Web Services

https://campus.barracuda.com/doc/22118445/

A web application is designed to take input from a human user and display output to a human user. In contrast, a web service is an application that is accessible on the web but is intended to be used by another application. Web services share business logic, data, and processes through a programmatic interface. Web services allow businesses to communicate with each other and with clients without requiring inside knowledge of each other's infrastructure and security configurations. They are used to assist organizations in streamlining business processes, providing increased efficiency and reduced application integration costs.

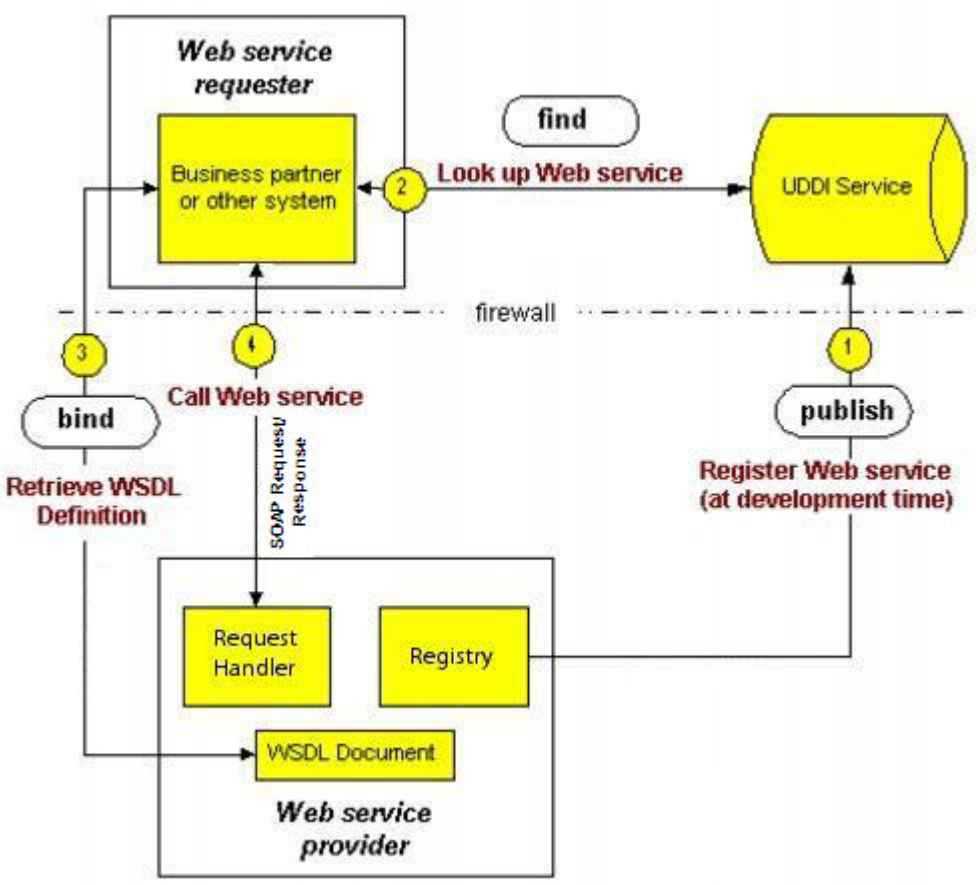
Web Services Implementation

Web services use a universal language to send data and instructions to one another over the Internet with no translation required. The term web service describes a standardized way of integrating web based applications using the Extensible Markup Language (XML), Simple Object Access Protocol (SOAP), Web Services Description Language (WSDL), and Universal Description, Discovery, and Integration (UDDI) open standards over an Internet protocol (usually HTTP). XML is used to tag the data, SOAP is used to exchange the data, WSDL is used for describing the services available, and UDDI is used for listing available services.

For example, consider two banks needing to share account balance information. Figure 14.1 illustrates the steps to share the data using a web service:

1. Bank A creates a web service description (in the XML-based WSDL format) that describes web service required inputs and outputs (for example, the customer's account number and password) and sends a SOAP request to register it with a UDDI service.
2. Bank B sends a SOAP request to the UDDI service to look up information about Bank A's web service. (While using a UDDI service is common practice, it is not a requirement for a web service.)
3. Bank B sends a SOAP request to Bank A's web service to retrieve the WSDL definition and bind to that web service.
4. Bank B sends a SOAP request to Bank A's web service that conforms to the WSDL definition.

In this example, access to account information must be restricted to approved intermediaries, requiring authentication through passwords, public keys, or other mechanisms. In addition, the bank might want to prioritize requests (such as by how much customers are paying for the service), confirm that payment for the service is received, or send a receipt. Each function requires that information be secure from unauthorized access, attack, and data theft.



A business can combine multiple web services to accomplish a task. For example, a travel service might define one web service for interacting with a client application, another web service for communicating with a credit card service (with the travel service acting as the client of the credit card service), another for communicating with one or more hotel services, and another for communicating with one or more airline services.

WSDL

A web service description (WSDL document) is a human-readable document, written from the web service perspective, that describes the expectations and functionality of a particular web service, and clarifies how client and service should interact. A potential client reads the web service description, to learn how to correctly interact with the service.

WSDL is an XML grammar for describing network services as collections of communication end points capable of exchanging messages. WSDL service definitions provide documentation for distributed systems and serve as a recipe for automating the details of application communications.

Web Service Vulnerabilities

Web services are vulnerable to many of the same attack risks as other web applications, but also face additional vulnerabilities including:

- Publicly available WSDL documents provide a blueprint for the service. The document details messaging request and response, expected parameters (including data type), and available operations for the service. By analyzing its WSDL document, a hacker not only knows exactly what the service is supposed to do, but also which parts are open to attack through techniques such as malformed SOAP messages and other XML parser attacks. A WSDL document may reveal what tools generated the Web service, providing attackers with more insight into potential vulnerabilities.
- SOAP and XML are standards used to wrap data for easy consumption. SOAP envelops information to deliver messages seamlessly between applications. XML includes metadata to describe the structure of the information. CDATA is used to delineate information in the message that should not be parsed. Malicious code or characters can be embedded in the elements or CDATA, allowing unintentional display or execution by the receiving application or service. XML encapsulation (a form of cross site scripting) can embed commands that tie up system resources or gain unauthorized access.
- XML-based attacks can overload XML parsers which process SOAP messages. Attackers that put in recursive relationships to create entity expansions, bogus parameters, or even significant amounts of white space, can cause XML parsers to be overloaded or to behave unpredictably.
- Any type of application behind a web service interface, including a packaged application, an internally developed application, a desktop application, or a legacy mainframe application carries its own security vulnerabilities. These inherent security risks are even more exposed through a web service interface. Because each application type approaches security its own way, it’s a significant security challenge to protect these services.

Web Service Protections

The following table describes possible web service attack techniques and the corresponding protection provided by the Barracuda Web Application Firewall.

Technique	Description of Attack	Barracuda Web Application Firewall Protection
Schema Poisoning	Manipulating the WS schema to alter the data processed by the application.	Protects against schema poisoning by validating that content adheres to the defined WSDL and schema.

XML Parameter Tampering	Injection of malicious scripts or content into XML parameters.	Protects against parameter tampering by validating that parameter values are consistent with the WSDL and schema specifications.
Inadvertent XDoS	Sending poorly encoded SOAP messages that cause the application to fail.	Inspects SOAP at the header, envelope, and message level to ensure proper structure and content.
WSDL Scanning	Scanning the WSDL (business API) to uncover sensitive information about the application data format.	Uses web services cloaking to hide the true internal URI of sensitive web services.
Coercive Parsing	Injection of malicious content into the XML.	Utilizes real-time WS-I checking and content inspection to block malicious payloads.
Oversized Payload	Sending oversized files to create an XDoS attack (similar to a buffer overflow attack).	Inspects transmitted data and enforces element, document, and other maximum sizes.
Recursive Payload	Sending mass amounts of nested data to create an XDoS attack on the XML parser.	Validates WSDL and schema formats, inspects SOAP headers, envelopes, and messages, and ensures that WS-I standards are met.
SQL Injection	Hiding a malicious SQL command inside a SOAP wrapper attempting to uncover or modify back-end data.	Utilizes real-time WS-I checking and content inspection to compare to schema.
Replay Attacks	Using repetitive SOAP messages to force an XDoS attack	Includes request-level throttling technology to ensure resources cannot reach a fail state.
External Entity Attack	Parses XML input from untrusted sources using an incorrectly configured XML parser.	Can suppress external URI references to protect against external manipulation of data.
Information Disclosure	Exposes unencrypted Web Service message data to anyone watching application traffic.	Has extensive SSL security capabilities at the ASIC level to ensure end to end encryption of XML traffic.
Malicious Code Injection	Delivers scripts embedded deep within SOAP messages directly to applications and databases.	Ensures SOAP messages conform to customized policies.
Identity Centric Attack	Forges credentials in an attempt to access sensitive data	Enforces authentication (basic or strong) at the SOAP message level.
Processing Instructions (PI)	Uses PI (a text data section that is ignored by the XML parser) to pass instructions to applications.	Can block requests containing Processing Instructions (PI).
Inline or external DTDs	Uses DTDs (a text data section that is ignored by the XML parser) to pass instructions to applications.	Can block requests containing both inline or external DTDs.

External References	Uses requests containing external entities including external URI references or external DTDs (text data sections that are ignored by the XML parser) to pass instructions to applications.	Can block requests containing external entities including external URI references or external DTDs.

Figures