Załącznik 2 do Projektu Umowy

OPIS PRZEDMIOTU ZAMÓWIENIA

Klaster sprzętowy wysokiej dostępności (HA) dwóch urządzeń sieciowych o funkcjonalności load balancera polegającej na równoważeniu obciążeń i rozdzielaniu ruchu pomiędzy aplikacje/serwery w celu optymalizacji wydajności, efektywności i przepustowości infrastruktury wraz z oprogramowaniem o minimalnej funkcjonalności opisanej poniżej.

Termin realizacji – 30 dni, z zastrzeżeniem postanowień oferty

Szkolenie – dla 5 administratorów

Warunki utrzymaniowe – zgodnie z umową

Gwarancja – minimum 24 miesiące od dnia dostawy i wdrożenia (podpisania protokołu), chyba że w ofercie wskazano inaczej.

System musi realizować co najmniej następujące funkcje:

1. Rozkład ruchu pomiędzy serwerami
2. Firewalla stanowego
3. System ochrony DNS oraz globalnego równoważenia obciążenia za pomocą protokołu DNS
4. Web Application Firewalla
5. Posiadać subskrypcję dla serwisu reputacyjnego (licencja w modelu subskrypcyjnym)
6. Posiadać subskrypcję dla serwisu z sygnaturami aktualnych kampanii zagrożeń w Internecie ((licencja w modelu subskrypcyjnym)
7. Wysoka dostępność i analityka

Wszystkie wymienione w niniejszym dokumencie funkcje muszą być dostępne w obrębie nie więcej niż jednego urządzenia fizycznego.

Wszystkie wymienione w niniejszym dokumencie funkcje (poza subskrypcjami dla serwisu reputacyjnego oraz serwisu z sygnaturami aktualnych kampanii zagrożeń w Internecie) muszą być licencjami, które nie mają ograniczeń czasowych i nie wygasają (są tzw. licencjami wiecznymi).

Zamawiający nie dopuszcza rozwiązania realizowanego w postaci maszyny (maszyn) wirtualnej w środowisku serwerowym (Vmware, Hyper-V, vcloud, KVM, Oracle VM, VirtualBox itp. Zastrzeżenie to nie dotyczy licencji laboratoryjnych.

Szczegółowy opis funkcji systemu:

1. System musi umożliwiać realizację rozkład ruchu pomiędzy serwerami
   1.1 System musi umożliwiać realizację rozdziału ruchu w oparciu o informację z warstw 4-7 modelu ISO/OSI.
   1.2 System musi realizować obsługę inteligentnego równoważenia ruchu dla farm serwerów przy wsparciu dla protokołów: TCP, UDP, FTP, TFTP, http, SSL/TLS,

DNS, RADIUS, SQL, RDP SSH PPTP NTP MQTT MQTT-TLS ISAKMP SMTP SIP

1.3 System musi mieć możliwość balansowania ruchu w oparciu o algorytmy:
- 1.3.1 Round Robin
- 1.3.2 cykliczna dla usługi
- 1.3.3 cykliczna dla serwera
- 1.3.4 ważona dla usługi
- 1.3.5 ważona dla serwera
- 1.3.6 obciążenie serwerów
- 1.3.7 ilość połączeń
- 1.3.8 ważona ilość połączeń dla serwerów
- 1.3.9 ważona ilość połączeń dla usługi
- 1.3.10 czas odpowiedzi
- 1.3.11 ilość sesji
- 1.3.12 hashing (URL, Domain, source IP, Destination IP)

1.4 System musi umożliwiać mechanizm dowiązania sesji (session persistence) w oparciu o:
- 1.4.1 cookie (wstawione przez load balancer lub ustanowiony na poziomie serwiera aplikacyjnego)
- 1.4.2 adres źródłowy
- 1.4.3 adres docelowy
- 1.4.4 identyfikator sesji SSL
- 1.4.5 SESSIONID
- 1.4.6 SIP call
- 1.4.7 Hash
- 1.4.8 Host
- 1.4.9 Msrdp
- 1.4.10 Universal (wybranie dowolnego fragmentu payloadu i utworzenie na jego podstawie profilu persystencji

1.5 System ma umożliwiać monitorowanie stanów serwerów i na tej podstawie dokonywania decyzji o przełączaniu w oparciu o:
- 1.5.1 ICMP
- 1.5.2 TCP
- 1.5.3 UDP
- 1.5.4 HTTP
- 1.5.5 HTTP/2
- 1.5.6 HTTPS
- 1.5.7 Inband
- 1.5.8 SNMP DCA
- 1.5.9 Server (co najmniej: ServerBandwidth, CPUPercentUsage, MemoryUsage, TotalClientCount)
- 1.5.10 Skrypty własne

1.6 System ma wspierać content switching w oparciu o
- 1.6.1 polityki: URL, URL query, URL wildcard Domain, Source IP, Destination IP, Nagłówek http, Dane HTTP i TCP, UDP
- 1.6.2 protokoły w przychodzących pakietach

1.7 System musi umożliwiać obsługę list kontroli dostępu dla 3 i 4 warstwy ISO/OSI

1.8 System musi umożliwiać zabezpieczenia przed atakami DNS, przynajmniej takimi jak DNS query floods, DNS amplification, zabezpieczenie przed atakami SYN flood

1.9 Selektywną kompresję danych

1.10 Terminację sesji SSL

1.11 Rozwiązanie musi pracować w trybie pełnego proxy

1.12 Praca w trybie pełnego proxy nie może powodować degradacji wydajności rozwiązania.

1.13 Buforowanie połączeń TCP w przypadku osiągnięcia zadanej ilości sesji dla danego serwera

1.14 Wsparcie dla HTTP/2, w tym wsparcie dla kompresji nagłówków

1.15 System musi posiadać funkcję definiowania maksymalnej ilości obsługiwanych przez dany serwer połączeń, w przypadku przekroczenia zdefiniowanej wartości musi istnieć możliwość wysłania klientowi strony błędu lub przekierowania klienta na inny serwer.

1.16 System musi zapewniać możliwość klonowania puli serwerów umożliwiając wysyłanie kopii ruchu do zewnętrznych systemów monitoringu lub urządzeń typu IDS/IPS.

1.17 System musi umożliwiać obsługę sieci co najmniej w zakresie: Routingu statycznego, Routingu dynamicznego OSPF, RIP, BGP, IS-IS, Link Aggregation, 802.3ad, VLAN 802.1q Jumbo Frames, z wykorzystaniem integracji z Equal Cost Multi-Path (ECMP)

1.18 System musi wspierać wysoką dostępność: Active/passive Active/active Connection Mirroring

1.19 Urządzenie powinno implementować TCP proxy z mechanizmem zamykania okna w stronę serwera www w przypadku zbyt wolnego odbierania danych przez zdalnego klienta.

1.20 Urządzenie musi mieć możliwość strojenia profili połączeń TCP, w tym co najmniej:

- 1.20.1 Ustawienia timeout idle
- 1.20.2 keep alive interval
- 1.20.3 zero window timeout
- 1.20.4 Initial Receive Window Size
- 1.20.5 Wyboru algorytmu wykrywania natłoku.
- 1.20.6 Wsparcia hybrydowych algorytmów wykrywania natłoku, opartych o stratę pakietów i round-trip-time
- 1.20.7 Mechanizmu rate pacing
- 1.20.8 określenia maksymalnego wielkości bufora per połączenie TCP celem ochrony przed wysyceniem pamięci
- 1.20.9 określenia maksymalnego bufora wysyłania i odbioru (send/receive buffer)

1.21 Urządzenie musi mieć możliwość włączenia ignorowania nagłówków przeglądarki dotyczących cachowania (Cache-control)

1.22 Urządzenie musi umożliwiać selektywne cachowanie zwracanego kontentu a cachowanie powinno być zależne od:

- 1.22.1 Listy URN-ów i prefixów URN-ów z których zwracany jest kontent
- 1.22.2 Maksymalnej wielkości zwracanego obiektu
- 1.22.3 minimalnej wielkości zwracanego obiektu

1.23 Powinna istnieć możliwość ograniczania wielkości cachu, ilości cachowanych obiektów celem ochrony pamięci urządzenia przed przepełnieniem.

1.24 Urządzenie musi wspierać multipleksacje wielu zapytań http w tej samej sesji TCP

1.25 Urządzenie musi umożliwiać kompresję zwracanej zawartości http. Użycie kompresji powinno być zależne od:

1.25.1 Listy dozwolonych URI

1.25.2 Listy wykluczonych URI

1.25.3 Listy możliwych do kompresji Content-Type

1.25.4 Listy wykluczonych Content-Type

1.25.5 Minimalnej wielkości zwracanego obiektu

1.26 Kierowanie ruchu do odpowiedniej pooli zasobów

1.26.1 Urządzenie powinno realizować mechanizm kierowania żądań od użytkowników końcowych do pool (grup) zasobów serwerowych na podstawie następujących atrybutów:

1.26.1.1 Wartości http hostname, początku/końca nagłówka hostname, ciąg znaków znajdujący się w nagłówku hostname

1.26.1.2 port

1.26.1.3 metody http

1.26.1.4 nazwy cookie, ciągu znaków na początku/końcu nazwy cookie, ciągu znaku zawartego w nazwie cookie

1.26.1.5 URN-a, oraz ciągu znaków występujących na początku/końcu lub zawartych w URN-ie

1.26.1.6 User agent

1.26.1.7 Dowolnego nagłówka lub kombinacji nagłówków

1.26.1.8 Wykorzystania cipherów w TLS

1.26.1.9 Common Name Klienta, oraz ciągu znaków występujących na początku/końcu lub zawartych w Common Name

1.26.2 Reguły kierowania żądań powinny wymuszać co najmniej następujące akcje:

1.26.2.1 Wybór pooli zasobów (grupy IP adresów i portów)

1.26.2.2 Kierowania żądania do konkretnego adresu IP

1.26.2.3 Redyrekcji

1.26.2.4 Wyłączanie / włączanie ochrony aplikacyjnej

1.26.2.5 Wybór polityki ochrony aplikacyjnej

1.26.2.6 Włączania/ wyłączania cachowania

1.26.2.7 Włączania/ wyłączania kompresji

1.26.2.8 Dodania / usunięcia nagłówka / cookie http

1.26.2.9 Wyboru parametrów profilu (zestawu parametrów, w tym CA, certyfikatu, klucza) dla połączeń w stronę serwerów aplikacyjnych.

1.26.3 Powinna istnieć możliwość nadpisania statycznych reguł kierowania ruchu w języku skryptowym, na podstawie dowolnych atrybutów żądania http, oraz stanu sesji zapisanego w pamięci urządzenia.

1.27 System musi zapewniać obsługę certyfikatów z kluczami typu ECDSA wykorzystującymi krzywe eliptyczne (ECC) zarówno od strony klienta, jak i od strony puli serwerów.

1.28 Sprzętowe wsparcie dla algorytmów AES, AES-GCM, RSA, DSA, DH, ECDSA, ECDH, SHA2. Wsparcie dla Perfect Forward Secrecy.

1.29 Dla protokołu TLS 1.2 wymagana jest obsługa AES-GCM zarówno od strony klienta, jak i od strony puli serwerów.
1.30 Wsparcie dla protokołu TLS 1.3.
1.31 System musi zapewniać obsługę certyfikatów podpisanych funkcją skrótu SHA-2 zarówno od strony klienta, jak i od strony puli serwerów.
1.32 System musi posiadać funkcję walidacji certyfikatów klientów łączących się przy wykorzystaniu protokołu SSL.
1.33 System musi obsługiwać sieci VLAN w standardzie 802.1q
1.34 System musi obsługiwać agregację linków w standardzie 802.3ad (LACP)
1.35 System musi obsługiwać Jumbo Frames
1.36 System musi posiadać funkcjonalność bramy VXLAN oraz NVGRE
1.37 Klucze prywatne zapisane na dysku urządzenia muszą być zaszyfrowane. Nie dopuszcza się rozwiązań przechowujących klucze prywatne w formie jawnej
1.38 Rozwiązanie musi posiadać wbudowany w system operacyjny język skryptowy, posiadający co najmniej następujące cechy:
1.38.1 Analiza, zmiana oraz zastępowanie parametrów w nagłówku http oraz w zawartości pakietów
1.38.2 Obsługa (parsowanie, modyfikacja, kreowanie odpowiedzi) protokołów: http, http2, WebSocket, DNS, RADIUS
1.38.3 Język skryptowy musi umożliwiać dynamiczne przypisanie polityki bezpieczeństwa aplikacyjnego w oparciu o dowolną kombinację atrybutów http, SSL/TLS.Parsowanie parametrów komunikacji TLS/SSL, w szczególności odczyt certyfikatów X.509 i wstrzykiwanie odczytanych parametrów w formie nagłówków http.
1.38.4 Możliwość zdefiniowania reakcji na zagrożenie wykryte przez Web Application Firewall na podstawie danych dostarczonych przez WAF, odczytanych w komunikacji http, SSL, WebSocket oraz historycznych atrybutów konfiguracji (kontekstu)
1.38.5 Język skryptowy oparty o Tool Command Language powinien mieć możliwość delegowania złożonych zadań do silnika NodeJS i użycia bibliotek „społecznościowych”, celem np. parsowania protokołów jeszcze nie zaimplementowanych w rozwiązaniu Application Delivery Controller

2. System musi umożliwiać realizację firewalla stanowego
2.1 konfigurację reguł firewallowych za pomocą wbudowanego w system interfejsu graficznego
2.2 Rozwiązanie powinno śledzić i ograniczać wystąpienia anomalii warstw niższych, w tym: IPv4, IPv6, TCP, UDP, ICP
2.3 Rozwiązanie powinno zawierać system sztucznej inteligencji analizujący na bieżąco ruchu i wykrywające ilościowe anomalie ruchu
2.4 Rozwiązanie powinno zawierać w pełni funkcjonalny firewall warstwy 4.
2.5 Rozwiązanie powinno być wyposażone w system ochrony przed atakami DDoS realizowany przynajmniej częściowo za pomocą dedykowanych komponentów hardware
2.5.1 Rozwiązanie powinno obsłużyć sprzętowo mitygację minimum 100 wektorów ataków DDoS

3. System ochrony DNS musi realizować co najmniej następujące funkcjonalności:

- 3.1.1 DNS Autorytatywny
- 3.1.2 Global Server Load Balancing
- 3.1.3 Dynamiczne podpisywanie domen DNS SEC
- 3.1.4 DNS Caching
- 3.1.5 DNS Resolver
- 3.1.6 Walidacja podpisów odpowiedzi DNS SEC
- 3.1.7 Rozkład ruchu pomiędzy serwerami DNS innych producentów
- 3.1.8 Ochrona przed atakami DDoS na protokół DNS
- 3.1.9 Ochrona przed atakami DDoS na niższe (niż DNS) protokoły w modelu ISO/OSI

3.2 System musi natywnie wspierać IPv4 i IPv6.
3.3 System musi obsługiwać sieci VLAN w standardzie 802.1q.
3.4 System musi obsługiwać agregację linków w standardzie 802.3ad (LACP).
3.5 System nie może wykorzystywać bibliotek i kodu „bind" do obsługi ruchu od użytkowników końcowych.
3.6 Urządzenie powinno wykorzystywać hardwarowe wsparcie dystrybucji requestów DNS pomiędzy wątki procesorów urządzenia. Hardwarowe rozwiązanie powinno zapewniać linearne skalowanie wydajności. (tego Łukasz nie był pewien i pytał o to nas ale nie uzyskał odpowiedzi)
3.7 Rozwiązanie powinno wspierać rozrzucanie ruchu DNS na podstawie następujących algorytmów:
- 3.7.1 cykliczna
- 3.7.2 ważona
- 3.7.3 najmniejsza liczba połączeń
- 3.7.4 najszybsza odpowiedź serwera
- 3.7.5 najmniejsza liczba połączeń i najszybsza odpowiedź serwera
- 3.7.6 najmniejsza liczba połączeń i najszybsza odpowiedź serwera w zdefiniowanym czasie
- 3.7.7 dynamicznie ważona oparta na SNMP/WMI
- 3.7.8 definiowana na podstawie grupy priorytetów dla serwerów
- 3.7.9 musi istnieć możliwość modyfikacji metod równoważenia obciążenia pomiędzy serwerami przy wykorzystaniu wbudowanego języka skryptowego

3.8 Rozwiązanie musi posiadać wbudowany w system operacyjny język skryptowy, posiadający co najmniej następujące cechy:
- 3.8.1 Analiza, zmiana oraz zastępowanie parametrów DNS
- 3.8.2 Generowania odpowiedzi na podstawie zapytania DNS oraz stanu zapisanego w pamięci urządzenia (np. historyczne zapytania DNS od danego użytkownika)
- 3.8.3 Powinna istnieć możliwość nadpisania statycznych reguł kierowania ruchu, na podstawie dowolnych atrybutów żądania DNS, oraz stanu sesji zapisanego w pamięci urządzenia.
- 3.8.4 Język skryptowy powinien umożliwiać iteracje poprzez wszystkie rekordy odpowiedzi DNS, oraz odczytywanie, modyfikacje i usuwanie poszczególnych rekordów
- 3.8.5 Język skryptowy powinien umożliwiać odczyt, modyfikacje i wstawienie sekcji additional
- 3.8.6 Język skryptowy powinien umożliwiać odczyt i modyfikację:

3.8.6.1 Typu rekordu
3.8.6.2 TTL
3.8.6.3 Nazwy rekordu
3.8.6.4 Klasy (class)

3.8.7 Język skryptowy powinien umożliwiać operacje porównania stringów: equals, contain, starts with, ends with

3.8.8 Język skryptowy powinien wspierać REGEXP

3.8.9 Język skryptowy powinien umożliwiać selektywne (per request) wyłączanie funkcji GSLB, DNS Resolver i Cache, autorytatywnego DNS

3.9 Powinna istnieć możliwość weryfikacji listy rekordów i reakcji / obsługi wyjątków, jeżeli domena lub zwracany adres IP znajduje się na liście

3.10 Powinna istnieć możliwość zaimportowania i użyciu statycznej listy „wyjątków" powyżej 200 000 rekordów

3.11 Powinna istnieć możliwość definiowania obsługi ruchu zarówno dla żądań jak i odpowiedzi DNS

3.12 Rozwiązanie powinno pełnić funkcje serwera autorytatywnego

3.13 Celem podwyższenia wydajności, zony statyczne powinny być ładowane do pomięci i obsługiwane z pamięci RAM (to jest Autoratytatywny DNS)

3.14 Autorytatywny serwer DNS powinien działać zarówno jako primary/master jak i secondary/slave (to jest Autoratytatywny DNS)

3.15 Rozwiązanie powinno wspierać zone transfer (to jest Autoratytatywny DNS, który robi DNS Express)

3.15.1 obsługiwać TSIG

3.15.2 inicjować transfer zony po otrzymaniu notyfikacji (NOTIFY)

3.15.3 Powinna istnieć możliwość przekazania notyfikacji do innych serwerów DNS

3.16 Powinna istnieć możliwość definiowania nazw dla których jest implementowany mechanizm GSLB (Global Server Load Balancing).

3.17 Nazwy zwracające dynamiczne rekordy (GSLB) powinny nadpisywać statyczną definicje strefy (zony).

3.18 Mechanizm GSLB powinien obsługiwać następujące typy rekordów:

3.18.1 A
3.18.2 AAAA
3.18.3 CNAME
3.18.4 MX
3.18.5 NAPTR
3.18.6 SRV

3.19 Silnik GSLB powinien zwracać adresy IP do zasobów, które obsługują aplikacje z zapytania DNS. Adresy IP będą dalej nazywane serwerami virtualnymi.

3.20 Silnik GSLB powinien uwzględniać stan zasobów znajdujących się za adresami IP, do których może być skierowany użytkownik poprzez zwrócenie adresu IPv4 lub IPv6.

3.21 Stan zasobów powinien być wymieniany asynchronicznie pomiędzy:

3.21.1 Innymi autorytatywnymi serwerami DNS implementującymi GSLB

3.21.2 Urządzeniami ADC (Application Delivery Controller) pochodzącymi od tego samego producenta

3.22 Serwery autorytatywne powinny pracować w klastrze N+1, tak, że kompletne odcięcie jednego centrum przetwarzania danych nie spowoduje utraty synchronizowanych informacji pomiędzy pozostałymi

3.23 Rozwiązanie nie powinno wymuszać twardego limitu na liczbę jednostek w klastrze

3.24 Rozwiązanie powinno wspierać mechanizm Route Healt Injection (RHI)

3.25 W przypadku, gdy nie jest możliwe otrzymanie stanu zasobów / serwerów virtualnych, powinna istnieć możliwość zdefiniowania i użycia monitora stanu aplikacji / serwera virtualnego.

3.26 Rozwiązanie GSLB powinno wspierać co najmniej następujące metody monitorowania:

- 3.26.1 http, https
- 3.26.2 IMAP
- 3.26.3 LDAP
- 3.26.4 MSSQL, MySQL, PostgresSQL
- 3.26.5 NNTP
- 3.26.6 POP3
- 3.26.7 RADIUS i RADIUS Accounting
- 3.26.8 SIP
- 3.26.9 SNMP
- 3.26.10 SMTP
- 3.26.11 SOAP
- 3.26.12 TCP
- 3.26.13 UDP
- 3.26.14 Monitora skryptowego, definiującego sekwencje wymiany wiadomości
- 3.26.15 Monitora zewnętrznego, czyli uruchomienia procesu np. bash, perl, zwracającego stan zasobów na podstawie logiki zawartej w zewnętrznym programie / skrypcie.

3.27 Silnik GSLB powinien wspierać co najmniej następujące algorytmy load balancingu:

- 3.27.1 Cykliczna
- 3.27.2 Ważona
- 3.27.3 Na podstawie adresu IP resolvera, implementująca persystencji (przywiązywanie sesji)
- 3.27.4 Do pierwszego dostępnego virtualnego serwera, przy czym rozwiązanie powinno umożliwiać zdefiniowanie kolejności virtualnych serwerów.
- 3.27.5 Lokalizacji pytającego serwera, na podstawie zaszytej bazy geolokacyjnej

3.28 Jeżeli wirtualny serwer jest zdefiniowany na urządzeniu ADC od tego samego producenta, silnik GSLB powinien wspierać co najmniej następujące algorytmy load balancingu:

- 3.28.1 Na podstawie wolnych zasobów procesora urządzenia ADC
- 3.28.2 Ilości hopów pomiędzy LDNS a ADC
- 3.28.3 Na podstawie aktualnego wysycenia połączeń do serwera wirtualnego. Mniejsze wysycenie oznacza większe prawdopodobieństwo wyboru
- 3.28.4 Aktualnej ilości obsługiwanych połączeń przez ADC
- 3.28.5 Pakietów na sekundę
- 3.28.6 Round Trip Time
- 3.28.7 Subiektywnej oceny wydajności przypisanej do wirtualnego serwera na urządzeniu ADC przez administratora
- 3.28.8 Średniej ważonej z powyższych metryk

3.29 Powinna istnieć możliwość grupowania obszarów geograficznych z dokładnością do poziomu województwa, tak aby zwiększyć prawdopodobieństwo połączenia Klienta z najbliższym CPD

3.30 Powinna istnieć możliwość grupowania prywatnych adresacji w obszary i wykorzystania tych obszarów do algorytmu load balancingu

3.31 Silnik GSLB powinien umożliwiać grupowanie zasobów / wirtualnych serwerów w Centra Przetwarzanie danych na podstawie ich rzeczywistej lokalizacji

3.32 System GSLB powinien wspierać mechanizmy oceny dostępności Centrum Przetwarzania Danych

3.33 W przypadku niedostępności Centrum Przetwarzania Danych, GSLB nie powinno zwracać adresów IP posadowionych w niedostępnym CPD, chyba że są one zdefiniowane statycznie.
3.34 Powinna istnieć możliwość delegacji procesu monitorowania wirtualnych serwerów posadowionych na rozwiązaniach firm trzecich do urządzenia ADC od tego samego dostawcy znajdującego się w tym samym CPD co wirtualne serwery.
3.35 Powinna istnieć możliwość delegacji procesu monitorowania wirtualnych serwerów, posadowionych na rozwiązaniach firm trzecich, do urządzenia ADC mającego routing do strefy, w której znajdują się monitorowane zasoby
3.36 Powinna istnieć możliwość grupowania nazw, tak żeby dany klient był obsługiwany z tego centrum przetwarzania danych dla wszystkich nazw ze zdefiniowanego zbioru
3.37 Rozwiązanie powinno umożliwiać implementacje cache'u wielopoziomowego
3.38 Rozwiązanie powinno wspierać trzy tryby pracy:
  3.38.1 Cache transparentny, gdzie cachowane są pełne odpowiedzi. W przypadku potrzeby rozwiązania zapytania (cache miss), cache transparentny przesyła zapytanie do drugiego poziomu cache-u
  3.38.2 Resolvera z cachem
  3.38.3 Resolvera walidującego podpisy DNS SEC z cachem
3.39 Rozwiązanie powinno umożliwiać granularną konfigurację każdego typu cache'u, tak aby uzyskać optymalny stosunek czasu wyszukiwania rekordów do cache hit ratio. Rozwiązanie powinno umożliwiać strojenie przynajmniej następujących parametrów
  3.39.1 Dla cache-u transparentnego: dostępnej pamięci dla cachu wiadomości i cachu rekordów
  3.39.2 W trybie pracy resolver z cachem: dostępnej pamięci dla cachu wiadomości, cachu rekordów i cachu autorytatywnych serwerów nazw
  3.39.3 W trybie pracy resolver walidującego DNS SEC z cachem: dostępnej pamięci dla cachu wiadomości, cachu rekordów, cachu autorytatywnych serwerów nazw i cachu kluczy DNS SEC
3.40 Resolver powinien zbierać i utrzymywać statystyki czasu odpowiedzi serwerów autoratywnych i wykorzystywać te statystyki w celu skrócenia czasu rozwiązywania nazw
3.41 Rozwiązanie powinno wykorzystywać IPv4 i IPv6 do komunikacji z serwerami autorytatywnymi
3.42 Rozwiązanie powinno umożliwiać przeszukiwanie cachowanych rekordów
3.43 Rozwiązanie powinno umożliwiać selektywne usuwanie cachowanych rekordów
3.44 System powinien filtrować w oparciu o Response Policy Zone
3.45 System powinien filtrować w oparciu o pliki płaskie, gdzie w pliku płaskim do każdej domeny może być przypisana akcja, np. NX Domain, odpowiedź CNAME, A record itd.
3.46 Resolver powinien zbierać co najmniej następujące statystyki:
  3.46.1 Ilość zapytań
  3.46.2 Odpowiedzi
  3.46.3 Wykorzystanych autorytatywnych serwerów
  3.46.4 Odpowiedzi wygenerowanych lokalnie
  3.46.5 Trafień w Response Policy Zone
  3.46.6 Trafień / nietrafień w cache, per typ cachu (cache wiadomości, resource rekord, autorytatywne servery)
  3.46.7 Wyników rozwiązywań zapytań, np. sukces, timeout, przepełnienie kolejki
  3.46.8 Odpowiedzi ze strony serwerów autorytatywnych
3.47 Statystyki powinny być dostępne za pomocą SNMP

3.48 Powinna istnieć możliwość definiowania wyjątków dla zapytań trafiających do systemu resolvera, przynajmniej dla następujących przypadków:
- 3.48.1 Powinna istnieć możliwość przekazania zapytania dla predefiniowanych subdomen do pooli serwerów DNS odpowiedzialnych za te domeny
- 3.48.2 Powinna istnieć możliwość przekazania zapytania dla predefiniowanych domen odwrotnych do pooli serwerów DNS odpowiedzialnych za te domeny odwrotne
- 3.48.3 Generowania predefiniowanej odpowiedzi, jeżeli w oryginalnej odpowiedzi z serwera autorytatywnego znajdzie się IP adres o złej reputacji.

3.49 Silnik systemu (resolvera) powinien mieć możliwość skorzystania z bazy reputacyjnej adresów IP celem usuwania tychże adresów z odpowiedzi od serwera autorytatywnego

3.50 Rozwiązanie powinno umożliwiać rozbudowę o system sygnatur IP

3.51 Rozwiązanie powinno filtrować po dowolnej kombinacji następujących typów zapytań:
a, axfr, dnskey, isdn, mb, naptr, nsec3param, rrsig, sshfp , a6, caa, ds, ixfr, md, nimloc, null, rt, tkey , aaaa, cert, eid, key, mf, ns, nxt, sig, tsig , afsdb, cname, gpos, kx, mg, nsap, opt, sink, txt , any, dhcid, hinfo, loc, minfo, nsap-ptr, ptr, soa, wks , apl, dlv, hip, maila, mr, nsec, px, spf, x25 , atma, dname, ipseckey, mailb, mx, nsec3, rp, srv, zxfr

3.52 Rozwiązanie powinno być odporne na ataki typu „Phantom domain attack”.

3.53 Rozwiązanie powinno być odporne na ataki typu Lock-up domain”

3.54 Rozwiązanie powinno być wyposażone w system ochrony przed atakami DDoS

3.55 Rozwiązanie powinno śledzić i ograniczać
- 3.55.1 Liczbę pakietów o nieprawidłowej składni
- 3.55.2 Liczbę pakietów per typ zapytania: A, PTR, NS, SOA, CNAME, MX, AAAA, TXT, SRV, AXFR, IXFR, ANY, NXDOMAIN, niemieszczących się w powyższych kategoriach.
- 3.55.3 Występowanie ataków typu reflection

3.56 System musi posiadać moduł analizy ruchu DNS. Moduł powinien zbierać następujące metryki:
- 3.56.1 Typy zapytań
- 3.56.2 Domeny
- 3.56.3 Najaktywniejsi Klienci per IP, Kraj
- 3.56.4 Typy ataków
- 3.56.5 Wektory ataków

3.57 Dla funkcjonalność GSLB system musi realizować co najmniej następujące funkcjonalności:
- 3.57.1 Global Server Load Balancing
- 3.57.2 Dynamiczne podpisywanie domen DNS SEC
- 3.57.3 DNS Caching
- 3.57.4 Walidacja podpisów odpowiedzi DNS SEC
- 3.57.5 Rozkład ruchu pomiędzy serwerami DNS innych producentów
- 3.57.6 Ochrona przed atakami DDoS na protokół DNS
- 3.57.7 Ochrona przed atakami DDoS na niższe (niż DNS) protokoły w modelu ISO/OSI

3.58 System musi natywnie wspierać IPv4 i IPv6.

3.59 System musi obsługiwać sieci VLAN w standardzie 802.1q.

3.60 System musi obsługiwać agregację linków w standardzie 802.3ad (LACP).

3.61 System nie może wykorzystywać bibliotek i kodu „bind” do obsługi ruchu od użytkowników końcowych.

3.62 Rozwiązanie powinno wspierać rozrzucanie ruchu DNS na podstawie następujących algorytmów:
- 3.62.1 cykliczna
- 3.62.2 ważona
- 3.62.3 najmniejsza liczba połączeń
- 3.62.4 najszybsza odpowiedź serwera
- 3.62.5 najmniejsza liczba połączeń i najszybsza odpowiedź serwera
- 3.62.6 najmniejsza liczba połączeń i najszybsza odpowiedź serwera w zdefiniowanym czasie
- 3.62.7 dynamicznie ważona oparta na SNMP/WMI
- 3.62.8 definiowana na podstawie grupy priorytetów dla serwerów
- 3.62.9 musi istnieć możliwość modyfikacji metod równoważenia obciążenia pomiędzy serwerami przy wykorzystaniu wbudowanego języka skryptowego

3.63 Rozwiązanie musi posiadać wbudowany w system operacyjny język skryptowy, posiadający co najmniej następujące cechy:
- 3.63.1 Analiza, zmiana oraz zastępowanie parametrów DNS
- 3.63.2 Generowania odpowiedzi na podstawie zapytania DNS oraz stanu zapisanego w pamięci urządzenia (np. historyczne zapytania DNS od danego użytkownika)
- 3.63.3 Powinna istnieć możliwość nadpisania statycznych reguł kierowania ruchu, na podstawie dowolnych atrybutów żądania DNS, oraz stanu sesji zapisanego w pamięci urządzenia.
- 3.63.4 Język skryptowy powinien umożliwiać iteracje poprzez wszystkie rekordy odpowiedzi DNS, oraz odczytywanie, modyfikacje i usuwanie poszczególnych rekordów
- 3.63.5 Język skryptowy powinien umożliwiać odczyt, modyfikacje i wstawienie sekcji additional
- 3.63.6 Język skryptowy powinien umożliwiać odczyt i modyfikację:
  - 3.63.6.1 Typu rekordu
  - 3.63.6.2 TTL
  - 3.63.6.3 Nazwy rekordu
  - 3.63.6.4 Klasy (class)
- 3.63.7 Język skryptowy powinien umożliwiać operacje porównania stringów: equals, contain, starts with, ends with
- 3.63.8 Język skryptowy powinien wspierać REGEXP
- 3.63.9 Język skryptowy powinien umożliwiać selektywne (per request) wyłączanie funkcji GSLB, DNS Cache

3.64 Powinna istnieć możliwość weryfikacji listy rekordów i reakcji / obsługi wyjątków, jeżeli domena lub zwracany adres IP znajduje się na liście

3.65 Powinna istnieć możliwość zaimportowania i użyciu statycznej listy „wyjątków” powyżej 200 000 rekordów

3.66 Powinna istnieć możliwość definiowania obsługi ruchu zarówno dla żądań jak i odpowiedzi DNS

3.67 Powinna istnieć możliwość definiowania nazw dla których jest implementowany mechanizm GSLB (Global Server Load Balancing).

3.68 Nazwy zwracające dynamiczne rekordy (GSLB) powinny nadpisywać statyczną definicje strefy (zony).

3.69 Mechanizm GSLB powinien obsługiwać następujące typy rekordów:
- 3.69.1 A
- 3.69.2 AAAA
- 3.69.3 CNAME
- 3.69.4 MX
- 3.69.5 NAPTR
- 3.69.6 SRV

3.70 Silnik GSLB powinien zwracać adresy IP do zasobów, które obsługują aplikacje z zapytania DNS. Adresy IP będą dalej nazywane serwerami virtualnymi.

3.71 Silnik GSLB powinien uwzględniać stan zasobów znajdujących się za adresami IP, do których może być skierowany użytkownik poprzez zwrócenie adresu IPv4 lub IPv6.

3.72 Stan zasobów powinien być wymieniany asynchronicznie pomiędzy:
- 3.72.1 Innymi systemami implementującymi GSLB
- 3.72.2 Urządzeniami ADC (Application Delivery Controller) pochodzącymi od tego samego producenta

3.73 System z funkcją GSLB powinien pracować w klastrze N+1, tak, że kompletne odcięcie jednego centrum przetwarzania danych nie spowoduje utraty synchronizowanych informacji pomiędzy pozostałymi

3.74 Rozwiązanie nie powinno wymuszać twardego limitu na liczbę jednostek w klastrze

3.75 Rozwiązanie powinno wspierać mechanizm Route Healt Injection (RHI)

3.76 W przypadku, gdy nie jest możliwe otrzymanie stanu zasobów / serwerów virtualnych, powinna istnieć możliwość zdefiniowania i użycia monitora stanu aplikacji / serwera virtualnego.

3.77 Rozwiązanie GSLB powinno wspierać co najmniej następujące metody monitorowania:
- 3.77.1 http, https
- 3.77.2 IMAP
- 3.77.3 LDAP
- 3.77.4 MSSQL, MySQL, PostgresSQL
- 3.77.5 NNTP
- 3.77.6 POP3
- 3.77.7 RADIUS i RADIUS Accounting
- 3.77.8 SIP
- 3.77.9 SNMP
- 3.77.10 SMTP
- 3.77.11 SOAP
- 3.77.12 TCP
- 3.77.13 UDP
- 3.77.14 Monitora skryptowego, definiującego sekwencje wymiany wiadomości
- 3.77.15 Monitora zewnętrznego, czyli uruchomienia procesu np. bash, perl, zwracającego stan zasobów na podstawie logiki zawartej w zewnętrznym programie / skrypcie.

3.78 Silnik GSLB powinien wspierać co najmniej następujące algorytmy load balancingu:
- 3.78.1 Cykliczna
- 3.78.2 Ważona
- 3.78.3 Na podstawie adresu IP resolvera, implementująca persystencji (przywiązywanie sesji)
- 3.78.4 Do pierwszego dostępnego virtualnego serwera, przy czym rozwiązanie powinno umożliwiać zdefiniowanie kolejności virtualnych serwerów.
- 3.78.5 Lokalizacji pytającego serwera, na podstawie zaszytej bazy geolokacyjnej

3.79 Jeżeli wirtualny serwer jest zdefiniowany na urządzeniu ADC od tego samego producenta, silnik GSLB powinien wspierać co najmniej następujące algorytmy load balancingu:
- 3.79.1 Na podstawie wolnych zasobów procesora urządzenia ADC
- 3.79.2 Ilości hopów pomiędzy LDNS a ADC
- 3.79.3 Na podstawie aktualnego wysycenia połączeń do serwera wirtualnego. Mniejsze wysycenie oznacza większe prawdopodobieństwo wyboru
- 3.79.4 Aktualnej ilości obsługiwanych połączeń przez ADC
- 3.79.5 Pakietów na sekundę
- 3.79.6 Round Trip Time
- 3.79.7 Subiektywnej oceny wydajności przypisanej do wirtualnego serwera na urządzeniu ADC przez administratora
- 3.79.8 Średniej ważonej z powyższych metryk

3.80 Powinna istnieć możliwość grupowania obszarów geograficznych z dokładnością do poziomu województwa, tak aby zwiększyć prawdopodobieństwo połączenia Klienta z najbliższym CPD

3.81 Powinna istnieć możliwość grupowania prywatnych adresacji w obszary i wykorzystania tych obszarów do algorytmu load balancingu

3.82 Silnik GSLB powinien umożliwiać grupowanie zasobów / wirtualnych serwerów w Centra Przetwarzanie danych na podstawie ich rzeczywistej lokalizacji

3.83 System GSLB powinien wspierać mechanizmy oceny dostępności Centrum Przetwarzania Danych

3.84 W przypadku niedostępności Centrum Przetwarzania Danych, GSLB nie powinno zwracać adresów IP posadowionych w niedostępnym CPD, chyba że są one zdefiniowane statycznie.

3.85 Powinna istnieć możliwość delegacji procesu monitorowania wirtualnych serwerów posadowionych na rozwiązaniach firm trzecich do urządzenia ADC od tego samego dostawcy znajdującego się w tym samym CPD co wirtualne serwery.

3.86 Powinna istnieć możliwość delegacji procesu monitorowania wirtualnych serwerów, posadowionych na rozwiązaniach firm trzecich, do urządzenia ADC mającego routing do strefy, w której znajdują się monitorowane zasoby

3.87 Powinna istnieć możliwość grupowania nazw, tak żeby dany klient był obsługiwany z tego centrum przetwarzania danych dla wszystkich nazw ze zdefiniowanego zbioru

3.88 Rozwiązanie powinno wspierać cache transparentny, gdzie cachowane są pełne odpowiedzi. W przypadku potrzeby rozwiązania zapytania (cache miss), cache transparentny przesyła zapytanie do drugiego poziomu cache-u

3.89 Rozwiązanie powinno umożliwiać granularną konfigurację cache'u, tak aby uzyskać optymalny stosunek czasu wyszukiwania rekordów do cache hit ratio. Rozwiązanie powinno umożliwiać strojenie dostępnej pamięci dla cachu wiadomości i cachu rekordów

3.90 Rozwiązanie powinno wykorzystywać IPv4 i IPv6 do komunikacji z serwerami autorytatywnymi

3.91 Rozwiązanie powinno umożliwiać przeszukiwanie cachowanych rekordów

3.92 Rozwiązanie powinno umożliwiać selektywne usuwanie cachowanych rekordów

3.93 System powinien filtrować w oparciu o Response Policy Zone

3.94 System powinien filtrować w oparciu o pliki płaskie, gdzie w pliku płaskim do każdej domeny może być przypisana akcja, np. NX Domain, odpowiedź CNAME, A record itd.

3.95 Statystyki powinny być dostępne za pomocą SNMP

3.96 Powinna istnieć możliwość definiowania wyjątków dla zapytań trafiających do systemu przynajmniej dla następujących przypadków:

3.96.1 Powinna istnieć możliwość przekazania zapytania dla predefiniowanych subdomen do pooli serwerów DNS odpowiedzialnych za te domeny
3.96.2 Powinna istnieć możliwość przekazania zapytania dla predefiniowanych domen odwrotnych do pooli serwerów DNS odpowiedzialnych za te domeny odwrotne
3.96.3 Generowania predefiniowanej odpowiedzi, jeżeli w oryginalnej odpowiedzi z serwera autorytatywnego znajdzie się IP adres o złej reputacji.
3.97 Rozwiązanie powinno umożliwiać rozbudowę o system sygnatur IP
3.98 Rozwiązanie powinno filtrować po dowolnej kombinacji następujących typów zapytań:
a, axfr, dnskey, isdn, mb, naptr, nsec3param, rrsig, sshfp , a6, caa, ds, ixfr, md, nimloc, null, rt, tkey , aaaa, cert, eid, key, mf, ns, nxt, sig, tsig , afsdb, cname, gpos, kx, mg, nsap, opt, sink, txt , any, dhcid, hinfo, loc, minfo, nsap-ptr, ptr, soa, wks , apl, dlv, hip, maila, mr, nsec, px, spf, x25 , atma, dname, ipseckey, mailb, mx, nsec3, rp, srv, zxfr
3.99 Rozwiązanie powinno być odporne na ataki typu „Phantom domain attack”.
3.100 Rozwiązanie powinno być odporne na ataki typu Lock-up domain”
3.101 Rozwiązanie powinno być wyposażone w system ochrony przed atakami DDoS
3.102 Rozwiązanie powinno śledzić i ograniczać
3.102.1 Liczbę pakietów o nieprawidłowej składni
3.102.2 Liczbę pakietów per typ zapytania: A, PTR, NS, SOA, CNAME, MX, AAAA, TXT, SRV, AXFR, IXFR, ANY, NXDOMAIN, niemieszczących się w powyższych kategoriach.
3.102.3 Występowanie ataków typu reflection
3.103 System musi posiadać moduł analizy ruchu DNS. Moduł powinien zbierać następujące metryki:
3.103.1 Typy zapytań
3.103.2 Domeny
3.103.3 Najaktywniejsi Klienci per IP, Kraj
3.103.4 Typy ataków
3.103.5 Wektory ataków

4. Web Application Firewall musi działać w oparciu o pozytywny model bezpieczeństwa (tylko to, co znane i prawidłowe jest dozwolone), model ten tworzony jest na bazie automatycznie budowanego przez WAF profilu aplikacji Web. Firewall aplikacyjny musi działać jednocześnie z wykorzystanie pozytywnego i negatywnego modelu bezpieczeństwa:
4.1 Pozytywny model bezpieczeństwa powinien kontrolować co najmniej:
4.1.1 wystąpienie URL-i, długość URL-i,
4.1.2 typ servleta występujący pod danym url-em – format komunikacji (http form, JSON, XML, GWT)
4.1.3 przejścia pomiędzy URL-ami (servletami)
4.1.4 dopuszczalne metody http,
4.1.5 dopuszczalne cookie,
4.1.6 dopuszczalne parametry w polityce,
4.1.7 parametry dynamiczne,
4.1.8 typ/format parametrów (alfanumeryczny, integer, dynamiczny, statyczny, JSON, XML, e-mail, telefon, plik uploadowany)

4.1.9 dopuszczalne parametry w danym servlecie
4.1.10 długość zapytań
4.1.11 nazwy hosta
4.1.12 wystąpień i długość parametrów (per każdy parametr)
4.1.13 wystąpień i długości nagłówków
4.1.14 wystąpień i długości cookies
4.1.15 oczekiwanych typów znaków per każdy parametr
4.1.16 typów rozszerzeń plików; w tym długości URLa, requestu, query stringu, post data dla danego typu pliku URL-i podatnych na CSRF

4.2 Profil aplikacji web musi być tworzony na podstawie analizy ruchu sieciowego

4.3 WAF musi umożliwiać definiowania dopuszczalnego przepływu sekwencji zapytań w obrębie aplikacji z uwzględnieniem jej logiki biznesowej

4.4 Tworzenie profilu bezpieczeństwa Web Application Firewall dla danej aplikacji musi odbywać się na podstawie analizy ruchu sieciowego w szczególności na podstawie publicznego ruchu produkcyjnego.

4.5 System musi posiadać funkcję definiowania i edycji szablonów konfiguracji aplikacji. Szablony powinny służyć do optymalizacji procesu wdrażania systemu zarówno dla znanych aplikacji biznesowych, jak i własnych aplikacji klienta. W ramach opisanych szablonów musi istnieć możliwość automatycznej kontroli poszczególnych elementów konfiguracji szablonu i zabezpieczenie ich przed modyfikacją i usunięciem.

4.6 Algorytmy tworzenia profilu bezpieczeństwa WAF powinny odrzucać próby ataków w procesie nauki.

4.7 Musi istnieć możliwość definicji zaufanych adresów źródłowych, z których algorytm tworzenia profilu bezpieczeństwa WAF będzie akceptować wszystkie zachowania jako prawidłowe, tak aby administrator mógł przyspieszyć proces tworzenia profilu bezpieczeństwa.

4.8 Musi istnieć możliwość ręcznego konfigurowania/modyfikacji reguł polityki bezpieczeństwa

4.9 Musi istnieć możliwość ochrony dynamicznych oraz ukrytych parametrów zapytań http

4.10 WAF musi automatycznie wykrywać false positive i wyłączać odpowiadające nim sygnatury dla danego parametru

4.11 WAF musi posiadać funkcjonalność automatycznego wykrywania stron logowania użytkowników oraz automatycznie włączać dla tych stron ochronę przed atakami brute force.

4.12 Mechanizm zabezpieczenia przed manipulacją cookie serwera aplikacyjnego powinien być oparty o wstrzykiwanie cookie z podpisem oryginalnego cookie aplikacji.

4.13 WAF powinien chronić przed kradzieżą sesji poprzez porównywanie „odcisku palca” (fingerprint) przeglądarki z sesją użytkownika. Mechanizm musi działać także dla TLS (TLS fingerprintig)

4.14 Mechanizm zabezpieczenia przed Cross-Site Request Forgery powinien dodawać losowy token do odpowiedzi http zawierających odwołania do chronionego zasobu (servleta).

4.15 System musi zapewniać możliwość wyboru polityki bezpieczeństwa na podstawie: Host, URL, Nagłówków, Cookie

4.16 Oprócz pozytywnego modelu zabezpieczeń WAF musi posiadać również funkcje identyfikacji incydentów poprzez sygnatury (negatywny model zabezpieczeń)

4.17 WAF musi posiadać mechanizmy ochrony przed atakami: Broken Access Control, SQL Injection, Cross-Site Scripting, Cross-Site Request Forgery, Session hijacking, Command Injection, Cookie/Session Poisoning, Parameter/Form Tampering, Forceful Browsing, Brute Force Login, Web Scraping, Cookie manipulation/poisoning, Dynamic Parameter tampering, Buffer Overflow, Stealth Commanding, Unused HTTP Methods, Malicious File Uploads, Hidden Field Manipulation

4.18 Musi istnieć możliwość selektywnego włączania/wyłączania sygnatur per parametr

4.19 Dla każdej chronionej aplikacji internetowej urządzenie powinno umożliwiać wybór stosowanych technologii i systemu operacyjnego w celu poprawnego doboru wykorzystywanych sygnatur uwzględniając, ale nie ograniczając się do:

4.19.1 Bazy danych: ORACLE, MySQL, Microsoft SQL Server, PostgreSQL, Sybase, IBM DB2, CouchDB, Elasticsearch, MongoDB, SQLite, Sybase/ASE

4.19.2 System Operacyjny: Windows, Linux, UNIX

4.19.3 Język aplikacji, frameworki, biblioteki: ASP, ASP.NET, PHP, Java Servlets, JavaScript, AngualrJS, Backbone.js, CodeIgniter, Django, Java Server Faces, BEA WebLogic, CGI, Elasticsearch, Front Page Server Extension, Lotus Domino, Macromedia ColdFusion, Outlook Web Access, SSI, WebDAV, jQuery, SSI, Apache Struts, ef.js, Ember.js, Express.js, GraphQL, Handlebars, JavaServer Faces, Laravel, MooTools, Moustache, Python, React, RequireJS, Ruby, Spring Boot, UIKit, Underscore.js, Vue.js, WebDAV, Zend, ZURB Fundation

4.19.4 Serwer WWW, silniki: Apache, Apache Tomcat, Microsoft IIS, serwerów proxy, Jenkins, Jetty, Joomla, Macromedia JRun, Nginx, Node.js, Oracle Application Server, Oracle Identity Manager, Redis, Typo3 CMS

4.20 WAF musi posiadać mechanizmy ochrony przed atakami DoS ukierunkowanymi na warstwę aplikacyjną (np. Slow Loris, http Smuggling,)

4.21 WAF musi rozróżniać rzeczywistych użytkowników od automatów podczas ataku (D)DoS poprzez:

4.21.1 Wstrzykiwanie skryptu JavaScript i weryfikacji rezultatów jego wykonania

4.21.2 Mechanizmu browser fingerprinting, w celu wykrycia tzw. headless broser

4.21.3 Sygnatur botów

4.21.4 Wykorzystanie CAPTCHA (tylko w przypadku, gdy powyższe mechanizmy nie rozstrzygają czy podłączony jest rzeczywisty użytkownik).

4.22 System powinien umożliwiać proaktywne wykrywanie i blokowanie botów (j.w.), zanim wywołają atak DDoS, web scraping lub brute force.

4.23 System musi rozróżniać ruch generowany przez boty za pomocą zasobów lokalnych

4.24 System musi posiadać możliwość rozbudowy licencyjnej o wysyłanie części ruchu do dalszej analizy w scrubbing center, w celu ochrony przed botami

4.25 WAF musi zawierać moduł sztucznej inteligencji, który na bieżąco obserwuje ruch od użytkowników końcowych, celem budowy i utrzymania modelu prawidłowego ruchu do aplikacji. WAF na podstawie behawioralnej analizy ruchu

bieżącego i zbudowanego modelu, powinien wykrywać i chronić aplikację przed atakiem DDoS w warstwie 7. W systemie nie może być żadnego licencyjnego limitu dla tej funkcji.

4.26 System powinien kategoryzować boty i umożliwiać przepuszczanie ruchu od pożytecznych botów (np. search enginy), blokując ruch od szkodliwych botów.

4.27 Moduł ochrony przed DDoS powinien wykrywać ataki per:Source IP, Urządzenie, na bazie „odcisku palca" urządzenia, Obszar geolokacyjny, URL, Globalnie – website

4.28 Powinna istnieć możliwość przypisania różnych poziomów detekcji ataków (D)DoS dla danych URL-i portala. Np. /infoportal/* powinien posiadać luźniejszą politykę detekcji i zapobiegania ataków DDoS niż /portal/*.

4.29 System powinien wykrywać i chronić przed atakami DDoS na tzw. ciężkie servlety, czyli takie wywołujące złożone operacje obliczeniowe np. skomplikowane zapytania do baz danych.

4.30 Wykrycie ataku na ciężkie servlety powinno opierać się przynajmniej o ilość zapytań (TPS) oraz czas odpowiedzi

4.31 System powinien umożliwiać definicję maksymalnego czasu próbki ruchu, maksymalnej pojemności próbki ruchu, interwału czasowego pomiędzy pobieraniem próbki ruchu.

4.32 System powinien umożliwiać automatyczny zapis przykładowego ruchu do plików zgodnych z formatem TCP dump, w momencie wykrycia ataku (D)DoS.

4.33 Powinna istnieć możliwość doboru odpowiedzi w zależności do rodzaju naruszenia

4.34 WAF musi posiadać możliwość uwzględniania w logach dotyczących incydentów informacji o uwierzytelnionym użytkowniku oraz blokowania określonej liczby incydentów wykonywanych w zdefiniowanym czasie przez tego użytkownika.

4.35 WAF powinien umożliwiać usuwanie nagłówków serwera aplikacyjnego zdradzających technologię oraz wersję oprogramowania; bez uszczerbku na wydajności WAF-a.

4.36 WAF powinien umożliwiać wstrzykiwanie nagłówków np. w celu ochrony przed Clickjack-iem

4.37 WAF powinien umożliwiać podmianę kodów statusów zwracanych przez serwer aplikacyjny

4.38 W obrębie licencji WAF dostarczony musi być moduł ochrony protokołu HTTP, SMTP oraz FTP DNS, API

4.39 WAF musi mieć możliwość wgrania pliku swagger file (Open API) w celu budowy polityki typu whitelist dla API

4.40 WAF musi posiadać wsparcie dla aplikacji AJAX oraz JSON.

4.41 WAF powinien wyświetlać stron blokowania (błędu) w technologiach AJAX i JSON

4.42 WAF musi posiadać wsparcie dla Google Web Toolkit

4.43 WAF musi posiadać wsparcie dla GraphQL

4.44 WAF musi posiadać możliwość ochrony komunikacji XML poprzez:

4.44.1 walidację Schema/WSDL,

4.44.2 wybór dozwolonych metod SOAP,

4.44.3 szyfrację /deszyfrację fragmentów wiadomości SOAP,

4.44.4 wsparcie dla WS-Security (szyfracja, deszyfracja, veryfikacja i podpisywanie),
4.44.5 definiowanie możliwości użycia załączników wiadomości SOAP,
4.44.6 włączanie/wyłączanie podążania za odnośnikami do schematów SOAP,
4.44.7 walidację SOAP Action Header,
4.44.8 włączanie/wyłączanie możliwości użycia DTD
4.44.9 włączanie/wyłączanie możliwości użycia zewnętrznych referencji
4.44.10 włączanie/wyłączanie możliwości użycia początkowych białych znaków
4.44.11 włączanie/wyłączanie możliwości użycia numerycznych nazw
4.44.12 włączanie/wyłączanie możliwości użycia Processing Instructions
4.44.13 włączanie/wyłączanie możliwości użycia CDATA
4.44.14 ograniczenie długości: dokumentu, elementu, nazwy, wartości atrybutu, Namespace
4.44.15 ograniczenia ilości: zagnieżdżeń w dokumencie, dzieci per element, atrybutów per element, deklaracji NameSpace-ów
4.44.16 definicję dopuszczalnych znaków
4.44.17 definicję sygnatur.

4.45 WAF musi umożliwiać blokowanie zapytań z danego obszaru geograficznego. Aktualizacje bazy geolokacyjnej powinny być dostępne w ramach podstawowych opłat wsparcia.

4.46 WAF musi umożliwiać automatyczne budowanie polityk w oparciu o skanowanie przez zewnętrznych dostawców (przynajmniej trzech) np. Cenzic, HP WebInspect, IBM AppScan, Qualys Guard, WhiteHat Sentinel.

4.47 WAF musi posiadać mechanizmy normalizacji w celu obrony przed technikami ukrywania ataku. Mechanizmy normalizacji muszą wspierać/wykrywać:
4.47.1 Directory traversal
4.47.2 Kodowanie typu %u
4.47.3 Kodowanie typu IIS backslash
4.47.4 IIS Unicode codepoints
4.47.5 Bare byte decoding
4.47.6 Apache whitespace
4.47.7 Bad unescape
4.47.8 Wstrzykiwanie komentarzy (np. <!-- -->)

4.48 Mechanizm normalizacji powinien umożliwiać definiowanie maksymalnego zagnieżdżonego kodowania.

4.49 Urządzenie musi wspierać następujące tryby pracy:
4.49.1 Tryb wykrywania, logowania i blokowania ataków
4.49.2 Tryb wykrywania i logowania ataków bez blokowania
4.49.3 Tryb uczenia się bez blokowania
4.49.4 Tryb uczenia się z blokowaniem i logowaniem

4.50 WAF w trybie nauki, musi umożliwiać automatyczne, stopniowe przełączanie polityki bezpieczeństwa w tryb blokowania, np. servlety/parametry, dla których został zaobserwowany wystarczający ruch dla algorytmu nauki, zostaną przełączone w tryb blokowania, podczas gdy pozostałe pozostaną w trybie transparentnym.

4.51 WAF musi umożliwiać integracje systemami antywirusowymi po protokole ICAP w celu wykrywania wirusów w przesyłanych plikach.

4.52 WAF musi wykrywać i maskować numery kart kredytowych, wyciekających z chronionej aplikacji; oraz dowolnie inny ciąg znaków zdefiniowany poprzez PCRE regular expression.

4.53 WAF musi chronić ruch przesyłany po IPv6 bez degradacji wydajności wynikającej z innych czynników niż różnice protokołów IPv4 i IPv6

4.54 System musi umożliwiać szyfrowanie wskazanych pół (np. pole do wprowadzania danych typu hasło) w czasie rzeczywistym, wprowadzanym w przeglądarce internetowej.

4.55 Szyfrowanie musi być również dostępne, jeżeli formularz logowania wykorzystuje technologię AJAX. Szyfrowanie tych pól musi odbywać się z wykorzystaniem klucza publicznego osadzanego przez rozwiązanie w odpowiedzi serwera aplikacyjnego. System nie może wymagać zmiany po stronie samej aplikacji ani wymagać instalacji dodatkowego oprogramowania na urządzeniu końcowym.

4.56 System musi umożliwiać szyfrowanie w czasie rzeczywistym nazw wskazanych pól w kodzie HTML oraz dodawać dodatkowe pola typu input, by strona logowania www nie była statyczna (dodawanie dodatkowych pól typu input musi być niewidoczne dla użytkownika końcowego na stronie www).

5. Wraz z rozwiązaniem należy dostarczyć 3 letnią subskrypcję dla serwisu reputacyjnego (licencja w modelu subskrypcyjnym).
   5.1 Serwis reputacyjny powinien być dostępny jako rozszerzenie systemu, bez konieczności wprowadzania zmian w architekturze sprzętowej oraz programowej proponowanego rozwiązania.
   5.2 Serwis reputacyjny musi realizować co najmniej następujące funkcje:
      5.2.1 Automatyczna aktualizacja informacji o zagrożeniach nie rzadziej niż co 5 minut
      5.2.2 Rozpoznawać i blokować komunikację dla minimum poniższych:
         5.2.2.1 Anonimowych proxy
         5.2.2.2 Sieci Botnet
         5.2.2.3 Aktywnych źródeł usług oferujących lub dystrybuujących malware, rootkity, robaki oraz wirusy
         5.2.2.4 Źródeł ataków DDoS/DoS
         5.2.2.5 Źródeł Exit Node sieci Tor
         5.2.2.6 Adresów IP zainfekowanych przez malware
         5.2.2.7 Adresów IP świadczących usługi hostingowe dla phishingu lub fraudów.
         5.2.2.8 Źródeł ataków cross-site scripting, iFrame injection, SQL injection, cross domain injection czy domain password brute force
         5.2.2.9 Źródłowych adresów IP skanerów służących do rekonesansu poprzez skanowanie hostów oraz domen

6. Wraz z rozwiązaniem należy dostarczyć 3 letnią subskrypcję dla serwisu z sygnaturami aktualnych kampanii zagrożeń w Internecie (licencja w modelu subskrypcyjnym).
   6.1 Subskrypcje na sygnatury kontekstowe powiązane z konkretną kampanią ataków
   6.2 Serwis sygnatur z aktualnymi kampaniami musi realizować następujące funkcje i informacje:

6.2.1 Natychmiastowa aktualizacja sygnatur z aktualnymi kampaniami
6.2.2 Musi dostarczać informacji o:
6.2.2.1 powiązaniu z konkretnym aktorem stojącym za atakami
6.2.2.2 wektorze ataku lub wykorzystanej technice
6.2.2.3 intencji ataku
6.2.2.4 datach dotyczących pierwszej obserwacji ataku, ostatnio widzianego ataku
6.2.2.5 statusie kampanii
6.2.2.6 analizie wykorzystanego payloadu
6.2.2.7 informacji o dodatkowych szkodach

7. Wysoka dostępność i analityka
7.1 System musi posiadać co najmniej następujące interfejsy administracyjne:
7.1.1 GUI przy wykorzystaniu protokołu https
7.1.2 Zarządzanie poprzez SSH
7.1.3 Zarządzanie poprzez API REST
7.2 System musi posiadać moduł analizy ruchu http. Moduł powinien zbierać następujące metryki
7.2.1 Czas odpowiedzi per serwer
7.2.2 Czas odpowiedzi per URI
7.2.3 Ilość sesji użytkownika
7.2.4 Przepustowość
7.2.5 Adres źródła
7.2.6 Kraj
7.2.7 User Agent (wykorzystywana przez klienta aplikacja)
7.2.8 Metoda dostępu
7.3 System musi posiadać następujące funkcje zarządzania siecią:
7.3.1 Obsługa protokołu SNMP v1/v2c/v3
7.3.2 Możliwość budowania własnych zdarzeń SNMP z własnymi numerami OID
7.3.3 Zewnętrzny syslog
7.3.4 Możliwość wysyłania logów syslog do więcej niż jednego miejsca docelowego
7.3.5 Zbieranie danych i ich wyświetlanie
7.3.6 Zbieranie danych zgodnie z ustawieniami administratora
7.3.7 Osobna brama domyślna dla interfejsu zarządzającego
7.3.8 Wsparcie dla przynajmniej 2 wersji oprogramowania (multi-boot) lub zaoferowanie rozwiązania polegającego na przełączeniu się na węzeł pracujący na innej wersji oprogramowania;
7.3.9 zapisywanie konfiguracji (możliwość szyfrowania i eksportu kluczy)
7.3.10 Dedykowany podsystem monitorowania stanu pracy urządzenia (always on management) z funkcjami restartu, wstrzymania oraz sprzętowego resetu systemu.
7.4 System musi posiadać funkcję integracji z zewnętrznymi serwerami uwierzytelnienia użytkowników LDAP, RADIUS, TACACS.
7.5 Autoryzacja administratorów systemu musi bazować na rolach użytkowników

7.6 System musi posiadać funkcję definiowania i edycji szablonów konfiguracji aplikacji. Szablony powinny służyć do optymalizacji procesu wdrażania systemu zarówno dla znanych aplikacji biznesowych, jak i własnych aplikacji klienta. W ramach opisanych szablonów musi istnieć możliwość automatycznej kontroli poszczególnych elementów konfiguracji szablonu i zabezpieczenie ich przed modyfikacją i usunięciem.

7.7 Rozwiązanie musi oferować podział na tzw. partycje administracyjne. Zdefiniowany użytkownik może zarządzać konfiguracją tylko i wyłącznie wewnątrz swojej partycji.

7.8 Wykrycie awarii urządzeń w klastrze odbywać się musi przy użyciu, co najmniej następujących metod:

7.8.1 Weryfikacja stanu pracy urządzenia poprzez analizę aktywności w sieci (Network failover)

Dla oferowanego systemu wymagana jest minimum 2 letnia (24 miesięcy) gwarancja, z zastrzeżeniem postanowień z oferty wykonawcy. W obrębie gwarancji zawarte musi być:

1.1 Dostęp do aktualnych wersji oprogramowania oraz dokumentacji producenta
1.2 Sposób obsługi zgłoszeń gwarancyjnych w trybie 7x24
1.3 Wymiana sprzętu następnego dnia roboczego po identyfikacji usterki
1.4 W wypadku awarii dyski zostają u Zamawiającego

System w postaci jednego urządzenia musi spełniać wymogi przedstawione w tabeli 1

Tabela 1. Każde z dwóch urządzeń systemu musi spełniać wymogi przedstawione poniżej (Wymagania dla jednego urządzenia systemu)

Lp.	Parametr	Wymagania
	Pamięć	Nie mniej niż 256 GB
	Dysk twardy	Dwa dyski SSD o pojemności nie mniejszej niż 1TB U.2 SSD pracujące w RAID1
	Przepływność dla warstwy 4	Nie mniej niż 190 Gbps
	Przepływność dla warstwy 7	Nie mniej niż 125 Gbps, licencyjna możliwość aktualizacji do 190 Gbps
	Ilość jednocześnie obsługiwanych połączeń	Nie mniej niż 145 milionów, licencyjna możliwość aktualizacji do 180 milionów

	Ilość transakcji SSL na sekundę dla klucza o długości 2048	Nie mniej niż 115 tysięcy, licencyjna możliwość aktualizacji do 200 tysięcy
	Ilość transakcji SSL na sekundę dla szyfru ECDSA P-256	Nie mniej niż 90 tysięcy, licencyjna możliwość aktualizacji do 140 tysięcy
	Przepływność ruchu szyfrowanego	Nie mniej niż 75 Gbps, licencyjna możliwość aktualizacji do 90 Gbps
	Ilość połączeń na sekundę w warstwie 4	Nie mniej niż 1 milion 900 tysięcy, licencyjna możliwość aktualizacji do 2 milion 500 tysięcy
	Kompresja sprzętowa	Nie mniej niż 80 Gbps, licencyjna możliwość aktualizacji do 90 Gbps
	Sprzętowa ochrona DDoS	Nie mniej niż 160 milionów SYN cookies na sekundę
	Liczba wirtualnych instancji	Nie mniej niż 24, licencyjna możliwość aktualizacji do 36
	Gęstość interfejsów	Nie mniej niż szesnaście interfejsów z możliwością obsadzenia wkładkami SFP+ 10G lub 25G (SR lub LR), nie mniej niż cztery interfejsy z możliwością obsadzenia wkładkami 100G lub 40G QSFP+, oddzielny interfejs zarządzania, port konsolowy, port USB Dla każdego z dostarczonych urządzeń powinny zostać dostarczone cztery wkładki SFP+ 10G SR Dopuszcza się tylko moduły w pełni wspierane przez producenta tego urządzenia.
	Zarządzanie	Panel i wyświetlacz LCD (dotykowy) z funkcjami: ustawienia adresu IP na potrzeby zarządzania, ustawienia parametrów portu szeregowego, wyświetlania podstawowych alarmów, możliwości restartu urządzenia, wyświetlania informacji o systemie Funkcjonalność „Always On Management”
	Obudowa	Przeznaczona do montażu w szafie rack 19”, wysokość nie większa niż 1 U

	Zasilanie	Nie mniej niż dwa redundantne zasilacze - prąd zmienny 230V AC
	Wymagana certyfikacja	EN 62368-1:2014+A11:2017 EN 60950-1:2006+A11:2009+A1:2010+A12:2011+A2:2013 EN 55032:2012/AC:2013 Class A EN 55035:2017 EN 300 386 V1.6.1 (2012) - Lub równoważne