması və ya digər təhlükəsizlik təhdidləri üçün şifrəsi açılmış trafikə nəzarət etməlidir. Bu, müdaxilənin aşkarlanması sistemləri (IDS), məlumat itkisinin qarşısının alınması (DLP) həlləri və ya təhlükəsizlik məlumatı və hadisələrin idarə edilməsi (SIEM) platformaları kimi müxtəlif təhlükəsizlik vasitələrindən istifadə etməklə edilə bilər. Şifrədən çıxarılan trafikin monitorinqi şəbəkə təhlükəsizliyini qorumaq və müvafiq qaydalara və siyasətlərə uyğunluğu təmin etmək üçün çox vacibdir.

Məxfilik və təhlükəsizlik mülahizələri: SSL deşifrəsinin tətbiqi məxfiliklə bağlı narahatlıqları artıra bilər, çünki o, şifrələnmiş kommunikasiyaların tutulmasını və yoxlanılmasını nəzərdə tutur. Təşkilatlar SSL deşifrəsini həyata keçirərkən təhlükəsizlik və məxfilik arasındakı tarazlığı diqqətlə nəzərdən keçirməli və şifrənin açılması prosesinin yalnız qanuni təhlükəsizlik məqsədləri üçün istifadə olunmasını təmin etmək üçün aydın siyasətlər qurmalıdırlar. Bundan əlavə, icazəsiz girişin və ya sui-istifadənin qarşısını almaq üçün proksi serveri və onun CA sertifikatını qorumaq vacibdir. Proksi server proqramının müntəzəm olaraq yenilənməsi, təhlükəsizlik

zəifliklərinin monitorinqi və güclü giriş nəzarətinin həyata keçirilməsi SSL şifrəsinin açılması prosesinin təhlükəsizliyini və məxfiliyini qorumağa kömək edə bilər.

Bu addımlara və ən yaxşı təcrübələrə əməl etməklə, təşkilatlar şifrələnmiş trafikdə görünürlük əldə etmək, potensial təhlükəsizlik təhdidlərini aşkar etmək və məlumat itkisinin qarşısını almaq üçün SSL deşifrəsini uğurla həyata keçirə bilər.

III FƏSİL. PROKSİ SERVERİN YAZILMASI

3.1 HTTP sorğuların qəbul edilməsi.

Python-da HTTP sorğularını qəbul etmək veb proqramlaşdırmanın əsas aspektlərindən biridir, çünki bu, müştəri sorğularına cavab verə bilən server proqramların yaradılmasına imkan verir. Python bu prosesi asanlaşdırmaq üçün geniş çeşiddə kitabxanalar və mühitlər təklif edir, onlardan bəziləri Flask, Django və FastAPI-dir. Bu mühitlər HTTP sorğuları və cavabları ilə məşğul olmağın əsas mürəkkəbliklərini aradan qaldırır, tərtibatçıların miqyaslana bilən və möhkəm veb proqramları qurmasını asanlaşdırır. Çərçivənin seçimi layihənin xüsusi tələblərindən, həmçinin tərtibatçının tanışlığından və üstünlüklərindən asılı olaraq dəyişir. Flask çox vaxt sadəliyi və istifadə asanlığı ilə seçilir, Django isə daxili xüsusiyyətləri və güclü icma dəstəyi ilə tanınır. FastAPI, asinxron proqramlaşdırma və API sənədləri üçün daxili dəstəyi ilə müasir yanaşma təklif edən ekosistemə daha yeni əlavələrdən biridir.

Seçilmiş proqramlaşdırma mühitindən asılı olmayaraq, Python-da HTTP sorğularının qəbulu prosesi bir neçə əsas addımı əhatə edir. Birincisi, tərtibatçı proqramın cavab verəcəyi marşrutları və ya son nöqtələri müəyyən edən server tərəfi proqram yaradılmalıdır. Bu marşrutlar, daxil olan sorğuların işlənməsi və müvafiq cavabların yaradılması üçün məntiqi müəyyən edən işləyicilər və ya görünüşlər kimi tanınan xüsusi funksiyalara uyğunlaşdırılır. Müştəri serverə HTTP sorğusu göndərdikdə, proqram sorğunu müvafiq marşruta uyğunlaşdırır və əlaqəli prosesləri yerinə yetirir. Bu proseslər sorğunu emal edir, istəyə bağlı olaraq sorğunun gövdəsindən, başlıqlardan və ya URL parametrlərindən verilənlərə daxil olur və sonra data, status kodları və başlıqları daxil edə bilən HTTP cavabını qaytarır. Cavab, onu tətbiq məntiqinə uyğun olaraq emal edən müştəriyə geri göndərilir.

Python-da HTTP sorğularını qəbul edərkən xətaların idarə edilməsi və təhlükəsizliyin nəzərə alınması da vacib məsələlərdəndir. SQL inyeksiyası və ya saytlar arası skript (XSS) hücumları kimi təhlükəsizlik zəifliklərinin qarşısını almaq üçün istifadəçi daxiletməsini yoxlamaq və təmizləmək çox vacibdir. Bu, seçilmiş çərçivə tərəfindən təmin edilmiş daxili yoxlama mexanizmlərindən istifadə etməklə və ya Marshmallow

və ya Pydantic kimi üçüncü tərəf kitabxanalarından istifadə etməklə əldə edilə bilər. Bundan əlavə, gözlənilməz problemlər yarandıqda belə tətbiqin işləməyə davam etməsini təmin etmək üçün istisnaları və səhvləri zərif şəkildə idarə etmək vacibdir. Bu, istisnaları tutmaq və idarə etmək üçün bloklardan, fərdi səhv idarəedicilərindən və ya ara proqramdan istifadə etmək lazımdır. HTTP sorğularını idarə etmək üçün ən yaxşı təcrübələri qəbul etməklə tərtibatçılar Python-da istifadəçilərinin ehtiyaclarına cavab verən təhlükəsiz, etibarlı və səmərəli veb proqramlar yarada bilərlər.

Şəkil 3.1 və 3.2 - də python proqramlaşdırma dilində veb sorğuların qəbul edilə bilməsi üçün yazılan proqram göstərilmişdir.

```python
import http.server
import socketserver
import logging

# Set up logging
logging.basicConfig(filename='http_requests.log', level=logging.INFO, format='%(asctime)s - %(message)s')

class CustomRequestHandler(http.server.SimpleHTTPRequestHandler):
    def do_GET(self):
        self.log_full_request()
        self.send_response(200)
        self.end_headers()
        self.wfile.write(b'HTTP request logged successfully.')

    def do_POST(self):
        self.log_full_request()
        self.send_response(200)
        self.end_headers()
        self.wfile.write(b'HTTP request logged successfully.')

    def log_full_request(self):
        logging.info(f'{self.command} {self.path} HTTP/{self.request_version}')
        for header, value in self.headers.items():
            logging.info(f'{header}: {value}')

        content_length = int(self.headers.get('Content-Length', 0))
        if content_length > 0:
            payload = self.rfile.read(content_length)
            logging.info(f'Payload: {payload.decode()}')

        logging.info('-----')
```

Şəkil 3.1. Veb sorğuların qəbul edilməsi üçün proqram

```python
# Set up the server
PORT = 8080
Handler = CustomRequestHandler
httpd = socketserver.TCPServer(("", PORT), Handler)

print(f"Serving on port {PORT}")
httpd.serve_forever()
```

Şəkil 3.2. Veb serverin konfiqurasiya edilmə kodu

Bu Python skripti *http.server* və *socketserver* kitabxanalarından istifadə edərək əsas HTTP serverini qurur. O, daxil olan *HTTP GET* və *POST* sorğularını *http_requests.log* adlı fayla qeyd edir. Skript *http.server.SimpleHTTPRequestHandler* alt təsnifatı və *do_GET, do_POST* və *log_full_request* üsullarını tətbiq etməklə fərdi sorğu işləyicisi yaradır. Server 8080 portunu dinləmək üçün konfiqurasiya edilmişdir.

1. Lazımi kitabxanaların daxil edilməsi
    - *http.server*: HTTP serverinin (Veb server) icra edilə bilməsi üçün siniflər təqdim edən moduldur.
    - *socketserver*: Şəbəkə serverləri üçün mühit təmin edən və server proqramlarının yaradılmasını asanlaşdıran moduldur.
    - *logging*: Tətbiqlər üçün çevik hadisə qeyd sistemini təmin etmək üçün moduldur.
2. Loqlama mexanizminin konfiqurasiya edilməsi
    - *logging.basicConfig(...):* Giriş səviyyəsi, jurnal faylının adı və jurnal mesajı formatı kimi əsas qeyd parametrlərini konfiqurasiya edir.
3. Xüsusi sorğu işləyicisi olan *CustomRequestHandler* sinfini müəyyənləşdirilməsi:
    - Bu sinif *http.server.SimpleHTTPRequestHandler*-dən yaranan sinif olmaqla *http.server* modulu tərəfindən təmin edilən əsas HTTP sorğularını idarə edir.
4. *do_GET* metodunu yaradılması:
    - Bu üsul server HTTP GET sorğusu yarada bilmək üçün çağırılır.
    - O, sorğu təfərrüatlarını qeyd etmək üçün *log_full_request* metodunu çağırır, 200 OK cavabı göndərir və müştəriyə yekun mesajı yazır.
5. *do_POST* metodunun yaradılması:
    - Bu üsul server HTTP POST sorğusu qəbul etdikdə çağırılır.
    - *do_GET* kimi, sorğu təfərrüatlarını qeyd edir, 200 OK cavabı göndərir və müştəriyə yekun mesajı yazır.
6. *log_full_request* metodunun yaradılması:
    - Bu üsul sorğunun əmrini, yolunu və HTTP versiyasını qeyd edir.

- Metod, sorğunun başlıqları üzərində təkrarlanır və onları qeyd edir.
- Əgər sorğunun əsas hissəsi varsa (POST sorğusu olduğu halda), o, deşifrədən sonra əsas hissəni qeyd edir.
- Sorğuları ayırd etməyi asanlaşdırmaq üçün log faylına ayırıcı xətt əlavə edir.

7. Serverin konfiqurasiya edilməsi
    - Serverin qulaq asmağı üçün port nömrəsini (8080) qeyd olunur.
    - Göstərilən port və fərdi sorğu işləyicisi ilə *socketserver.TCPServer* nümunəsini yaradılır. Bu, daxil olan əlaqələri dinləyən və *CustomRequestHandler* sinfindən istifadə edərək onları idarə edən TCP serveri yaradır.
    - Serverin göstərilən portu dinlədiyini göstərmək üçün mesaj çap edilir.
    - Serveri işə salan və gələn sorğuları idarə etmək üçün onu qeyri-müəyyən müddətə işlək vəziyyətdə saxlayan server nümunəsində *serve_forever* metodu çağırılır.

3.2 HTTP srogularin modifikasiya edilməsi

Aşağıdakı Python skripti HTTP POST sorğuları üçün yazılmış proqramın əsas hissələridir. O, fayldan HTTP məlumatını oxuyur, məlumatın müəyyən hissəsini söz siyahısındakı sözlərlə əvəz edir və dəyişdirilmiş sorğuları hədəf URL-ə göndərir. Skript həmçinin başlıqları və sorğunun gövdəsindəki POST məlumatlarını dəyişdirmək imkanına malikdir. Yazılmış proqram aşağıda şəkillərdə göstərildiyi kimi ayrı-ayrı əsas hissələrə ayrılmaqla izah ediləcəkdir.

1.Lazımi kitabxanaların daxil edilməsi

```
1   import re
2   import requests
3
```

Şəkil 3.3. Lazımi kitabxanaların daxil edilməsi

Şəkil 3.3-də göstərildiyi kimi, skript əsas proqrama iki modulu daxil edir: müntəzəm ifadələr üçün *re* modulu və HTTP sorğularını idarə etmək üçün *requests* modulu.

2. Ümumi dəyişənlərin təyin edilməsi

Bunlar skript icra olunduğu müddətdə istifadə olunan qlobal dəyişənlərdir. Onlara HTTP başlıqlarını və dəyişdiriləcək sətri yadda saxlamaq üçün müntəzəm ifadələr, hədəf URL, gövdə və söz siyahısı üçün fayl adları, başlıqlar, POST məlumatları üçün lüğətlər və sorğu xətti üçün siyahı daxildir. Dəyişənlərin nə cür təyin olunmalı olduğu şəkil 3.4-də göstərilmişdir.

```
1  http_header_regex = '([\w-]+): '
2  string_to_change_regex = '(\$.*?\$)'
3  url = 'https://0a3500960335e87ec07ca5ae004a00ae.web-security-academy.net/login2'
4  payload_file = 'payload.txt'
5  wordlist_file = 'wordlist.txt'
6  headers_and_its_values = {}
7  parsed_data = {}
8  request_line = []
9
```

Şəkil 3.4. Ümumi dəyişənlər

3.Get_first_line() funksiyası

```
1  def get_first_line():
2      ...
3
```

Şəkil 3.5. Gövdənin birinci sətrini oxumaq üçün funksiya

Şəkil 3.5-də göstərilən *get_first_line()* funksiyası HTTP sorğusunun gövdəsinin birinci sətrini oxuyur və onu qlobal dəyişən olan *request_line*-a mənimsədir

4.add_headers_to_dictionary() funksiyası:

```
1  def add_headers_to_dictionary():
2      ...
3
```

Şəkil 3.6. Başlıqları lüğətə əlavə etmək üçün funksiya

Şəkil 3.6-da göstərilən funksiya protokol faylını sətir-sətir oxuyub, düzgün ifadələrdən istifadə edərək başlıq məlumatını çıxarıb *headers_and_its_values* lüğətinə açar-dəyər cütləri şəklində əlavə edir.

5.parse_post_data_to_dictionary() funksiyası:

```
1   def parse_post_data_to_dictionary(data):
2       ...
3
```

Şəkil 3.7. Məlumatı lüğətə əlavə edən funksiya.

Şəkil 3.7-də göstərilən funksiya POST məlumatını ehtiva edən sətiri götürür, onu açar-dəyər cütlərinə bölür və təhlil edilmiş verilənlərlə lüğəti qaytarır.

6.get_post_data_from_file() function:

```
1   def get_post_data_from_file():
2       ...
3
```

Şəkil 3.8. POST datanın oxunması

Şəkil 3.8-də göstərilən funksiya protokolun gövdəsi olan faylı oxuyur və *Content-Length* başlıq dəyərinə əsaslanaraq POST məlumatlarını çıxarır. Daha sonra, çıxarılan POST məlumatını sətir kimi qaytarır.

7. find_character_count(), find_change_line(), and change_request_payload() funksiyaları:

```
1   def find_character_count(string):
2       ...
3
4   def find_change_line():
5       ...
6
7   def change_request_payload(change_line, word, original_value_of_line):
8       ...
9
```

Şəkil 3.9. Modifikasiyadan məsul funksiyalar.

Şəkil 3.9-da modifikasiyadan məsul olan funksiyalar göstərilmişdir. Bu funksiyalar sorğunun söz siyahısındakı sözlərlə dəyişdirilməli olan hissəsini tapmaq və

dəyişdirmək üçün istifadə olunur. *find_character_count()* sətirdəki '$' işarələrinin sayını hesablayır. *find_change_line()* dəyişdiriləcək xəttin növünü (başlıq, məlumat və ya birinci sətir) müəyyən edir və həmin məlumatı qaytarır. *change_request_payload()* verilmiş sözə və xəttin orijinal dəyərinə əsasən sorğu xəttini, başlığı və ya POST məlumatını dəyişdirir.

8.bruteforce() function.

```
1   def bruteforce():
2       ...
3
```

Şəkil 3.10. bruteforce() funksiyası

Şəkil 3.10-da göstərilən funksiya söz siyahısı faylındakı sözlər arasında təkrarlanır və tələbi dəyişdirmək üçün *change_request_payload()* funksiyasını çağırır. Daha sonra sorğular modulundan istifadə edərək dəyişdirilmiş sorğunu hədəf URL-ə göndərir və cavab statusunu və cari sözü söz siyahısını çap edir.

9.main() funksiyası və proqramın icra edilməsi.

```
1   def main():
2       ...
3
4   if __name__ == '__main__':
5       main()
6
```

Şəkil 3.11. Proqramın icra edilməsi

Şəkil 3.11-də göstərilən main() funksiyası faydalı yük faylını oxumaq, başlıqları və POST məlumatlarını təhlil etmək və bruteforce hücumunu həyata keçirmək üçün skriptdəki digər funksiyaları çağırır. Skript nəticə olaraq main() funksiyasını çağırmaqla icra olunur.

3.3 Yazılmış proqram təminatının mövcud veb proksilər ilə müqayisəsi

Hal hazırda kibertəhlükəsizlik mütəxəssisləri tərəfindən istifadə edilən ən geniş yayılmış proksi server “Burp Suite” – dir. Burp Suite veb proqramlarının təhlükəsizliyinin yoxlanılması üçün güclü və geniş istifadə olunan bir vasitədir. Bu dissertasiya işində təqdim edilən proqram təminatı Burp Suit proqram təminatının aşağıdakı mənfi cəhətləri nəzərə alınaraq yazılmışdır:

1) İstifadənin maddi cəhətdən hər kibertəhlükəsizlik mütəxəssisi üçün əlverişli olmaması. Burp Suite-nin əsas çatışmazlıqlarından biri onun qiymətidir. Məhdud xüsusiyyətləri olan pulsuz *İcma Nəşri* təklif etsə də, tam xüsusiyyətli versiyası *Peşəkar Nəşr* lisenziya tələb edir ki, bu da fərdi istifadəçilər və ya büdcə məhdudiyyətləri olan kiçik təşkilatlar üçün olduqca bahalı ola bilər. Digər tərəfdən, bu dissertasiya işində yazılam proksi açıq mənbəli olduğundan istənilən şəxs və ya təşkilat tərəfindən istifadə edilə bilər.
2) İcma nəşrində məhdud funksionallıq. Burp Suite-nin pulsuz İcma Nəşrində Peşəkar Buraxılışda mövcud olan bəzi təkmilləşdirilmiş funksiyalar yoxdur. Məsələn, bu versiyada sorğu göndərmək sürəti olduqca yavaşdır. Belə ki, bizim proqram təminatı ilə müqayisə etdikdə 1 dəqiqədə göndərilən sorğular arasında situasiyadan asılı olaraq 2-3 dəfə fərq yaranır. Əksinə, təqdim edilən proqram isə heç bir məhdudiyyət olmadan geniş funksiyalar təqdim edir.
3) Kommersiya dəstəyinə etibar: Burp Suite-in kommersiya xarakteri istifadəçilərin alətin arxasında duran şirkət PortSwigger tərəfindən təmin edilən dəstək və yeniləmələrə etibar etməsi deməkdir. Burp Suite peşəkar dəstək təklif etsə də, istifadəçilər şirkətin cavab müddətindən və mövcudluğundan asılıdır. Digər tərəfdən, açıq mənbəli veb proksilər cəmiyyətin birgə səylərindən faydalanır ki, bu da tez-tez daha təcili yardım və davamlı inkişaf təmin edə bilər.
4) Qapalı mənbə kodu: Burp Suite qapalı mənbəli proqramdır, yəni istifadəçilərin əsas kod bazasına girişi yoxdur. Bu şəffaflığın olmaması açıq mənbə alətlərinə üstünlük verən bəzi təhlükəsizlik şüurlu istifadəçiləri narahat edə bilər, çünki bu, potensial təhlükəsizlik zəiflikləri üçün aləti yoxlamaq imkanlarını məhdudlaşdırır.

5) Fərdiləşdirmə məhdudiyyətləri: Burp Suite fərdiləşdirmə üçün genişləndirmə çərçivəsini təmin etsə də, alətin funksiyalarını dəyişdirmək və genişləndirmək imkanı açıq mənbəli veb proksi-serverlə müqayisədə hələ də bir qədər məhduddur. Təqdim olunan proksinin açıq mənbə xarakteri istifadəçilərə öz plaginlərini inkişaf etdirməyə və alətin ekosisteminə töhfə verməyə imkan verir, fərdiləşdirmə üçün daha çox çeviklik təmin edir.

Yuxarıda göstərilənlər təkcə “Burp Suite” proqram təminatı üçün deyil, ümumilikdə istənilən kommersiya məqsədli proqram təminatı üçün doğrudur.

NƏTİCƏ

Magistr dissertasiya işi əlaqədar texnologiyalar, onların iş prinsipləri, tətbiq sahələri araşdırılmaqla başlanmış, sonda isə bu texnologiyaları praktiki olaraq tətbiq edə biləcəyimiz proqram hazırlamaqla aşağıdakı nəticələrlə yekunlaşmışdır:

- Veb texnologiyaların ümumi arxitekturası, arxitekturanın ayrı-ayrı komponentləri analiz edilmişdir;
- HTTP protokolunun ilkin və hazırki versiyalarının müqayisəli şəkildə analizi aparılmışdır;
- Nüfuzetmə testlərində veb sorğuların rolu, onların modifikasiyasının vacibliyi təhlil edilmişdir;
- Proksi serverlərin arxitekturasını təhlil edilmişdir;
- Proksi serverlərin müxtəlif tip veb sorğulara hansı şəkildə cavab verdiyi ilə bağlı eksperimentlər aparılmışdır;
- Veb sorğuları qəbul etmək üçün proqram təminatı yaradılmışdır;
- Veb sorğuları modifikasiya edərək eyni sorğunu təkrarlaya bilən proqram təminatı yaradılmışdır;
- Yaradılan proqram təminatı müxtəlif situasiyalarda test edilmiş, effektivliyi araşdırılmışdır.

ƏDƏBİYYAT

1. https://github.com/orkhan-alibayli/Proxy-for-editing-http-requests
2. Alibayli O, Principles of Cyber Security / İnternational Eco Architecture and Design Perspective Congress, 2023
3. Agileera V, Cerullo F, Serrao C. Web Application Security. 2010
4. Andrew H, Michael S. Practical Malware Analysis. 2012
5. Authentication Vulnerabilities. https://portswigger.net/web-security/authentication
6. Barry P. Head First Python, 2nd Edition. 2016
7. Borso S. The Penetration Tester's Guide to Web Applications. 2019
8. Cole E. Network Security Bible. 2009
9. Dafydd S, Marcus P. The Web Application Hacker's handbook. 2011
10. Davies J. Implementing SSL/TLS using Cryptography and PKI. 2011
11. Friedman A, Singer P.W. Cybersecurity and Cyberwar: What Everyone Needs to Know. 2014
12. GET and POST requests using Python. https://www.geeksforgeeks.org/get-post-requests-using-python/ . 2023
13. Goerzen J, Bower T, Rhodes B. Foundations of Python Network Programming. 2011
14. Hoing A, Skiorski M. Practical Malware Analysis. 2012
15. HTTP header. https://developer.mozilla.org/en-US/docs/Glossary/HTTP_header
16. HTTP modules. https://docs.python.org/3/library/http.html
17. HTTP servers. https://docs.python.org/3/library/http.server.html .
18. Jain V.K. Cryptography and Network Security. 2019
19. John M. Zelle. An Introduction to Computer Science. 2004
20. Kevin M. The Art of Deception: Controlling the Human Element of Security. 2002

21. Khawaja G. Practical Web Penetration Testing: Secure Web Applications Using Burp Suite, Nmap. 2018
22. Marcus P, Dafydd S. The Web Application Hacker's handbook. 2011
23. Mastering Modern Web Penetration Testing. 2016
24. Meeuwisse R. Cybersecurity for all. 2009
25. More on SSL decryption. https://live.paloaltonetworks.com/t5/blogs/more-on-ssl-decryption/ba-p/342598 . 2020
26. Network Security, Firewalls, and VPNs. 2020
27. Oppliger R. Security Technologies for the World Wide Web. 2003
28. Pinto M, Stuttard D. Web Application Hacker's Handbook: Discovering and Exploiting Security Flaws. 2007
29. Prasad P. Mastering Modern Web Penetration Testing. 2016
30. Raef M. Cybersecurity for beginners. 2017
31. Recorded Future. The Threat Intelligence Handbook: A Practical Guide for Security Teams to Unlocking the Power of Intelligence. 2018
32. Reddy S. OWASP Top 10 for Layman. 2019
33. Rubinstein-Salzedo S. Cryptography. 2018
34. Seitz J. Black Hat Python: Programming for Hackers and Pentesters. 2014
35. Singh A. Socket programming with Python. 2019
36. SSL decryption? https://docs.paloaltonetworks.com/pan-os/

Veb sorğuların modifikasiyası üçün proksi serverin yazılması

Orxan Əlibəyli

XÜLASƏ

Hazırki internet erasını veb texnologiyalar olmadan təsəvvür edə bilməyin olduqca çətin olacağını nəzərə alsaq, bu texnologiyanın təhlükəsizliyinin təmin edilməsinin də olduqca vacib bir məsələ olduğu vurğulanmalıdır. Bu dissertasiya işinin əsas məqsədi də məhz bu sahəyə müəyyən dəstək verə bilməkdən ibarətdir.

Ümumiyyətlə veb sorğuların modifikasiyası nüfuzetmə testləri dediyimiz veb texnologiyaların təhlükəsizliyini yoxlamaq üçün olan proseslərin ayrılmaz tərkib hissəsidir. Modifikasiyaların həyata keçirlməsi əasasən xüsusi avtomatlaşdırma proqramları olmadan həyata keçirildiyindən effektiv nəticələrə nail olmaq çətin olur. Buna görə də bu dissertasiya işində yaradılan proqram bu kimi problemlərin həllində və nüfuzetmə testlərinin effektivlliyində kibertəhlükəsizlik professionallarına olduqca böyük köməklik göstərəcəkdir. Proqram həmçinin tətbiq edilən SSL/TLS modulları hesabına həm şifrələnmiş, həm də şirlənməmiş veb sorğuların modifikasiyasını rahatlıqla həyata keçirə bilir ki, bu da onu müxtəlif situasiyalarda istifadəyə olduqca yararlı edir.

Digər bir önəmli məqam isə ondan ibarətdir ki, yaradılam bu proqramın bütün kodları açıq şəkildə kibertəhlükəsizlik sahəsində olan peşəkarlar ilə paylaşılmışdır. Onlar həm bu proqramdan istifadə edə, həm də proqramın təkmilləşdirilməsi üçün öz təkliflərini edə bilərlər. Bu yaradılan proqram təminatının daha sürətli şəkildə inkişaf edə bilməsini təmin edəcəkdir.

Açar sözlər: Kibertəhlükəsizlik, veb sorğu, nüfuzetmə testi, şifrələmə, internet

Writing a proxy server for modifying web requests

Orkhan Alibayli

ABSTRACT

Considering that it would be very difficult to imagine the current Internet era without web technologies, it should be emphasized that ensuring the security of this technology is also a very important issue. The main goal of this dissertation is to be able to provide some support to this field.

In general, the modification of web requests is an integral part of the processes for checking the security of web technologies, which we call penetration tests. It is difficult to achieve effective results, since the implementation of modifications is mainly carried out without special automation programs. Therefore, the software created in this dissertation will be of great help to cyber security professionals in solving such problems and in the effectiveness of penetration tests. The program can also easily modify both encrypted and unencrypted web requests due to the implemented SSL/TLS modules, which makes it very useful in various situations.

Another important point is that all the code of this program has been openly shared with professionals in the field of cyber security. They can both use this program and make their suggestions for improving the program. This will ensure that the generated software can be developed faster.

Keywords: Cyber security, web survey, penetration testing, encryption, internet

Написание прокси-сервера для модификации веб-запросов

Орхан Алибейли

АБСТРАКТНЫЙ

Учитывая, что нынешнюю эпоху Интернета было бы очень сложно представить без веб-технологий, следует подчеркнуть, что обеспечение безопасности этой технологии также является очень важным вопросом. Основная цель этой диссертации - оказать некоторую поддержку в этой области.

В целом модификация веб-запросов является неотъемлемой частью процессов проверки безопасности веб-технологий, которые мы называем тестами на проникновение. Добиться эффективных результатов сложно, так как внедрение модификаций в основном осуществляется без специальных программ автоматизации. Поэтому программное обеспечение, созданное в этой диссертации, будет большим подспорьем специалистам по кибербезопасности в решении таких задач и в эффективности тестов на проникновение. Программа также может легко модифицировать как зашифрованные, так и незашифрованные веб-запросы благодаря реализованным модулям SSL/TLS, что делает ее очень полезной в различных ситуациях.

Еще одним важным моментом является то, что весь код этой программы был открыто предоставлен профессионалам в области кибербезопасности. Они могут как использовать эту программу, так и вносить свои предложения по улучшению программы. Это гарантирует, что сгенерированное программное обеспечение может быть разработано быстрее.

Ключевые слова: кибербезопасность, веб-опрос, тестирование на проникновение, шифрование, интернет.