BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC DÂN LẬP HẢI PHÒNG

-------o0o-------

NGHIÊN CỨU BẢO MẬT WEB SERVICE

ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC HỆ CHÍNH QUY

Ngành: Công nghệ Thông tin

Sinh viên thực hiện: Đoàn Đức Trung

Giáo viên hướng dẫn: Ths. Nguyễn Trịnh Đông

Mã số sinh viên: 110655

MỤC LỤC

DANH MỤC HÌNH VẼ

DANH MỤC KÝ HIỆU VÀ TỪ VIẾT TẮT

Tên viết tắt	Mô tả
FTP	File Transfer Protocol
HTTP	Hypertext Transfer Protocol
HTML	Hypertext Markup Language
UDDI	Universal Description Discovery and Integration
SOAP	Simple Object Access Protocol
SOA	Service Oriented Architecture
SOSA	Service Oriented Security Architecture
SSL	Security Sockets Layers
SAML	Security Assertion Markup Language
TCP/IP	Transmission Control Protocol/ Internet Protocol
XML	eXtensible Markup Language
XACML	eXtensible Access Control Markup Language
XKMS	XML Key Management Specification
XrML	eXentisble Rights Markup Language
WSE	Web Service Enhancement
WSDL	Web Service Description Language
WS	Web Service

TÓM TẮT NỘI DUNG

Ngày nay công nghệ thông tin đang là nền công nghệ mũi nhọn trong chiến lược phát triển kinh tế, xây dựng đất nước của hầu hết các quốc gia. Các sản phẩm công nghệ thông tin đã và đang được ứng dụng rộng rãi trong mọi lĩnh vực của đời sống kinh tế, xã hội và hầu hết đều đem đến những giá trị thiết thực. Đối tượng phục vụ chủ yếu của ngành công nghệ thông tin hiện nay chính là các tổ chức, các cơ sở doanh nghiệp…

Bảo mật luôn luôn là một vấn đề hàng đầu cho tất cả các loại ứng dụng, đặc biệt là các ứng dụng web. Từ những ngày đầu của Internet người ta đã quan tâm đến tính an toàn trong trao đổi thông tin. Tuy không có sự an toàn tuyệt đối nhưng những phát triển trong lĩnh vực này thì rất nhanh và mang lại nhiều thành quả vì đây là vấn đề cấp bách của nhiều doanh nghiệp. Không có một mức an toàn thích hợp, sự khai thác thương mại của Internet thì không hoàn toàn an toàn. Do đó những giải thuật để kiểm chứng, sự mã hóa khóa thông tin, và chữ ký số hóa có thể là những giải pháp cung cấp một mức đủ an toàn.

Chính vì thế sự an toàn của Web Service trên mạng cũng không thể nằm ngoài vấn đề này. Có thể nói ngày nay ngoài việc nghiên cứu làm sao để tạo ra một Web Service tốt mang lại nhiều lợi ích thì việc nghiên cứu để làm sao mang lại sự an toàn cho Web Service cũng là một trong những vấn đề quan trọng nhất. Thật khó tin tưởng để sử dụng một dịch như mua chứng khoán, chuyển tiền trực tuyến hoặc truyền cơ sở dữ liệu qua lại giữa hai máy tính mà lại không có sự an toàn cần thiết.

Em đã chọn đề tài làm đồ án tốt nghiệp là "***Nghiên cứu bảo mật Web Service***". Đồ án tập trung đi sâu vào tìm hiểu về công nghệ Web Service và các vấn đề bảo mật liên quan và sử dụng chúng để giải quyết bài toán đề ra.

CHƯƠNG 1: MỞ ĐẦU

1.1. Đặt vấn đề

Ngày nay, cùng với sự phát triển của Internet, Web Service cũng trở thành một kỹ thuật dùng để liên kết và tương tác giữa các ứng dụng trên các máy tính khác nhau thông qua môi trường Internet. Ngày càng có nhiều nhà cung cấp dịch vụ muốn đưa các dịch vụ ra công cộng và vấn đề lớn nhất mà các nhà cung cấp đang phải đối mặt chính là bảo mật cho Web Service. Việc đảm bảo an toàn cho Web Service là một vấn đề đặc biệt quan trọng, nhất là đối với những dịch vụ liên quan tài chính, thị trường chứng khoán và thương mại điện tử. Vấn đề bài toán đặt ra là làm thế nào để những thông tin, dữ liệu được trao đổi một cách an toàn mà không bị tấn công.

Để giải quyết vấn đề bảo mật trên đường truyền, có nhiều phương pháp, công cụ được xây dựng và một trong số đó là Web Service Enhancement 3.0. Bộ thư viện này cung cấp nhiều hình thức chứng thực và nhiều chuẩn đặc tả khác nhau về bảo mật cũng như phục vụ mục đích đa dạng của người sử dụng. Ngoài ra còn rất nhiều các kỹ thuật bảo mật khác đang được các doanh nghiệp nhỏ và vừa triển khai trên hệ thống mạng nhằm đảm bảo thật tốt vấn đề an ninh khi giao dịch trên Internet.

1.2. Nội dung bài toán

Với những yêu cầu mà thực tế đặt ra, đồ án này sẽ tìm hiểu và làm rõ các kỹ thuật bảo mật Web Service hiện có, cùng với đó sẽ tập trung đi sâu vào bộ công cụ Web Service Enhancement 3.0 nhằm giải quyết rõ hơn về vấn đề bảo mật. Cũng trong đồ án này sẽ thực hiện xây dựng một hệ thống đơn giản là thực hiện việc trao đổi dữ liệu giữa hai máy tính trong mạng cục bộ với nhau và bảo mật dữ liệu đó trên đường truyền. Quá trình thực hiện gọi tới Web Service và hiển thị dữ liệu sẽ được bảo mật bằng bộ thư viện Web Service Enhancement 3.0, nhằm bảo mật dịch vụ web cũng như phục vụ mục đích đa dạng của người dùng.

Có rất nhiều công cụ bảo mật cho hệ thống thông tin như FireWall, công nghệ bảo mật SSL, hệ thống xác thực (CA) và đặc biệt ứng dụng trong Web Service là bộ thư viện Web Service Enhancement của .NET Framework Microsoft. Web Service giao tiếp thông qua các thông điệp SOAP. Web Service Enhancement cung cấp những mở rộng của giao thức SOAP và cho phép người dùng tự định nghĩa các chính sách,bảo mật phục vụ việc truyền thông điệp trở nên đáng tin cậy

1.3. Mục tiêu của đồ án

Để thực hiện các vấn đề nêu ra như trên, đồ án sẽ lần lượt trình bày những kiến thức cần thiết để giải quyết yêu cầu của bài toán đặt ra. Đồ án sẽ tập trung vào một số các vấn đề sau:

- Tìm hiểu khái quát về kiến trúc hướng dịch vụ SOA.
- Tìm hiểu công nghệ Web Service, kiến trúc và các thành phần Web Service.
- Tìm hiểu Service Proxy (dạng Web Service triển khai ở phía người dùng).
- Tìm hiểu các kỹ thuật bảo mật Web Service.
- Triển khai ứng dụng về bảo mật Web Service ứng dụng WSE 3.0.

1.4. Cấu trúc của đồ án

Đồ án bao gồm các chương như sau:

- Chương 1: Giới thiệu về kiến trúc hướng dịch vụ.
- Chương 2: Web Service
- Chương 3: Kỹ thuật bảo mật Web Service
- Chương 4: Các kỹ thuật bảo mật Web Service
- Chương 5: Triển khai ứng dụng và đánh giá kết quả
- Kết luận

CHƯƠNG 2: GIỚI THIỆU KIẾN TRÚC HƯỚNG DỊCH VỤ

2.1.Thực trạng hiện tại

Phần mềm ngày nay đang ngày càng trở nên phức tạp và dường như đang vượt khỏi khả năng kiểm soát của các mô hình phát triển phần mềm hiện có. Hàng chục năm qua,nhiều kiến trúc phần mềm đã được xây dựng và triển khai nhằm giải quyết các vấn đề này.Thế nhưng độ phức tạp phần mềm vẫn cứ tiếp tục tăng và dường như đã trở nên vượt quá khả năng xử lý của các kiến trúc truyền thống.

Nguyên nhân khiến cho độ phức tạp của các hệ thống phần mềm không ngừng tăng cao như thế là do sự xuất hiện của nhiều công nghệ mới tạo nên môi trường không đồng nhất, trong khi nhu cầu về trao đổi, chia sẻ, tương tác giữa các hệ thống không thể đáp ứng được trong một môi trường như vậy.

Một nguyên nhân khác cũng góp phần dẫn đến tình trạng khó khăn như thế chính là vấn đề lập trình dư thừa và không thể tái sử dụng.

Những vấn đề trước chưa giải quyết, mà nay các tổ chức lại phải đối mặt với những thách thức mới: đáp ứng nhanh chóng các sự thay đổi về thiết bị, giảm chi phí phát triển, tăng tính tương thích và khả năng tái sử dụng,... Tất cả đã tạo nên một áp lực nặng nề đối với các nhà phát triển phần mềm.

2.2. Phân tích, đánh giá một số mô hình kiến trúc phân tán hiện tại

Ba kiến trúc phân tán phổ biến nhất hiện này là CORBA, DCOM và EJB. Các kiến trúc này là sự mở rộng của các hệ thống hướng đối tượng bằng cách cho phép phân tán các đối tượng trên mạng. Đối tượng đó có thể có không gian địa chỉ bên ngoài ứng dụng, hoặc ở một máy khác với máy chứa ứng dụng trong khi vẫn được tham chiếu sử dụng như một phần của ứng dụng.

2.2.1. CORBA - Common Object Request Broker Architecture

CORBA được định nghĩa bởi Object Management Group (OMG), là một kiến trúc phân tán mở, độc lập nền tảng và độc lập ngôn ngữ.[1]

CORBA Component Model (CCM) là một cải tiến đáng kể nhằm định nghĩa các mô hình thành phần so với CORBA. Nó định nghĩa ra quy trình thiết kế,phát triển, đóng gói,triển khai và thực thi các thành phần phân tán.CCM định nghĩa khái niệm cổng cho các thành tố. Các cổng này được sử dụng để kết nối các thành phần có sẵn

với nhau, tạo các hệ thống phân tán phức tạp hơn. Mỗi thành phần CCM có một đối tượng Home chịu trách nhiệm quản lý chu kỳ sống của đối tượng và được triển khai bên trong một trình chứa.

Ưu điểm của CORBA là các lập trình viên có thể chọn bất kỳ ngôn ngữ,nền tảng phần cứng, giao thức mạng và công nghệ để phát triển mà vẫn thỏa mãn các tính chất của CORBA. Tuy nhiên CORBA có 1 một số nhược điểm đó là ngôn ngữ lập trình cấp thấp, rất phức tạp, khó học và cần một đội ngũ phát triển có kinh nghiệm. Ngoài ra các đối tượng CORBA cũng khó có thể tái sử dụng.

2.2.2. EJB - Enterprise Java Bean

Kiến trúc EJB là một kiến trúc thành tố bên phía máy chủ dùng cho việc phát triển và triển khai các ứng dụng phân tán hướng đối tượng cỡ vừa và lớn.

Kiến trúc EJB có ba tầng với tầng đầu tiên là tầng trình diễn, tầng thứ hai là tầng xử lý nghiệp vụ và tầng thứ ba là các tài nguyên như cơ sở dữ liệu máy chủ. Các đối tượng EJB giao tiếp qua Remote Method Invocation (RMI).Các Máy khách sẽ sử dụng phương thức được định nghĩa trong Giao diện kết nối từ xa. Mỗi bean bên trong trình chứa, chịu trách nhiệm việc tạo giao diện, lưu trữ dữ liệu.Trình chứa sẽ triệu gọi các phương thức *callback* của mỗi thể hiện bean khi có sự kiện tương ứng. Không giống như CORBA Component Model, EJB không định nghĩa các cổng kết nối trực tiếp giữa các thành phần liên quan bởi vì mỗi bean bên trong trình chứa là một thực thể độc lập không có bất kỳ ràng buộc bên ngoài nào.[1]

EJB là một kiến trúc tốt cho việc tích hợp các hệ thống vì nó độc lập nền tảng nhưng nó cũng gặp vấn đề là không phải là một chuẩn mở, khả năng giao tiếp với các chuẩn khác vẫn còn hạn chế.

2.2.3. DCOM - Distributed Component Object Model

DCOM là một mô hình phân tán dễ triển khai với chi phí thấp, hỗ trợ việc ghép kín giữa các ứng dụng và hệ điều hành. Mô hình Component Object Model (COM) định nghĩa cách thức các thành phần và Máy khách liên lạc trao đổi với nhau trên cùng một máy. DCOM mở rộng COM bằng cách sử dụng các giao thức trên mạng chuẩn khi cần trao đổi dữ liệu với máy móc. DCOM hỗ trợ kết nối giữa các đối tượng và có thể được thay đổi lúc đang chạy. Các đối tượng DCOM được triển khai bên trong các gói nhị phân chứa các mã lệnh quản lý chu kỳ sống của đối tượng và việc đăng ký nó[1].

DCOM mang đến nhiều ưu điểm như tính ổn định, không phụ thuộc vị trí địa lý, quản lý kết nối hiệu quả và dễ dàng mở rộng, là một lựa chọn tốt cho các doanh nghiệp sử dụng công nghệ của Windows để chạy các ứng dụng có yêu cầu cao về chính xác và ổn định. Tuy nhiên, các công nghệ của Microsoft có một nhược điểm lớn là chúng bị giới hạn trên nền tảng Windows.

❖ Tóm lại:

Các kiến trúc trên đều hướng đến việc xây dựng một hệ thống "hướng dịch vụ" tuy nhiên chúng vẫn còn gặp phải một số vấn đề sau:

- Kiến trúc cài đặt bên phía nhà cung cấp và phía sử dụng phải giống nhau. Điều này đồng nghĩa với khó khăn mỗi khi có sự thay đổi từ một trong hai phía.
- Các chuẩn trên đa phần là chuẩn đóng, chúng hầu như không thể kết hợp, hoạt động với chuẩn khác..

2.3. Khái niệm SOA

Theo định nghĩa của IBM: "*SOA is an architecture style for creating an Enterprise IT Architecture that exploits the principle of máy chủ orientation to achieve a tiglter relationship between the business and the information systems that support the business…"*[6].

Theo đó SOA là phong cách kiến trúc để tạo ra một công trình kiến trúc IT, kiến trúc đó khai thác các nguyên tắc của hướng dịch vụ để đạt được các mối quan hệ chặt chẽ giữa doanh nghiệp và hệ thống thông tin nhằm hỗ trợ các doanh nghiệp.

Kiến trúc hướng dịch vụ là một hướng tiếp cận với việc thiết kế và tích hợp các phần mềm, chức năng, hệ thống theo dạng mô đun, mỗi mô đun sẽ có một tính chất "kết nối lỏng" và có khả năng truy cập thông qua môi trường mạng. Hiểu một cách đơn giản thì một hệ thống SOA là một tập hợp các dịch vụ được chuẩn hoá trên mạng trao đổi với nhau trong ngữ cảnh một tiến trình nghiệp vụ [3].

SOA đưa ra giải pháp để giải quyết các vấn đề tồn tại của các hệ thống hiện nay như: phức tạp, không linh hoạt và không ổn định. Một hệ thống triển khai theo mô hình SOA có khả năng dễ mở rộng, liên kết tốt. Đây chính là cơ sở và nền tảng cho việc tích hợp, tái sử dụng lại những tài nguyên hiện có. SOA cung cấp cơ chế cho phép các hệ thống hoạt động trên các platform khác nhau có thể giao tiếp với nhau.

Thiết kế SOA tách riêng phần thực hiện dịch vụ với giao tiếp gọi dịch vụ. Điều này tạo nên một giao tiếp nhất quán cho ứng dụng khách. Thay vì xây dựng các ứng dụng đơn lẻ và đồ sộ, nhà phát triển sẽ xây dựng các dịch vụ tinh gọn có thể triển khai và tái sử dụng trong toàn bộ quy trình nghiệp vụ. Điều này cho phép tái sử dụng phần mềm tốt hơn, cũng như tăng sự linh hoạt vì nhà phát triển có thể cải tiến dịch vụ mà không làm ảnh hưởng đến ứng dụng của máy khách.

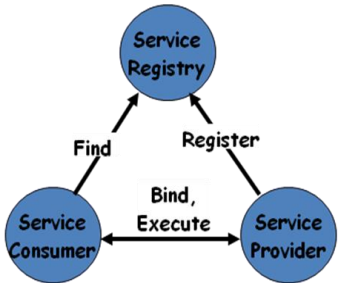
Thật ra, tư tưởng về một hệ thống SOA không phải là mới. Comnon Object Request Broker Architecture (CORBA) và mô hình Distributed Component Object Model (DCOM) của Microsoft hay như Enterprise Java Bean (EJB) của Java đã cung cấp tính năng này từ lâu. Tuy nhiên những cách tiếp cận này còn gặp phải những vấn đề khó khăn như trên và SOA không chỉ là một cải tiến đáng kể giúp giải quyết những yếu điểm của các công nghệ trước mà còn đem đến nhiều ưu điểm nổi trội hơn.

2.4. Đối tượng trong hệ thống xây dựng theo SOA

Service Provider: Cung cấp dịch vụ phục vụ cho một nhu cầu nào đó.

Service Consumer: Người dùng sử dụng các dịch vụ của Servie Provider.

Service Registry: Nơi lưu trữ thông tin về các dịch vụ khác nhau, Service Consumer dựa trên những thông tin này để tìm kiếm và lựa chọn Service Provider.



Hình 2.1: Hoạt động của SOA

2.5. Nguyên tắc chính của hệ thống SOA

2.5.1. Sự phân định ranh giới rạch ròi giữa các dịch vụ

Các dịch vụ thực hiện quá trình tương tác chủ yếu thông qua thành phần giao tiếp.Thành phần giao tiếp sẽ qui định về định dạng thông điệp nào sẽ được chấp nhận và thông điệp nào sẽ không được xử lý.Đây là cách duy nhất để các đối tượng bên ngoài có thể truy cập thông tin và chức năng của dịch vụ.Chỉ cần gửi các thông điệp theo các định dạng đã được định nghĩa mà không cần phải quan tâm đến cách xử lý của dịch vụ như thế nào.

2.5.2. Các dịch vụ tự hoạt động

Các dịch vụ cần phải được triển khai và hoạt động như những thực thể độc lập mà không lệ thuộc vào một dịch vụ khác. Dịch vụ phải có tính bền vững cao, nghĩa là nó sẽ không bị sụp đổ khi có sự cố. Để thực hiện điều này, dịch vụ cần duy trì đầy đủ thông tin cần thiết cho quá trình hoạt động của mình để có thể tiếp tục hoạt động trong trường hợp một dịch vụ cộng tác bị hỏng và để tránh các cuộc tấn công từ bên ngoài (như gửi thông điệp lỗi, hay gửi thông điệp ồ ạt) bằng cách sử dụng các kỹ thuật về an toàn, bảo mật ...)

2.5.3. Các dịch vụ chia sẻ lược đồ

Các dịch vụ nên cung cấp thành phần giao tiếp ra bên ngoài,và hỗ trợ chia sẻ cấu trúc thông tin,ràng buộc dữ liệu thông qua các lược đồ dữ liệu chuẩn (độc lập ngôn ngữ, độc lập hệ nền). Như thế hệ thống sẽ có tính liên kết và khả năng dễ mở rộng.

2.5.4. Tính tương thích của dịch vụ dựa trên chính sách

Một dịch vụ khi muốn tương tác với một dịch vụ khác thì phải thỏa mãn các chính sách và yêu cầu của dịch vụ đó như là mã hóa, bảo mật... Để thực hiện điều này, mỗi dịch vụ cần phải cung cấp công khai các yêu cầu, chính sách đó.

2.6. Các tính chất của một hệ thống SOA

2.6.1. Loose coupling (kết nối “lỏng”)

Vấn đề kết nối ám chỉ đến một số ràng buộc giữa các mô đun với nhau. Có hai loại kết nối là rời và chặt. Các mô đun kết nối lỏng có một số ràng buộc được mô tả trong khi các mô đun kết nối chặt lại có nhiều ràng buộc không thể biết trước. Hầu như mọi kiến trúc phần mềm đều hướng đến tính kết nối lỏng giữa các mô đun. Mức độ kết dính của mỗi hệ thống ảnh hưởng trực tiếp đến khả năng chỉnh sửa hệ thống của chính nó.Mức độ kết nối tăng dần khi bên sử dụng dịch vụ cần biết thông tin ngầm định của bên cung cấp dịch vụ được cung cấp.Ngược lại, nếu bên sử dụng dịch vụ không cần biết thông tin chi tiết trước khi triệu gọi thì quan hệ giữa hai bên càng có tính lỏng.SOA hỗ trợ kết nối lỏng thông qua việc sử dụng hợp đồng và liên kết.

Kết nối lỏng hỗ trợ gỡ bỏ ràng buộc điều khiển giữa những hệ thống đầu cuối. Mỗi hệ thống có thể tự quản lý độc lập nhằm tăng hiệu suất, khả năng mở rộng và khả năng đáp ứng cao.Kết nối lỏng đem đến sự độc lập giữa bên cung cấp và bên sử dụng nhưng nó đòi hỏi các giáo diện phải theo chuẩn và một thành phần trung gian quản lý, trung chuyển yêu cầu giữa các hệ thống đầu cuối.

2.6.2. Sử dụng lại dịch vụ

Bởi vì các dịch vụ được cung cấp lên trên mạng và được đăng ký ở một nơi nhất định nên chúng dễ dàng được tìm thấy và tái sử dụng.Các dịch vụ có thể được tái sử dụng lại bằng cách kết hợp lại với nhau theo nhiều mục đích khác nhau. Tái sử dụng lại các dịch vụ còn giúp loại bỏ những thành phần trùng lặp và tăng độ vững chắc trong cài đặt, nó còn giúp đơn giản hoá việc quản trị.

2.6.3. Sử dụng dịch vụ bất đồng bộ

Trong phương thức triệu gọi dịch vụ bất đồng bộ, bên gọi gửi một thông điệp với đầy đủ thông tin ngữ cảnh tới bên nhận. Bên nhận xử lý thông tin và trả kết quả về thông qua một "kênh thông điệp", bên gọi không phải chờ cho đến khi thông điệp được xử lý xong.Do bên gọi không phải chờ cho đến khi yêu cầu được xử lý xong và trả về nên không bị ảnh hưởng bởi việc xử lý trễ và lỗi khi thực thi các dịch vụ bất đồng bộ. Trên lý thuyết hệ thống SOA có thể gửi và nhận cả thông điệp đồng bộ và bất đồng bộ.

2.6.4. Quản lý các chính sách

Khi sử dụng các dịch vụ chia sẻ trên mạng, tùy theo mỗi ứng dụng sẽ có một luật kết hợp riêng gọi là chính sách và thiết kế tách biệt. Nếu không sử dụng chính sách, nhân viên phát triển phần mềm, nhóm điều hành và hỗ trợ phải làm việc với nhau trong thời gian để cài đặt và kiểm tra những chính sách. Ngược lại, nếu sử dụng chính sách, những nhân viên phát triển phần mềm giờ chỉ cần tập trung vào quy trình nghiệp vụ trong khi nhóm điều hành và nhóm hỗ trợ tập trung vào các luật kết hợp.

2.6.5. Khả năng cộng tác

SOA nhấn mạnh đến khả năng cộng tác giữa các hệ thống khác nhau. Mỗi dịch vụ cung cấp một giao diện có thể được triệu gọi thông qua một dạng kết nối.

2.6.6. Tự động dò tìm và ràng buộc động

SOA hỗ trợ khái niệm dò tìm dịch vụ. Người sử dụng cần đến một dịch vụ nào đó có thể tìm kiếm dịch vụ dựa trên một số tiêu chuẩn khi cần.Người sử dụng chỉ cần hỏi một registry về dịch vụ nào thoả yêu cầu tìm kiếm. Mối ràng buộc duy nhất giữa bên cung cấp và bên sử dụng là bản hợp đồng được cung cấp bởi *registry* trung gian. Mối ràng buộc này là ràng buộc trong thời gian chạy chứ không phải ràng buộc trong lúc biên dịch.Với SOA, bên sử dụng dịch vụ không cần biết định dạng của thông điệp yêu cầu và thông điệp trả về,cũng như địa chỉ dịch vụ cho đến khi cần

2.6.7. Tự hồi phục

Với quy mô và độ phức tạp của những ứng dụng phân tán ngày nay, khả năng phục hồi của một hệ thống sau khi bị lỗi trở thành một yếu tố quan trọng. Một hệ thống có khả năng tự hồi phục sau khi bị lỗi mà không cần sự can thiệp của con người

Độ tin cậy là mức độ đo khả năng một hệ thống xử lý tốt như thế nào trong tình trạng hỗn loạn. Trong kiến trúc hướng dịch vụ, các dịch vụ luôn có thể hoạt động hay ngừng bất kỳ lúc nào, nhất là đối với những ứng dụng tổng hợp từ những từ nhiều dịch vụ của nhiều tổ chức khác nhau. Độ tin cậy phụ thuộc vào khả năng phụ hồi của phần cứng sau khi bị lỗi. Một khía cạnh khác ảnh hưởng đến độ tin cậy là kiến trúc mà dựa trên đó ứng dụng được xây dựng. Một kiến trúc hỗ trợ kết nối và thực thi động khi chạy sẽ có khả năng tự phục hồi hơn một hệ thống không hỗ trợ những tính năng trên.

2.7. Lợi ích khi sử dụng SOA

Lợi ích kinh tế:

- Doanh nghiệp có thể tập trung tìm kiếm các giải pháp cho bài toán liên quan đến kinh tế. Thúc đẩy sự phát triển của hệ thống và mở rộng trong tương lai

Lợi ích kỹ thuật:

- Hệ thống sẽ đảm bảo các dịch vụ có tính độc lập cao (độ kết dính thấp) .
- Việc di dời các dịch vụ đến một máy tính khác không ảnh hưởng khả năng phục vụ yêu cầu khách hàng.
- Tính kết nối lỏng giúp tăng tính linh hoạt và khả năng triển khai cài đặt.
- Tăng khả năng mở rộng và khả năng sẵn sàng cung cấp bằng cách thêm nhiều thể hiện của một dịch vụ. Công nghệ chia tải sẽ tự động tìm và định tuyến yêu cầu đến dịch vụ thích hợp. SOA có thể chuyển tiếp nội dung yêu cầu đến một thể hiện khác khi cần, nhờ đó tăng khả năng sẵn sàng phục vụ
- Hỗ trợ đa thiết bị và đa nền tảng

2.8. Một số mô hình triển khai SOA

2.8.1. Service Registry

Đây là mô hình truyền thống để định vị và liên kết các dịch vụ trong một hệ thống SOA. Mô hình này về cơ bản chỉ cần các chuẩn Web services thông thường là SOAP, WSD và UDDI. Các liên kết dịch vụ trong mô hình là kết nối tĩnh và phải định nghĩa trong thiết kế, điều này làm cho mô hình trở nên cứng nhắc. Có một cách cải tiến làm cho mô hình này linh hoạt hơn là tìm kiếm, định vị các dịch vụ khi chạy. UDDI hỗ trợ nhiều cấu hình khác nhau cho cùng một dịch vụ cung cấp bởi nhiều nhà cung cấp dịch vụ khác nhau.

2.8.2. Service broker

Trong mô hình cơ bản, tất cả những thông điệp đều được trung chuyển qua Service broker. Dịch vụ này có thể làm nhiều chức năng như định tuyến dựa trên dữ liệu thông điệp, xử lý lỗi, chuyển đổi thông điệp, chia tải và lọc thông tin. Nó cũng có thể cung cấp dịch vụ bảo mật, chuyển đổi giao thức, lưu vết .Tuy nhiên, Service broker có thể xảy ra hiện tượng nghẽn cổ chai và là điểm dễ bị hỏng hóc. Mô hình broker phân tán là một bước cải tiến mới, ở đó mỗi nền tảng dịch vụ có một Broker cục bộ cho phép giao tiếp với một Service broker trung tâm và giao tiếp trực tiếp với các Service broker cùng cấp ở các nền tảng dịch vụ khác

2.8.3. Service bus

Đây là mô hình ra đời sau nhất trong ba mô hình nhưng nó đã được sử dụng trong các sản phẩm thương mại lớn(như IBM, BEA). Service bus cũng là mô hình có tính kết nối lỏng nhất trong các mô hình, trong đó các dịch vụ không kết nối trực tiếp với nhau thành một mạng Service bus

2.9. Kiến trúc phân tầng chi tiết của SOA

2.9.1. Tầng kết nối

Mục đích là kết nối đến các ứng dụng enterprise hoặc tài nguyên bên dưới và cung cấp chúng thành dạng những dịch vụ. Tầng này là tầng chuyên giao tiếp với các nhà cung cấp, hoạt động như một bộ chuyển đổi giữa các ứng dụng phi dịch vụ và mạng các dịch vụ khác.

Tầng này thực hiện kết nối đến các hệ cơ sở dữ liệu.

2.9.2. Tầng orchestration

Tầng orchestration chứa các thành phần đóng vai trò vừa là dịch vụ sử dụng vừa là dịch vụ cung cấp và sử dụng những dịch vụ của tầng kết nối và các dịch vụ orchestration khác để kết hợp những chức năng cấp thấp hơn thành những dịch vụ hoạt động ở cấp cao hơn, có hành vi gần với những chức năng nghiệp vụ thực tế hơn.

2.9.3. Tầng ứng dụng tổng hợp

Dữ liệu truyền qua lại giữa những dịch vụ cuối cùng cũng định hướng đến người sử dụng theo nhiều dạng giao diện khác nhau. Tầng này được xem là tầng tích hợp cuối cùng của quá trình tích hợp.

Tầng này đơn thuần sử dụng các dịch vụ, nó cung cấp các ứng dụng cho người dùng cuối. Nhờ tính linh hoạt của SOA và đặc tính của các dịch vụ được tổng hợp từ tầng orchestration, các ứng dụng tổng hợp có khả năng biểu diễn mọi loại thông tin từ mọi nguồn thông tin.

Tầng ứng dụng tổng hợp chia làm hai tầng nhỏ hơn là Portal và tầng Portlet:

- Portlet là thành phần cung cấp và sử dụng dịch vụ.Và sử dụng một số dịch vụ liên quan của tầng orchestration bên dưới và cho phép người sử dụng gửi thông tin bổ sung
- Portal là một bộ khung tích hợp sử dụng các Porlet, trang bị cho chúng vẻ ngoài thống nhất và thể hiện thành một giao diện hoàn chỉnh cho người dùng cuối.

⇨ Tóm lại:

Hệ thống SOA trở nên độc lập với các nền tảng. Các dịch vụ hoạt động trên các nền tảng khác nhau vẫn có thể giao tiếp với nhau nhờ vào các giao diện giao tiếp đã được chuẩn hóa để cộng tác xử lý một tác vụ nào đó.

2.10. Kiến trúc bảo mật hướng dịch vụ SOSA

Kiến trúc hướng dịch vụ (SOA) là một tập hợp các qui tắc cho việc thiết kế các dịch vụ có tính dễ mở rộng, khả năng kết hợp và tương tác cao. Các nguyên tắc này không chỉ có thể áp dụng cho các dịch vụ nghiệp vụ để hình thành các hệ thống nghiệp vụ SOA mà có thể dùng cho các dịch vụ bảo mật để tạo nên các hệ thống bảo mật SOA, cho các dịch vụ quản lý để xây dựng các hệ thống quản lý SOA …

Mô hình SOSA không hướng đến việc thay thế hoàn toàn các kiến trúc bảo mật hiện có, mà muốn đưa ra một giải pháp để liên kết các cơ sở hạ tầng có sẵn. Thay vào đó là tái sử dụng lại (chứ không phải thay mới) những kỹ thuật, dịch vụ bảo mật hiện có dựa trên những nguyên tắc của SOA để tạo nên một kiến trúc bảo mật hướng dịch vụ mới. Đây cũng là mục tiêu chính của các chuẩn mở về XML và Web Service mà đã và đang được phát triển: "không thay thế những gì đã có, mà làm cho chúng liên kết với nhau trong một môi trường đồng nhất".

Yếu tố đầu tiên và quan trọng nhất mà ta cần phải quan tâm đến khi thiết kế tầng liên kết này đó là "*thiết lập được sự tin cậy*": theo định nghĩa của tổ chức OASIS (Organization for the Advancement oì Structured Information Standards) thì "*sự tin cậy*" là cơ sở cho một thực thể khác dựa trên đó để thực hiện một số hành động hay xác nhận về đối tượng đó.

Mục tiêu cuối cùng của kiến trúc bảo mật hướng dịch vụ đó là xây dựng được các sự tin cậy giữa các dịch vụ với nhau bằng cách thiết lập và thi hành các chính sách về bảo mật

Định nghĩa của những dấu hiệu mật và chính sách là:

- Policy là một tập hợp các cơ chế xác nhận các chính sách.
- Cơ chế xác nhận (Policy Assertion) về hoạt động hệ thống
- Security Token: Security Token có thể là dạng binary (X.509, Kerberos ticket) hay XML (SAML, XrML).

Nói cách khác, hệ thống SOSA sẽ phải xây dựng được những yếu tố cơ bản sau:

- Các nghi thức chung dùng trong việc trao đổi các thẻ của các đối tượng sử dụng dịch vụ.
- Các nguyên tắc chung về cách xử lý các thẻ đó của phía cung cấp dịch vụ.
- Nếu thiết lập được cơ chế quản lý các mối liên kết giữa các đối tượng sử dụng dịch vụ và nhà cung cấp dịch vụ thì hệ thống sẽ vận hành một cách linh hoạt hơn, đặc biệt là trong bối cảnh các đối tượng trên phân tán trong những tổ chức, với những chính sách và cơ chế xử lý thẻ đặc thù

CHƯƠNG 3: WEB SERVICE

3.1. Giới thiệu về Service

3.1.1. Khái niệm

Theo IBM: "*Service is a repeatable task within a business process*"[5]. Theo đó, Service là một ứng dụng với người dùng, một thao tác được thực hiện một hoặc nhiều lần trong một tiến trình và được thực hiện bởi một hay nhiều người.

Service là một hệ thống có khả năng nhận một hay nhiều yêu cầu xử lý và sau đó đáp ứng lại bằng cách trả về một hay nhiều kết quả. Quá trình nhận yêu cầu và trả kết quả về được thực hiện thông qua các giao diện đã được định nghĩa trước đó. Thông thường việc giao tiếp này được thực hiện trên các giao diện đã được chuẩn hóa và sử dụng rộng rãi.

Một hệ thống được thiết kế theo kiểu hướng Service là một hệ thống trong đó các chức năng của hệ thống được xây dựng dựa trên các service có độ kết dính thấp. Các service trong hệ thống giao tiếp với nhau thông qua việc gởi nhận các thông điệp.

3.1.2. Các đặc điểm chính của Service

Mỗi service được xây dựng dựa trên các giao diện chuẩn hóa đã được sử dụng rộng rãi. Chi tiết hiện thực của mỗi service sẽ không được thể hiện ra bên ngoài. Mỗi service chỉ công bố một số các giao diện của nó cho user có thể dùng để gởi các yêu cầu và nhận kết quả trả về.

Mỗi Service có tính độc lập cao, có thể được xây dựng và đưa vào sử dụng mà không phụ thuộc vào các service khác.

Trao đổi dữ liệu: các Service không truyền các *class* và *type*.Thay vào đó, các *class* và *type* sẽ được đặc tả hình thức.

3.2. Tổng quan về Web Service

3.2.1. Khái niệm Web Service

Web Service là một giao diện truy cập mạng đến các ứng dụng chức năng, được xây dựng từ việc sử dụng các công nghệ chuẩn Internet[5].

Thuật ngữ Web Service diễn tả một cách thức tích hợp các ứng dụng trên nền website lại với nhau bằng cách sử dụng các công nghệ XML, SOAP, WSDL,UDDI trên nền tảng các giao thức Internet với mục tiêu tích hợp ứng dụng và truyền thông điệp. XML được sử dụng để đánh dấu dữ liệu, SOAP được dùng để truyền dữ liệu,

WSDL được sử dụng để mô tả các dịch vụ có sẵn và UDDI được sử dụng để liệt kê những dịch vụ nào hiện tại đang có sẵn để có thể sử dụng. Web Service cho phép các tổ chức có thể trao đổi dữ liệu với nhau mà không cần phải có kiến thức hiểu biết về hệ thống thông tin đứng sau Firewall kia.

Không giống như mô hình khách/chủ truyền thống,Web Service không cung cấp cho người dùng một giao diện đồ hoạ nào, Web Service đơn thuần chỉ là việc chia sẻ các dữ liệu logic và xử lý các dữ liệu đó thông qua một giao diện chương trình ứng dụng được cài đặt xuyên suốt trên mạng máy tính.

Web Service cho phép các ứng dụng khác nhau từ các nguồn khác nhau có thể giao tiếp với các ứng dụng khác mà không đòi hỏi nhiều thời gian lập trình, do tất cả các quá trình giao tiếp đều tuân theo định dạng XML, cho nên Web Service không bị phụ thuộc vào bất kì hệ điều hành hay ngôn ngữ lập trình nào.

Web Service cung cấp tính trừu tượng cho các giao diện chuẩn, cho nên sẽ không nảy sinh ra bất kì vấn đề gì trong quá trình tương tác. Web Service cho phép giao tiếp giữa các nền tảng khác nhau có thể hoạt động cùng nhau theo nguyên tắc tạo ra một nền tảng trung gian có liên quan.

⇨ Tóm lại: Web Service là:

- Làm việc xuyên qua tường lửa và proxy
- Sẵn sàng đối với các nền tảng máy trạm khác nhau
- Một dịch vụ phần mềm được trình bày trên web thông qua giao thức SOAP, được mô tả bằng một tệp WSDL và được đăng ký trên UDDI.

3.2.2. Đặc điểm Web Service

Cho phép khách/chủ tương tác với nhau cả trong môi trường khác nhau.

XML và HTTP là nền tảng kỹ thuật chính. Phần lớn kỹ thuật của Web Service được xây dựng là những dự án nguồn mở cho nên độc lập và vận hành được với nhau

Web Service rất linh động: với UDDI và WSDL thì việc mô tả và phát triển Web Service có thể tự động hóa.

Web Service bao gồm nhiều mô đun và có thể công bố trên mạng Internet.

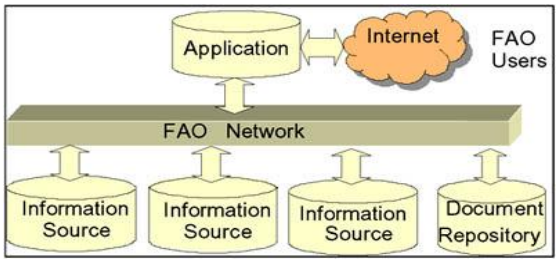
Web Service có thể chia sẻ và gọi thực hiện qua mạng và có độ an toàn riêng tư.

3.3. Một số mô hình áp dụng Web Service

3.3.1. Sử dụng để tương hợp dữ liệu tại FAO

Chức năng quan trọng nhất của FAO: thu thập,phân tích,đánh giá các thông tin hỗ trợ các chính phủ chống đói nghèo và đạt được an ninh lương thực.

Trong WAICENT, một lượng đồ sộ các dữ liệu được trình bày ở các định dạng hoàn toàn khác nhau, trên nhiều ngôn ngữ,không có các tiêu chuẩn cho việc trình bày. Điều quan trọng là chia sẻ dữ liệu giữa các hệ thống nhanh chóng và dễ dàng,các hệ thống đang tồn tại cần phải "nói chuyện" được với nhau. Bên trong tổ chức sử dụng hai công nghệ khác nhau (ASP và Java JSP/servlet) và không có tiêu chuẩn nào để quản lý các phương án ngôn ngữ văn bản hoặc kiến trúc dữ liệu[1].



Hình 3.1: Mô hình kết nối CSDL của FAO

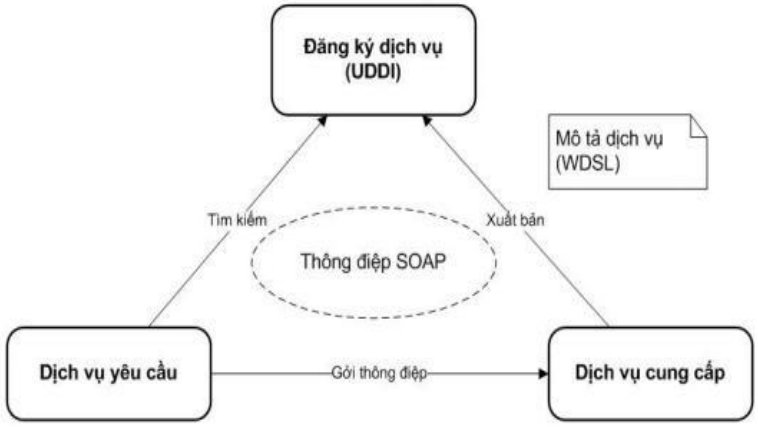
Đòi hỏi công nghệ cần hỗ trợ tính tương hợp các nguồn dữ liệu sẵn có và quản lý các phương án đa ngôn ngữ mà không phải thay đổi kiến trúc của cơ sở dữ liệu. Để khắc phục các vấn đề này,FAO đưa ra một tiếp cận dựa trên các công nghệ Web Service, XML. Mục tiêu chính: Tạo ra môi trường nơi mà các hệ thống thông tin mới dựa trên web có thể được phát triển nhanh chóng và dễ dàng,sử dụng bất cứ nền tảng công nghệ nào. Triển khai các máy tạo và phát triển báo cáo động của một kho văn bản XML để quản lý siêu dữ liệu và các phương án về ngôn ngữ theo một cách chung.

3.3.2. Sử dụng Web Service trong công nghệ di động

Hãng T-Mobile đặt niềm tin vào Web Service như một nền tảng cho việc quảng bá dữ liệu tới các khách hàng và nhân viên sử dụng di động. Hãng có phần mềm trung gian với khoảng 50-60 Web Services được tích hợp bên trong cho phép tích hợp các dịch vụ của T-Mobile với nhau như nhận dạng,cá nhân hóa và thanh toán hóa đơn,với các dịch vụ quảng bá nội dung thông tin cho máy di động cho người sử dụng được 250 đối tác cung cấp. Không cần phải quan tâm tới việc các hệ thống là Microsoft, Linux…, miễn là đầu ra ở dạng XML và sử dụng SOAP là được[1].

3.4. Mô hình Web Service, ưu và nhược điểm

3.4.1. Mô hình Web Service



Hình 3.2: Mô hình Web Service

Nhà cung cấp đăng ký Web Service với UDDI.

Người sử dụng tìm kiếm dịch vụ trên UDDI qua một URL thích hợp.

UDDI trả lại một bản mô tả WSDL cho nhà cung cấp.

Người sử dụng triệu gọi dịch vụ bằng một cuộc gọi SOAP tới nhà cung cấp

Nhà cung cấp trả lại kết quả của cuộc gọi SOAP cho người sử dụng

3.4.2. Ưu điểm

Cho phép chương trình được viết bằng các ngôn ngữ khác nhau trên các nền tảng khác nhau giao tiếp được với nhau dựa trên một nền tảng tiêu chuẩn

Đơn giản (chỉ dùng URL)

Làm việc với các giao thức chuẩn Web như XML, HTTP và TCP/IP.

Sự an toàn của máy chủ cơ sở dữ liệu luôn được bảo mật một cách chắc chắn.

Web Service làm giảm giá thành cho việc tích hợp các hệ thống khác nhau.

3.4.3. Nhược điểm

Phụ thuộc vào tốc độ đường truyền Internet.

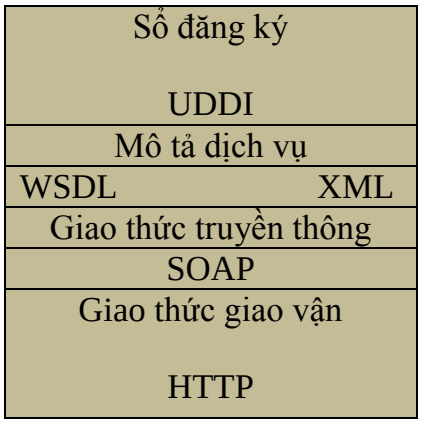
Web Service thiếu cơ chế khôi phục đủ tin cậy để đảm bảo giao dịch được khôi phục lại trạng thái ban đầu trong trường hợp xảy ra sự cố

Số lượng các ứng dụng cộng tác cùng hoạt động sẽ ảnh hưởng tới hiệu suất tối ưu của Web Service.

Tải trọng: ứng dụng Web Service là các ứng dụng sử dụng rất nhiều thông điệp. Khả năng bùng nổ số lượng giao dịch trao đổi sẽ làm hệ thống máy chủ ứng dụng và kiến trúc hạ tầng hệ thống thông tin của doanh nghiệp trở nên ngưng trệ.

Vì Web Service đòi hỏi kết nối thông qua khá nhiều máy chủ trung gian cho nên băng thông/tốc độ của hạ tầng mạng và các yêu tố liên quan tới hệ thống rõ ràng có vai trò quan trọng góp phần cải thiện hiệu năng của toàn bộ các ứng dụng WS

3.5. Các thành phần chính của Web Service



Hình 3.3: Các thành phần chính của Web Service

XML được sử dụng để định dạng dữ liệu, SOAP được sử dụng trao đổi dữ liệu, WSDL được sử dụng để mô tả dịch vụ hiện có và UDDI được sử dụng để liệt kê các Web Service hiện có.

3.5.1. Giao thức giao vận HTTP

3.5.1.1: Giao thức HTTP

Tầng giao vận liên quan tới cơ chế sử dụng để chuyển yêu cầu dịch vụ và thông tin phản hồi từ phía nhà cung cấp dịch vụ tới người sử dụng dịch vụ. Có rất nhiều tiêu chuẩn sử dụng xung quanh WS, nhưng phổ biến nhất vẫn là giao thức HTTP.

Giao thức HTTP thường được sử dụng đối với yêu cầu dịch vụ và đáp ứng.

3.5.1.2. Ưu điểm

HTTP là nền tảng hạ tầng phổ biến và sẵn sàng nhất.

Giao thức HTTP hoàn toàn mở và khai triển trên rất nhiều loại hệ thống

Hầu hết mọi tổ chức đều chấp nhận cho phép trao đổi thông tin dựa trên giao thức HTTP vượt qua tường lửa bảo vệ.

3.5.1.3. Nhược điểm

HTTP là một giao thức đơn giản và không có tính trạng thái, không được thiết kế đặc biệt cho mục đích vận chuyển dữ liệu của các ứng dụng.

Giao thức không hỗ trợ lưu trữ trạng thái

Không phải là một giao thức đáng tin cậy phù hợp với nhu cầu truyền dữ liệu.

3.5.2. Giao thức truyền thông SOAP

3.5.2.1. Khái niệm

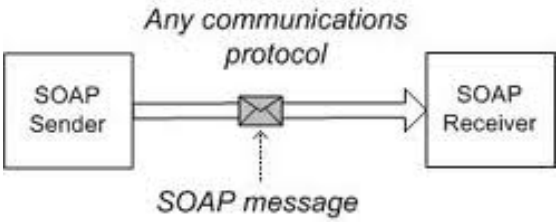
SOAP là gì:

- SOAP là giao thức truyền thông giữa các ứng dụng.
- SOAP được thiết kế để liên lạc qua Internet và làm việc qua tường lửa.
- SOAP độc lập nền tảng, độc lập ngôn ngữ.
- SOAP dựa trên XML, đơn giản và dễ mở rộng.

SOAP có đặc trưng:

- SOAP được thiết kế đơn giản và dễ mở rộng
- Tất cả các message SOAP đều được mã hóa sử dụng XML.
- SOAP sử dụng giao thức truyền dữ liệu riêng.
- SOAP không bị ràng buộc bởi ngôn ngữ lập trình hoặc công nghệ nào
- SOAP không quan tâm đến công nghệ gì được sử dụng để thực hiện miễn là người dùng sử dụng các message theo định dạng XML.

⇨ Tóm lại: SOAP là giao thức mà định nghĩa cái cách để chuyển một XML message từ A đến B dựa trên giao thức chuẩn web HTTP (hoạt động trên cổng 80) qua giao thức Internet TCP/IP.



Hình 3.4: Thông điệp SOAP

Tại sao phải có SOAP:

- Phát triển các ứng dụng cho phép các chương trình trao đổi qua Internet.
- Các ứng dụng liên lạc với nhau bằng cách sử dụng các cuộc gọi thủ tục ở xa giữa các đối tượng như DCOM,CORBA
- SOAP cung cấp cách để liên lạc giữa các ứng dụng chạy trên các hệ điều hành khác nhau,với các công nghệ khác nhau và ngôn ngữ khác nhau.

3.5.2.2. Định dạng thông điệp

Một thông điệp SOAP là một văn bản XML được mô tả bởi một thành phần Envelop,chứa một thành phần Body bắt buộc và một thành phần Header không bắt buộc. Thành phần Body có thể chứa một số Body Entries. Thành phần không bắt buộc Fault chỉ có trong thông điệp khi có báo cáo về một quá trình xử lý ngoại lệ.

Phần tử Body mô tả về phương thức dưới dạng XML và chỉ chứa các tham số hay các trường dưới dạng các thẻ.

Với Document người phát triển phải xử lý gần như là toàn bộ, họ phải đưa ra một loạt các tham số dưới dạng các thẻ XML

3.5.2.3. Mã hóa thông điệp

Dữ liệu được mã hoá và gói vào trong phần tử Body của một thông điệp và được gửi đến Host. Host giải mã dữ liệu được định dạng XML về dạng đối tượng ban đầu.

SOAP Remote Procedure Call (RPC encoding): Là kiểu mã hóa đơn giản nhất cho người phát triển. Bạn gọi tới một đối tượng từ xa, kèm theo là các tham số cần thiết. Các tham số được chuyển lần lượt dưới dạng XML và truyền đến đích sử dụng giao thức giao vận như HTTP hay SMTP. Sau khi nhận được, dữ liệu được chuyển trở lại thành dạng đối tượng và kết quả được trả về cho phương thức gọi. SOAP RPC xử lý tất cả công việc mã hóa và giải mã,thậm chí đối với các kiểu dữ liệu phức tạp.

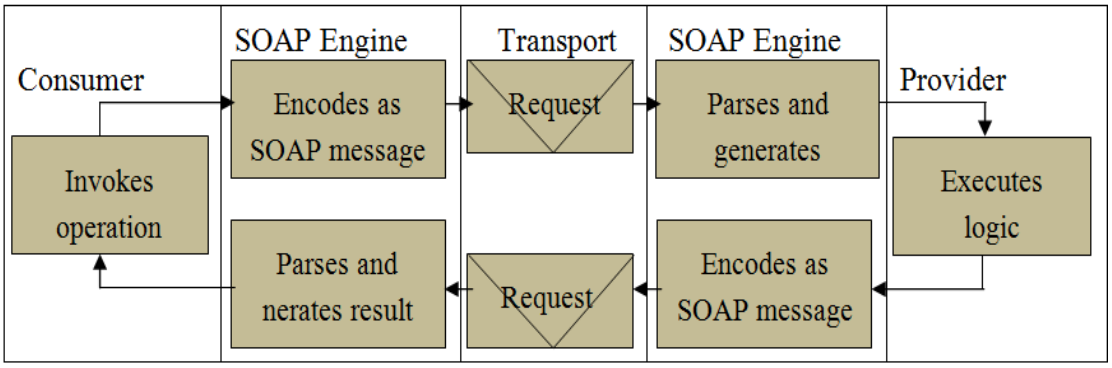
SOAP Remote Procedure Call Literal encoding (SOAP RPC-literal): Sử dụng một dạng thức mã hóa do người sử dụng chỉ định để mã và giải mã dữ liệu dạng XML.

SOAP document-style encoding: Toàn bộ XML được gửi đến máy chủ và người lập trình xác định giao thức giao vận,phân tích dữ liệu dạng XML ở thông điệp yêu cầu và đáp ứng để tìm dữ liệu cần thiết.

3.5.2.4. Quá trình xử lý thông điệp

Một thông điệp SOAP giúp cho khách hàng và nơi cung cấp Web Service hoàn thành những tác vụ mà không lo lắng đến sự phức tạp của việc xử lý thông điệp SOAP.

Một processor của khách hàng chuyển các lời yêu cầu phương thức vào trong một thông điệp SOAP.Thông điệp này được truyền qua tầng giao vận (HTTP và SMTP) tới processor của nơi cung cấp,tại đây thông điệp sẽ được phân tích thành lời yêu cầu phương thức.Sau đó nơi cung cấp sẽ thực hiện những bước logic cần thiết và trả lại kết quả cho processor của nó,processor này sẽ phân tích thông tin trong thông điệp hồi đáp.Thông điệp này được truyền qua tầng giao vận tới khách hàng yêu cầu. Processor của nó phân tích thông điệp hồi đáp thành kết quả dưới dạng một đối tượng.



Hình 3.5: Quá trình xử lý thông điệp SOA

3.5.3. Ngôn ngữ đánh dấu,mở rộng XML

3.5.3.1. Khái niệm XML

XML là nền tảng của Web Service và được dùng để trao đổi dữ liệu.

XML là một chuẩn nổi tiếng cho việc tổ chức,lưu trữ và trao đổi dữ liệu.

XML được hỗ trợ bởi hầu hết các ngôn ngữ lập trình hiện đại (DotNet, Java…)

XML được sử dụng rộng rãi trong việc trao đổi dữ liệu trên môi trường Internet.

XML dùng các thẻ để tổ chức và lưu trữ dữ liệu

3.5.3.2. Đặc điểm của XML

XML là tự do và mở rộng được. Trong XML các thẻ không được định nghĩa trước mà do người dùng tự phát minh ra thẻ.

XML rất quan trọng đối với sự phát triển của web trong tương lai. XML sẽ là công cụ xử lý và truyền dữ liệu phổ biến nhất.

XML là công cụ dùng được trên mọi nền phần cứng, độc lập với phần cứng và phần mềm để truyền (trao đổi,chia sẻ) thông tin.

3.5.3.3. XML được sử dụng như thế nào

XML được thiết kế để lưu trữ và trao đổi dữ liệu nhưng không hiển thị dữ liệu.

XML có thể trao đổi dữ liệu giữa các hệ thống không tương thích.

3.5.3.4. Cấu trúc tài liệu XML

- XML hợp khuôn dạng: khai báo XML và dữ liệu XML
- XML hợp lệ: Là tài liệu được kết hợp với định nghĩa kiểu tư liệu (Document Type Definition) và tuân theo tiêu chuẩn đó

3.5.3.5. Quy tắc cú pháp ngôn ngữ XML

Các khai báo XML cần được đặt ở dòng đầu tiên của tài liệu

Mọi phần tử XML đều phải có thẻ đóng: />

Tất cả các tài liệu XML phải có thẻ gốc trong đó thẻ đầu tiên là thẻ gốc.

Các thẻ XML phân biệt hoa_thường và khoảng trắng được giữ lại

Các giá trị thuộc tính phải luôn đặt trong ngoặc kép

3.5.3.6. Ưu điểm của XML

Đơn giản, ổn định, linh hoạt và có tính mở rộng cao

XML được chấp nhận rộng rãi. Rất nhiều công cụ và tiện ích sẵn có đáp ứng nhu cầu phân tích và chuyển đổi dữ liệu XML hoặc hiển thị chúng.

3.5.3.7. Nhược điểm của XML

Sự phức tạp

Việc chuẩn hóa

Dung lượng lớn

3.5.4. Ngôn ngữ mô tả dịch vụ WSDL

3.5.4.1. Khái niệm

WSDL là ngôn ngữ dựa trên XML và mô tả cách thức truy cập Web Service

WSDL thường được sử dụng với SOAP và cấu trúc XML để cung cấp Web Service qua Internet. Một máy khách kết nối tới Web Service có thể đọc WSDL để xác định hàm nào hiện đang có trên máy chủ. Khách có thể sử dụng SOAP để gọi một trong nhiều hàm được liệt kê trong WSDL.

⇨ Tóm lại:

WSDL mô tả Web Service theo cú pháp tổng quát XML, bao gồm các thông tin: tên service, giao thức và kiểu mã hoá, tham số, kiểu dữ liệu …

WSDL chỉ định các đặc tính vận hành của Web Service. Ngôn ngữ mô tả những khái niệm trả lời cho các câu hỏi sau:

- Cái gì (Web Service làm gì) ?
- Ở đâu (nơi chứa Web Service) ?
- Như thế nào (Web Service có thể kích hoạt bằng cách nào) ?

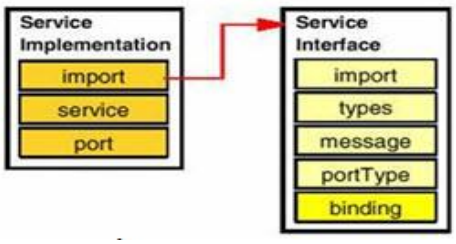
3.5.4.2. Cấu trúc WSDL

Một WSDL hợp lệ gồm có hai phần:

- Phần giao diện mô tả giao diện và giao thức kết nối.
- Phần thi hành mô tả thông tin để truy xuất service.

Cả 2 thành phần này sẽ được lưu trong hai tập tin XML, bao gồm:

- Tập tin giao diện service (cho phần 1)
- Tập tin thi hành service (cho phần 2)



Hình 3.6: Service Interface và Service Implementation

3.5.4.3. Tập tin giao diện – Service Interface

<*PortType*> : các phép toán được thực hiện bởi Web Service.

<*Message*> : mô tả thông điệp được gửi giữa Máy khách và Máy chủ.

<*Types*> : kiểu dữ liệu được sử dụng bởi Web Service.

<*Binding*> : các giao thức truyền thông được sử dụng bởi Web Service.

- *Types:* Định nghĩa loại dữ liệu được sử dụng bởi các WebService.

```
<wsdl:types>
<xsd:schema .../">
</wsdl:types>
```

- *Message*

Mỗi message được thiết kế phục vụ cho việc truyền hoặc nhận thông tin. Mỗi message có thể bao gồm một hoặc nhiều phần. Mỗi phần có thể được so sánh với thông số của một chức năng gọi trong ngôn ngữ lập trình truyền thống

```
<message name = "runtoken"> *
<part name = "nmtoken" element="qname"? type="qname"?/> *
</message>
```

- *PortType*

Các phần tử <*PortType*> là thành phần quan trọng nhất của WSDL.

Nó định nghĩa một Web Service, các tác vụ mà service cung cấp và định dạng các thông điệp được sử dụng để khởi động các tác vụ này

```
<wsdl: portType name="nmtoken"> *
<wsdl: operation name="nmtoken" .../> *
</wsdl: portType>
```

- *Binding*

<*binding*> dùng để định dạng thông điệp và chi tiết giao thức của mỗi *portType* và được gán 1 giao thức truyền tin để có thể truy xuất và tương tác với WSDL.

Binding chứa 1 hoặc nhiều hoạt động (operation)

Một binding gồm 2 thuộc tính: tên(name) và loại(type).

- Name: định nghĩa tên của binding.
- Type: chỉ ra cổng cho binding.

3.5.4.4. Tập tin thi hành – Service Implementation

- *Port*

Một <*port*> định nghĩa một thiết bị đầu cuối bằng cách chỉ định một địa chỉ duy nhất cho binding. Mỗi port có 2 thuộc tính:name và binding.

Name: cung cấp một cái tên duy nhất trong tất cả các port

Binding: chỉ các *binding* đang được sử dụng luật liên kết, xác định bởi WSDL

- *Service*

Các phần tử <*service*> định nghĩa các cổng hỗ trợ bởi Web Service.

3.5.4.5. Ưu điểm của WSDL

Như một yêu cầu cơ bản đối với ứng dụng của bất cứ Web Service,WSDL là yêu cầu bắt buộc đáp ứng nhu cầu công bố giao tiếp và thỏa thuận cho các dịch vụ khác kích hoạt.

3.5.4.6. Nhược điểm của WSDL

Tài liệu không cung cấp một số thông tin người sử dụng có nhu cầu như :

- Ai cung cấp dịch vụ ?
- Loại hình kinh doanh cung cấp dịch vụ ?
- Các dịch vụ khác cùng do nhà cung cấp dịch vụ này cung cấp ?
- Dịch vụ này sẽ cung cấp với chất lượng dịch vụ như thế nào ?
- Đây là dịch vụ miễn phí hay có thu phí ?

3.5.5. Tích hợp mô tả trình bày tổng hợp UDDI

3.5.5.1. Khái niệm

UDDI là một chuẩn công nghiệp cho việc công bố và tìm kiếm thông tin về Web Service. Nó định nghĩa một khung thông tin cho phép bạn mô tả và phân loại tổ chức của bạn, dịch vụ của nó và những chi tiết kỹ thuật về giao diện của Web Service mà bạn trình bày.

UDDI chỉ định cách thức lưu trữ và nhận thông tin về các dịch vụ và đặt biệt là nhà cung cấp dịch vụ cùng với các giao tiếp kỹ thuật

UDDI dựa vào những chuẩn đã có như là ngôn ngữ đánh dấu mở rộng (XML) và giao thức truy cập đối tượng đơn giản (SOAP). Tất cả các cài đặt của UDDI đều hỗ trợ các đặc tả UDDI.

⇨ Tóm lại: Để có thể sử dụng các dịch vụ, trước tiên khách phải tìm dịch vụ, ghi nhận thông tin về cách sử dụng dịch vụ và biết được đối tượng cung cấp dịch vụ. UDDI định nghĩa thành phần cho biết trước các thông tin này để cho phép máy khác truy tìm và nhận lại những thông tin yêu cầu sử dụng Web Service.

3.5.5.2. Đặc điểm của UDDI

UDDI là phần chứa các thông tin của web service, xây dựng trên nền tảng .NET

UDDI được miêu tả bởi ngôn ngữ WSDL và giao tiếp thông qua SOAP

Nhiệm vụ UDDI: tìm đúng dịch vụ và định nghĩa cách kích hoạt dịch vụ

3.5.5.3. Nội dung của thư mục UDDI

Một nội dung thư mục UDDI là một tệp XML mô tả một nghiệp vụ và các dịch vụ nó chào. Nội dung trong UDDI có ba phần:

- White pages: chứa thông tin liên hệ và các định dạng của Web Service. Những thông tin này cho phép đối tượng khác xác định được dịch vụ.
- Yellow pages: chứa thông tin mô tả Web Service. Những thông tin này cho phép các đối tượng thấy được từng loại của Web Service.
- Green pages: chứa thông tin kỹ thuật mô tả các hành vi và các chức năng
- Loại dịch vụ – tModel: chứa các thông tin về loại dịch vụ được sử dụng.

3.5.5.4. Cấu trúc sổ đăng ký UDDI

UDDI cung cấp 4 cấu trúc dữ liệu mô tả dịch vụ mà nó đưa ra: BusinessEntity, BusinessService, BusinessTemplate và tModels.

- BusinessEntity: mô tả nhà cung cấp dịch vụ
- BusinessService: chứa các thông tin chung về dịch vụ
- BindingTemplate: chứa thông tin kỹ thuật cách thức truy cập vào dịch vụ
- tModels (Technical Model- mô hình kỹ thuật): chứa các thông tin về loại Web Service sử dụng. Được sử dụng để lấy thông tin chi tiết về giao diện của Web Service và làm cho chúng có thể sử dụng lại giữa các dịch vụ tương thích.

3.5.5.5. Các kiểu sổ đăng ký UDDI

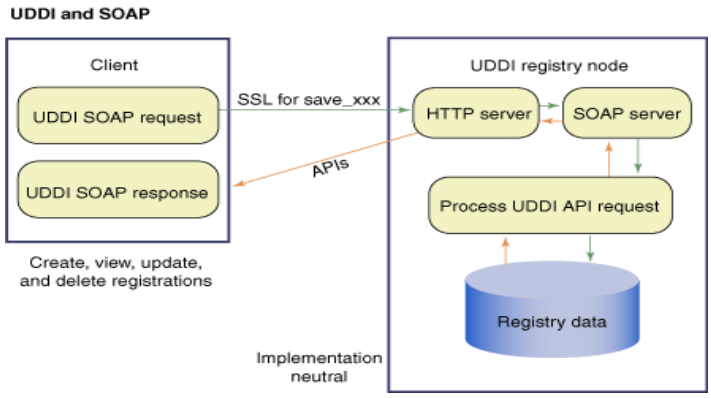
Mô hình đám mây các nút UDDI: một cơ chế khai triển công khai của tiêu chuẩn UDDI đó là Sổ đăng ký kinh doanh UDDI hoặc UBR. UBR bao gồm vài nút UDDI. Những nút này do các công ty như IBM, Microsoft hay SAP và NTT quản lý. Khi một nhà cung cấp dịch vụ muốn công bố dịch vụ của họ, họ sẽ tới một trong các địa chỉ UBR như: http://uddi.ibm.com. Và sau đó đăng ký rồi công bố dịch vụ của họ. Dữ liệu tiếp tục nhân bản tới các nút khác trong cùng hệ thống UBR.

Nhóm hoặc các sổ đăng ký cộng tác: những triển khai này tập trung vào một số lượng cụ thể các đối tác đã từng biết

Sổ đăng ký riêng tư: hầu hết các công ty đều hướng tới việc bắt đầu các dự án Web Service thông qua một sổ đăng ký UDDI riêng biệt

3.5.5.6. UDDI làm việc như thế nào

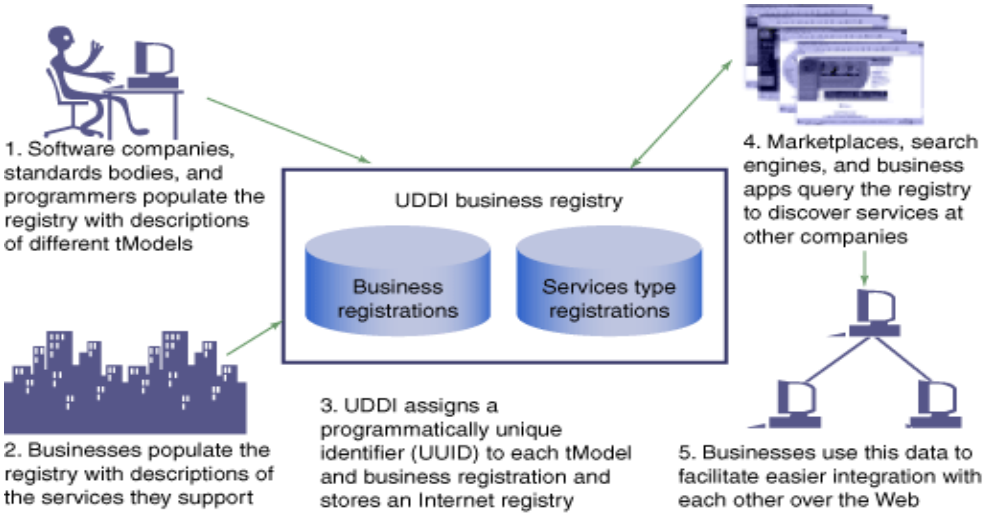
Bản ghi của UDDI chứa mô tả của doanh nghiệp,có thể truy cập bằng máy tính và các dịch vụ mà chúng hỗ trợ. UDDI cung cấp một lược đồ và mô hình lập trình với định nghĩa luật giao tiếp với bản ghi. Tất cả các hàm API trong đặc tả UDDI được định nghĩa trong XML, gói gọn trong một phong bì SOAP, và gửi qua HTTP.



Hình 3.7: Luồng thông báo UDDI giữa Máy khách và Registry

Hình vẽ miêu tả sự truyền tải thông báo UDDI, từ yêu cầu SOAP của máy khách thông qua giao thức HTTP đến một nút bản ghi đăng ký và quay lại. Máy chủ SOAP của hệ thống đăng ký tiếp nhận các thông điệp, xử lý nó, và trả lại một kết quả SOAP đến máy khách. Theo chính sách của bản ghi, các yêu cầu từ máy khách mà bắt buộc phải chỉnh sửa dữ liệu phải là các giao dịch đảm bảo an ninh và được xác thực.

Vậy UDDI làm việc như thế nào?



Hình 3.8: Cách thức làm việc của UDDI

Một bản ghi UDDI được xây dựng trên dữ liệu cung cấp bởi khách hàng của nó. Có vài bước để tạo ra dữ liệu hữu dụng trong UDDI. Như trong bước 1, công bố thông tin hữu ích đến bản ghi bắt đầu khi các công ty phần mềm và các cá nhân định nghĩa các đặc tả liên quan đến công nghiệp hay kinh doanh, mà họ đăng ký với UDDI. Những thứ này được biết như là các mô hình kỹ thuật, hoặc thông dụng hơn là tModels.

Trong bước 2, các công ty cũng đăng ký bản mô tả kinh doanh các dịch vụ của họ. Một bản ghi UDDI sẽ theo dõi tất cả các điểm này bằng cách gán cho mỗi điểm một định danh duy nhất, được biết đến như là một khóa định danh phổ biến duy nhất (Unique Universal Identifier - UUID) như trong bước 3. Một khóa UUID được đảm bảo là duy nhất và không bao giờ thay đổi trong một bản ghi UDDI. Những khóa này trông giống như một chuỗi số thập lục phân ngẫu nhiên có định dạng. Chúng có thể được sử dụng để tham chiếu đến một điểm mà chúng được gán vào. Các khóa UUID tạo trong một bản ghi chỉ có nghĩa nội trong bản ghi đó.

Các khách hàng khác, như là e-Marketplaces, máy tìm kiếm, và ứng dụng thương mại trong bước 4, sử dụng một bản ghi UDDI để khám phá các dịch vụ quan tâm. Và ngược lại, các doanh nghiệp khác có thể yêu cầu các dịch vụ này, cho phép sự tích hợp đơn giản và thay đổi theo thời gian như minh họa trong bước 5.

3.6. Sự khác nhau giữa SOA và Web Service

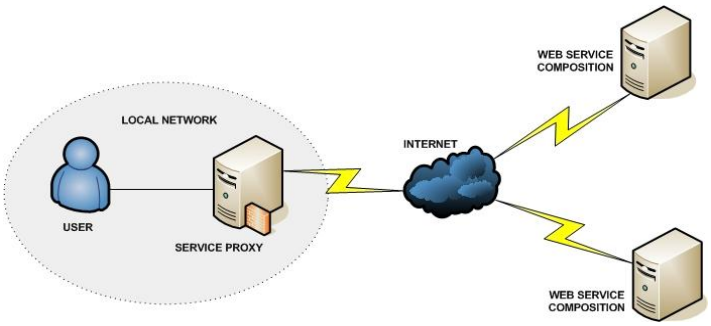
Trong chương trình, chúng ta đã nghiên cứu về cấu trúc SOA và các khái niệm cũng như thành phần Web Service và việc triển khai một hệ thống SOA và tích hợp với Web Service là điều không hề dễ. Ngày nay điều này không còn cần thiết nữa vì trong một hệ thống SOA, các chức năng này đã được "dịch vụ hóa" và cung cấp ra cho các đối tượng bên ngoài truy cập thông qua các nghi thức chuẩn của WebService.

Rõ ràng, theo định nghĩa thì Web Service là đặc tả công nghệ còn SOA là triết lý thiết kế phần mềm. Web Service đưa ra giải pháp kỹ thuật để thực hiện SOA, nhưng SOA cũng có thể thực hiện với các giải pháp kỹ thuật khác không phải Web Service. Tuy vậy, SOA và Web Service có mối quan hệ tương hỗ: sự phổ biến của Web Service giúp thúc đẩy sự phát triển của SOA, và kiến trúc tốt của SOA sẽ giúp Web Service thành công.

SOA là một phương pháp thiết kế, trong khi WebService chỉ là một công nghệ. SOA có thể được thực hiện qua công nghệ WebService nhưng cũng có thể thực hiện thông qua các công nghệ khác. Kiến trúc SOA sử dụng WebService như là một giải pháp chính để giải quyết vấn đề tích hợp nghiệp vụ giữa các hệ thống.

3.7. Tìm hiểu về Service Proxy

Service Proxy về bản chất cũng là một Web Service được triển khai ở phía Máy khách. Service Proxy chứa các đoạn mã để chỉ rõ sự kết hợp giữa các giao diện Web Service, Service Proxy thường nằm phía bên trong một hệ thống mạng máy tính phức tạp. Mô hình tổng quan của một hệ thống với Service Proxy được thể hiện thông qua hình dưới đây[5]



Hình 3.9: Minh hoạ mô hình Web Service với Service Proxy

Service Proxy sẽ thực thi phương thức giống như phương thức được triển khai trên các remote Web Service, tuy nhiên Service Proxy không thực hiện bất kì một thao tác tính toán nào cả, nó chỉ có nhiệm vụ nhận các yêu cầu từ phía khách rồi chuyển tiếp các thông điệp yêu cầu đến các remote Web Service, tại remote Web Service sẽ thực thi các thao tác tính toán trên các dữ liệu được chuyển đến đó và trả lại kết quả cho Service Proxy. Service Proxy nhận kết quả trả về và chuyển tiếp cho máy khách.

Một Service Proxy sẽ thực thi lần lượt ba thao tác yêu cầu dưới đây để thực hiện một lời gọi phương thức tới một remote Web Service:

- Truyền đối số
- Xây dựng lời gọi Web Service
- Đọc kết quả trả về từ Remote Web Service

Chúng ta thường sử dụng Service Proxy trong trường hợp số lượng mã tích hợp Web Service thường lớn, và tồn tại việc trùng lặp các lời gọi tới cùng một dịch vụ trong các vị trị khác nhau của chương trình.

Và khi sử dụng Service Proxy chúng ta hoàn toàn có thể:

- Nhóm dịch vụ bằng kĩ thuật đóng gói, lựa chọn các thứ bậc của dịch vụ.
- Chia lớp con từ lớp trừu tượng do đó cung cấp thêm các dịch vụ khác.
- Mỗi một lớp của Service Proxy trình bày Web Service.

Thông thường thì chúng ta không phải tự viết ra Service Proxy. Service Proxy có thể dễ dàng tự được sinh ra từ file WSDL

CHƯƠNG 4: CÁC KỸ THUẬT BẢO MẬT WEB SERVICE

4.1. Tổng quan về an toàn Web Service

Từ những giai đoạn đầu tiên của Internet, các doanh nghiệp luôn đòi hỏi rất khắt khe về vấn đề bảo mật trong thương mại điện tử. Những hạn chế của tường lửa như việc giám sát các gói tin được truyền tải dựa trên giao thức HTTP là chưa có; điều này có thể khiến cho máy chủ có nguy cơ bị những cuộc tấn công không hề biết được biết trước. Đã có rất nhiều các thuật toán đưa ra cơ chế và những chuẩn về bảo mật như sự mã hoá khoá thông tin, chữ ký số …; nhưng hầu hết chỉ tập trung vào việc đưa ra các định dạng bảo về dữ liệu trong quá trình trao đổi, không quan tâm đến việc xác định các nghi thức mà các bên cần thực hiện khi tương tác với nhau.

Ngoài ra, những chuẩn chung về việc chỉ ra nghi thức giao tiếp giữa Web Service là chưa có, đã khiến cho các sản phẩm hỗ trợ bảo mật của Web Service không thể tích hợp với nhau, mặc dù các sản phẩm này đều được thiết kế dựa trên chuẩn về bảo mật cho web service.

Một chuẩn an toàn chung cho các hệ thống giao dịch trên mạng thường phải tập trung vào những điều sau[7] :

- Identification: định danh được những ai truy cập tài nguyên hệ thống.
- Authentication: chứng thực truy cập tài nguyên của người muốn sử dụng.
- Authorization: cho phép giao dịch khi đã xác nhận định danh người truy cập.
- Integrity: toàn vẹn thông tin trên đường truyền.
- Confidentiality: độ an toàn, không ai có thể đọc thông tin trên đường đi.
- Auditing: kiểm tra, tất cả các giao dịch đều được lưu lại để kiểm tra.
- Non-repudiation: độ mềm dẻo, cho phép chứng thực hợp tính hợp pháp hóa của thông tin đến từ một phía thứ ba ngoài hai phía là người gửi và người nhận.

4.2. Bảo mật Web Service:

4.2.1. Khái niệm:

Web Service Security là một chuẩn an toàn cho SOAP và cả những phần mở rộng của SOAP, nó được dùng khi muốn xây dựng những web service toàn vẹn và tin cậy. Web Service Security đảm bảo cho tính an toàn, sự toàn vẹn thông điệp và tính tin cậy của thông điệp.

4.2.2. Chứng thực trong một ứng dụng

❖ ***Phía máy khách***

Máy khách sẽ cung cấp một dấu hiệu an toàn trong tập tin mô tả cũng như phải chỉ rõ một Callback handler để lấy tài khoản và mật khẩu trong thông điệp SOAP và gửi tới máy chủ.

❖ ***Phía máy chủ***

Để cấu hình máy chủ an toàn cần có một dấu hiệu an toàn hợp lệ cũng như phải chỉ rõ một Callback handler để đọc dấu hiệu an toàn trong SOAP máy khách và xác nhận nó.

4.2.3. Các bước tạo sự an toàn thông tin trong một ứng dụng

❖ ***Phía máy khách***

Chỉ rõ những thành phần của thông điệp mà phải có chữ ký hay một dấu hiệu chứng thục nào đó (nằm ở phần thân thông điệp)

Chỉ rõ một khóa trên hệ thống tập tin mà sẽ ký lên thông điệp. Chỉ những máy khách đã được cấp quyền mới có quyền sở hữu khóa này.

Chỉ rõ những giải thuật sẽ được sử dụng bởi khóa để ký lên thông điệp.

❖ ***Phía máy chủ***

Chỉ rõ những thành phần của thông điệp cần được ký. Nếu thông điệp đến không có một chữ ký hợp lệ, thì yêu cầu sẽ thất bại.

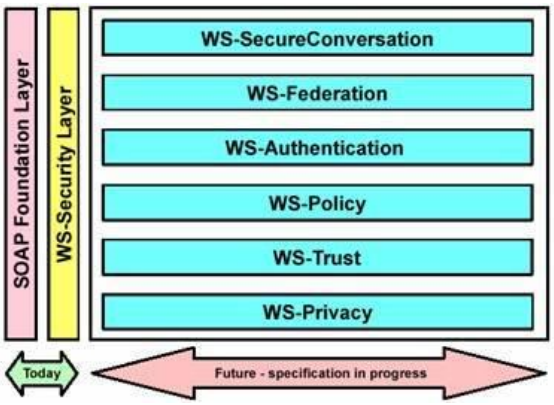
Chỉ rõ một khóa để duyệt chữ ký của thông điệp đến xem có hợp lệ hay không.

Chỉ rõ giải thuật mà khóa sử dụng để bảo đảm toàn vẹn của thông điệp gửi đến.

Thông điệp phản hồi phải được ký và cung cấp thông tin chữ ký khi phản hồi.

4.2.4. Những thành phần mở rộng của Web Service Security

Do Web Service Security chỉ là một lớp trong nhiều lớp của giải pháp an toàn đầy đủ, nên cần một mô hình an toàn chung lớn hơn để có thể bao phủ tất cả các khía cạnh an toàn khác như đăng ký (logging) và không từ chối (non-repudiation).



Hình 4.1: Mô hình an toàn cho Web service

Trong mô hình này các thành phần quan trọng bao gồm:

WS-SecureConversation Describes:quản lý và xác nhận thông điệp trao đổi giữa các phần, bao gồm sự trao đổi ngữ cảnh, thiết lập, dẫn xuất ra những phiên.

WS-Authentication Describes:quản lý những dữ liệu,chính sách cần chứng thực.

WS-Policy Describes:quản lý những ràng buộc của những chính sách an toàn ở các điểm trung gian và đầu cuối.

WS-Trust Describes: cho phép Web Service an toàn trao đổi,tương tác với nhau.

4.3. Giới thiệu các kỹ thuật Web Service Security

eXtensible Access Control Markup Language (XACML)

Security Assertion Markup Language (SAML)

XML Key Management Specification (XKMS)

Web Services Policy Framework (WS-Policy)

eXentisble Rights Markup Language (XrML)

Secure Socket Layer (SSL)

4.3.1. eXtensible Access Control Markup Language (XACML)

4.3.1.1: Tổng quan XACML

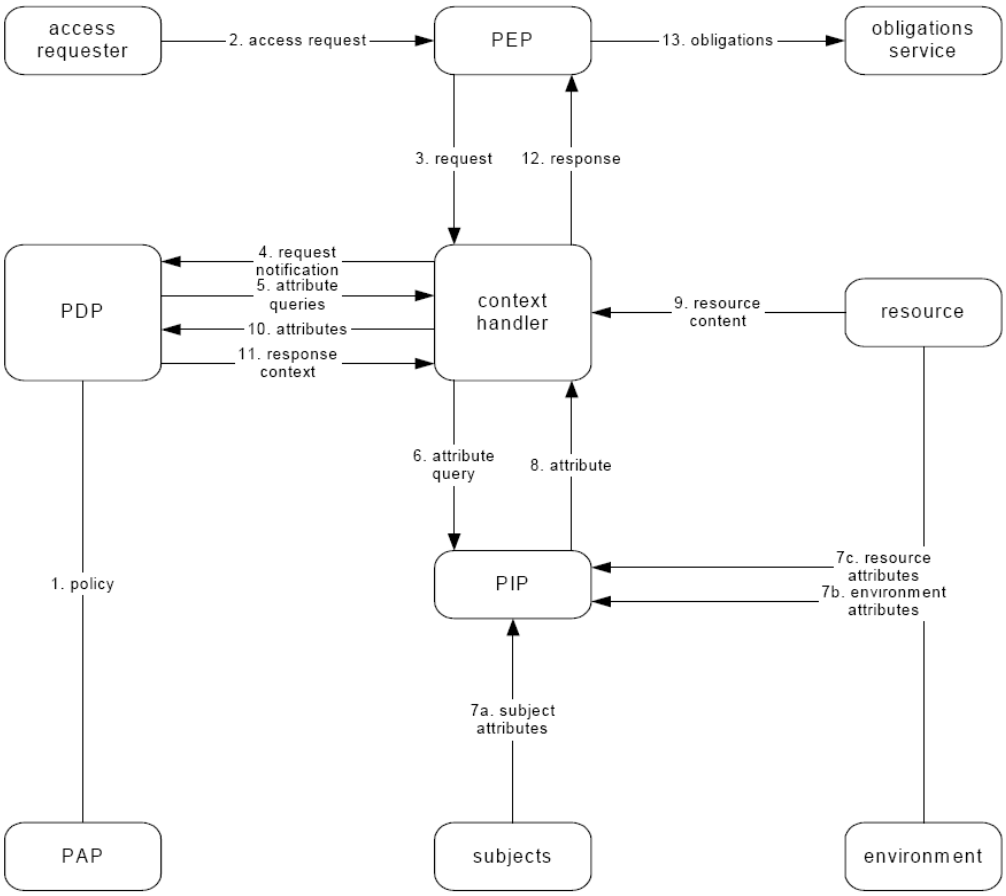
Các chính sách điều khiển truy cập rất phức tạp và phải được thi hành tại nhiều điểm. Trong một môi trường phân phối, ví dụ như thiết lập một dịch vụ web, thực hiện các chính sách điều khiển truy cập bằng cách cấu hình chúng tại mỗi điểm, khiến cho các chính sách trở nên đắt tiền và không đáng tin cậy. Hơn nữa, các chính sách điều khiển truy cập thường được thể hiện thông qua các ngôn ngữ độc quyền và khác nhau.

XACML được hình thành để giải quyết vấn đè này, bằng cách cung cấp một tiêu chuẩn, ngôn ngữ duy nhất để xác định các chính sách điều khiển truy cập. XACML phiên bản 2.0 đã được chấp nhận như một tiêu chuẩn OASIS cùng với sáu cấu hình của XACML: SAML 2.0, XML Digital Signature, Privacy Policy (chính sách bảo mật), Hierarchical Resource (phân cấp tài nguyên) và RBAC (Role-Based Access Control). XACML là một tiêu chuẩn bổ sung của OASIS để đưa ra các quyết định việc điều khiển truy cập [8]

XACML được thực hiện trong XML.

Các đối tượng của XACML được dùng để tạo ra một tiêu chuẩn cho việc miêu tả các thực thể điều khiển truy cập và các thuộc tính của chúng. Chúng đề nghị nhiều các điều khiển truy cập hơn việc từ chối và cấp quyền truy cập

4.3.1.2: Mô hình của XACML



Hình 4.2: XACML Architecture

PEP: Policy Enforcement Point: Thực hiện kiểm soát truy cập bằng cách yêu cầu quyết định và thực thi các quyết định ủy quyền.

PAP: Policy Administration Point: Tạo và lưu trữ chính sách bảo mật.

PDP: The Policy Decision Point: Nhận, xem xét yêu cầu. Sau đó áp dụng các chính sách cùng với việc đánh giá các chính sách đó rồi trả về PEP

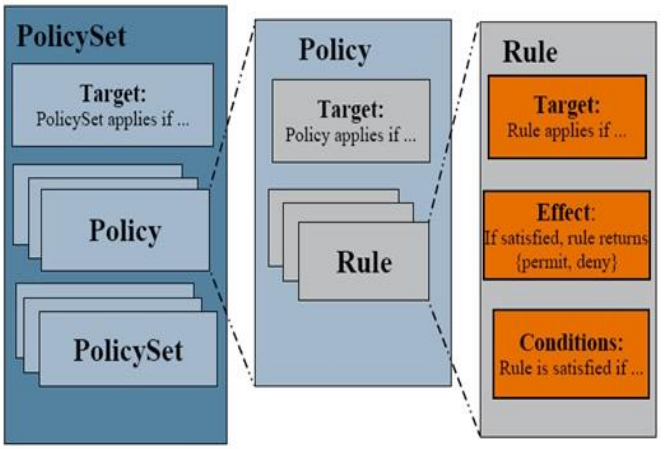
PIP: Policy Information Point: Là nguồn gốc của các giá trị thuộc tính hoặc các dữ liệu cần thiết để đánh giá chính sách.

Context Handler: Xác định để chuyển đổi các yêu cầu theo định dạng gốc của nó với hình thức XACML và chuyển đổi các quyết định ủy quyền theo hình thức XACML sang định dạng gốc.

Các chính sách XACML sẽ được nạp vào PAP, tại đây các chính sách sẽ được gửi tiếp tới PDP. PDP là điểm quyết định sẽ sử dụng chính sách nào cho các yêu cầu truy cập. Khi có một yêu cầu truy cập được gửi tới PEP, nó sẽ tiếp nhận các yêu cầu và thực hiện chúng bằng cách yêu cầu tới các văn bản xử lý. Các văn bản này lại được gửi yêu cầu tới PDP, tại đây các yêu cầu được xử lý và sau đó được gửi phản hồi lại cho Context Handler. Và tiếp tục gửi lại cho PEP – nơi thực hiện các chính sáchh sau khi đã qua quá trình xử lý và thực hiện tại PDP. Sau khi thực thi các chính sách PEP sẽ gửi các chính sách tới các Máy chủ chứng thực và tạo ra các tài nguyên để chia sẻ. Các tài nguyên này kết hợp cùng với PIP được lưu trữ trở lại cho Context Handler phục vụ cho những yêu cầu lần sau.

Các XACML Context Handler sẽ cách ly và xử lý các ứng dụng cho các đầu vào và đầu ra sử dụng PDP. Trong thực tế, đó là các Context Handler dùng để dịch các yêu cầu về truy cập ứng dụng từ định dạng ban đầu của nó sang định dạng theo chuẩn trên. Mấu chốt XACML là xác định các cú pháp cho một ngôn ngữ chính sách bất kỳ, ngữ nghĩa cho các quy tắc chính sách và giao thức nhằm đáp ứng các yêu cầu giữa PEP và PDP.

4.3.1.3: Thành phần của XACML

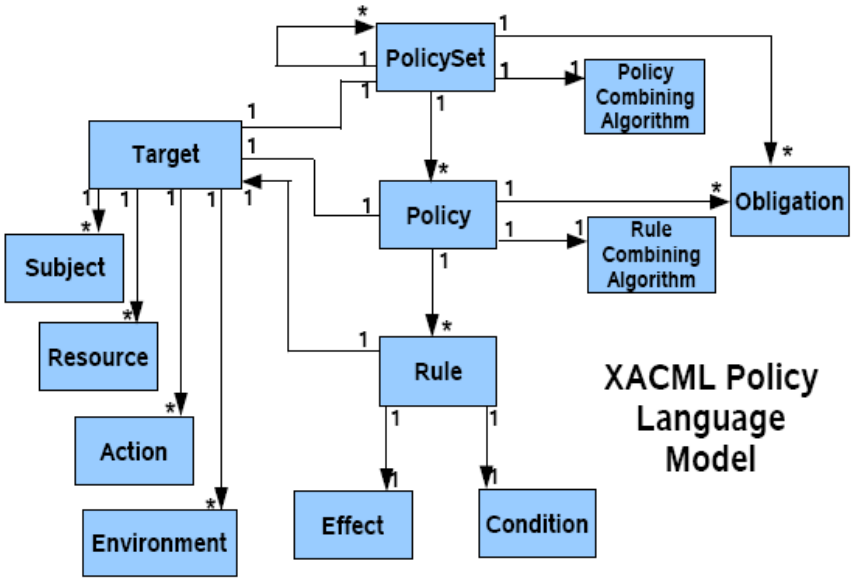


Hình 4.3: Thành phần của XACML

Một XACML bao gồm 3 thành phần cơ bản sau:

- Rule (quy tắc)
- Policy (chính sách)
- Policy Set (thiết lập chính sách)

4.3.1.4: Mô hình ngôn ngữ XACML



Hình 4.4: XACML Policy Language Model

Theo như lý thuyết được trình bày bên trên, xuất phát từ Target: bao gồm ba thành phần chính: suject, action, resource có mối quan hệ với target cụ thể như trên hình vẽ. Target cũng có mối quan hệ tương tự với Policy Set – Policy – Rule theo tỷ lệ cụ thể như hình vẽ. Giao tiếp giữa Policy Set và Policy thông qua việc kết hợp sử dụng các chính sách và tương tự từ Policy với Rule là việc kết hợp thông qua các quy tắc. Các mối quan hệ được miêu tả cụ thể như trên hình vẽ [7].

Mô hình cấu trúc XACML là một thể thống nhất trong đó các thành phần có mối quan hệ chặt chẽ với nhau thông qua các quy tắc đã được xác định trước

❖ **Cấu trúc XACML Request**

Bao gồm bốn thành phần:

- Thuộc tính đối tượng
- Thuộc tính tài nguyên
- Thuộc tính hành động
- Thuộc tính môi trường

Hình 4.5: XACML Request

❖ **Cấu trúc XACML Response**

Bao gồm ba thành phần

- Quyết định
- Trạng thái
- Trách nhiệm

Hình 4.6: XACML Response

4.3.2. Security Assertion Markup Language (SAML)

4.3.2.1: Tổng quan SAML

SAML là sự kết hợp giữa S2ML và AuthML, được phát triển thông qua OASIS. SAML là một tiêu chuẩn dựa trên XML, được hình thành như một khuôn khổ cho việc trao đổi thông tin liên quan đến an ninh, thể hiện dưới các xác nhận và sự tin tưởng giữa các bên tham gia trao đổi, nhằm xác thực giao tiếp người dùng, quyền lợi và các thuộc tính thông tin.

4.3.2.2: Hoạt động của SAML

Hỗ trợ việc khẳng định các chứng thực gốc duy nhất giữa các domain với nhau. Việc khẳng định có thể truyền đạt thông tin về các thuộc tính của đối tượng và có thể quyết định ủy quyền cho đối tượng được phép truy cập tài nguyên nhất định.

- Xác thực tin tưởng
- Chứng thực các vấn đề liên Domain
- Tập trung các vấn đề xác thực liên

SAML hỗ trợ ba loại hình xác nhận:

- Xác thực: Các đối tượng quy định được chứng thực tại thời điểm cụ thể
- Thuộc tính: Các đối tượng quy định có liên quan tới thuộc tính được cung cấp.
- Quyết định ủy quyền: một yêu cầu cho phép đối tượng quy định để truy cập vào tài nguyên quy định đã được cấp hoặc từ chối.

4.3.2.3: Đặc điểm của SAML

Một SAML duy nhất khẳng định có thể chứa một số báo cáo khẳng định về chứng thực, ủy quyền và các thuộc tính. Khẳng định là do cơ quan SAML, cụ thể là cơ quan thẩm định, cơ quan thuộc tính, hoặc là một điểm quyết định chính sách.Tuy nhiên, nó không cung cấp cơ chế để kiểm tra, thu hồi chứng trỉ. SAML cung cấp bối cảnh chứng thực, được truyền đạt (hoặc tham chiếu) một sự khẳng định của chứng thực đó. Khuôn khổ quy định của SAML là nhằm hỗ trợ nhiều tình huống kinh doanh thực trên thế giới, từ những người mà trong đó khách hàng là một trình duyệt để thêm những phần phức tạp nơi mà Web Service có liên quan [3].

Bảo mật thông tin SOAP, khẳng định SAML có thể được sử dụng trong thông điệp SOAP để thực hiện vấn đề an ninh và nhận dạng thông tin giữa các hành động

trong giao dịch.Các SAML Token của tổ chức WSS OASIS quy định cách xác nhận SAML nên được sử dụng cho mục đích này. The Liberty Alliance's Identity Web Service Frameword (ID-WSF) cũng sử dụng SAML xác nhận như là thẻ an ninh cơ sở để cho phép việc an toàn và tôn trọng sự riêng tư khi tiếp cận với các Web Service

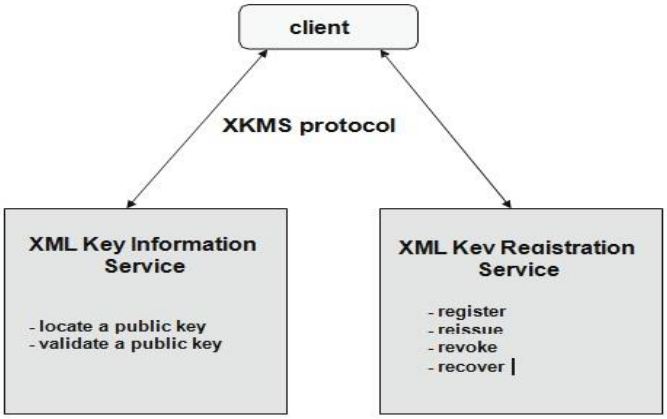
4.3.3. XML Key Management Specification (XKMS)

Các khóa công cộng là các khối cơ bản xây dựng cho chữ ký và chứng nhân kỹ thuật số. Khóa công khai quản lý bao gồm việc tạo ra, lưu trữ an toàn, phân phối, sử dụng và hủy bọ chúng. Các khóa công cộng có thể được tạo ra bởi một gói phần mềm chạy trên nền tảng của các ứng dụng khách hàng và sau đó đăng ký một khóa cơ sở hạ tầng công cộng , chứng nhận ủy quyền hoặc ứng dụng khách hàng có thể yêu cầu một chứng nhận tham gia đến một cơ sở hạ tầng để tạo ra các khóa này. Khi một bên sử dụng một khóa công khai, nó có nhu cầu để xác định tính hợp lệ của nó, nghĩa là, nó cần phải xác minh các khóa công cộng chưa hết hạn hoặc đã bị thu hồi bởi nhà cung cấp Web Service. Khóa công cộng có thể được cấp bằng nhiều cách khác nhau, có thể có nhiều hơn một khóa công khai liên quan tới khóa công cộng. Tuy nhiên, các khóa cơ sở hiện nay dựa trên bộ công cụ độc quyền, làm cho tương tác giữa các ứng dụng khách hàng và các hạ tầng trờ nên tốn kém và khó khăn hơn. Hơn nữa, các ứng dụng khách hàng phải tự thực hiện rất tốn kém các hoạt động như xác nhận chữ ký, xác nhận dây chuyền và kiểm tra thu hồi. Do đó cần phải đơn giản hóa các nhiệm vụ của các bên khi chúng ta công khai khóa công cộng, cũng như cho phép các chứng thực khác nhau, hoặc thậm chí khóa khác nhau. Hơn nữa, các khóa công cộng có thể được đại diện trong XML, và là cơ sở của XML Encryption và XML chữ ký. Những vấn đề mô tả ở trên đã dẫn đến việc định nghĩa các chuẩn đối với XML Key Management [9]

Hơn nữa,WS-Security xác định các cơ chế cơ bản cho việc cung cấp thông điệp an toàn, thông điệp SOAP được bảo vệ bởi WS-Security trình bày ba vấn đề chính đó là: tính không tương thích định dạng bảo mật thẻ;sự khác biệt không gian tên và sự tin cậy an ninh thẻ. Để khắc phục những vấn đề trên, cần thiết phải xác định tiêu chuẩn mở rộng để WS-Security cung cấp các phương pháp nhằm đưa ra, đổi mới và xác nhận thẻ bảo mật và để thiết lập và đánh giá sự xuất hiện, mối quan hệ tin tưởng lẫn nhau.

XKMS là một giao thức được phát triển bởi W3C để mô tả sự phân phối và đăng ký khóa công cộng, nó làm giảm các ứng dụng phức tạp cú pháp của nền tảng được sử dụng để thiết lập các mối quan hệ tin tưởng.

Trong hình vẽ sau, một dịch vụ X-KISS cung cấp cho khách hàng hai chức năng và có thể thực hiện bởi chính các dịch vụ X-KISS hoặc của một khóa cơ sở cơ bản. Đó là chức năng: xác định ví trí và tính xác thực. Đối với mục đích mã hóa, các chức năng cho phép một người gửi không cần biết chính xác liên kết với người nhận để có được thông điệp đó. Các dịch vụ X-KISS không thực hiện bất kỳ sự khẳng định về tính hợp lệ của các liên kết giữa dữ liệu và khóa[4].



Hình 4.7: XKMS Services

Đối với việc xác thực của một khóa, những thông tin được cung cấp bởi người ký có thể là chưa đủ cho người nhận có thể thực hiện việc xác minh mật mã và quyết định có nên tin tưởng vào khóa ký kết này hay không, hoặc là các thông tin không thể thực thi trong một định dạng người nhận có thể sử dụng được. Các chức năng xác nhận cho phép các khách hàng để có được từ các dịch vụ X-KISS một sự khẳng định rõ ràng, đó là hiệu lực của sự ràng buộc giữa các khóa và các dữ liệu công cộng, ví dụ: một danh từ hoặc một tập hợp các thuộc tính mở rộng. Hơn nữa, các dịch vụ X-KISS đại diện cho tất cả các yếu tố dữ liệu mà được liên kết với cùng một khóa công khai.

XKRSS định nghĩa một giao thức cho việc đăng ký và quản lý các khóa thông tin quan trọng. Bạn có thể đăng ký các khóa với một dịch vụ XKMS bằng cách sau: Các dịch vụ XKMS tạo ra một cặp khóa cho khách hàng và đăng ký các khóa công khai của chính cặp khóa đó và gửi các khóa riêng của cặp khóa này cho khách hàng của mình sử dụng chúng. Các khách hàng cũng có thể nói cho dịch vụ XKMS để họ giữ lại các khóa riêng tư nhằm phục vụ cho trường hợp khách hàng khi bị mất.

4.3.4. Web Services Policy Framework (WS-Policy)

Các dịch vụ Web Policy Framework tiêu chuẩn cung cấp một mô hình mở rộng và ngữ pháp cho phép các dịch vụ web mô tả chính sách của chúng. Các tiêu chuẩn WS-Policy đã được hình thành để cung cấp một mô hình chung, phù hợp với việc thể hiện tất cả các loại mô hình chính sách miền cụ thể, từ việc vận chuyển cấp an ninh, chính sách sử dụng nguồn tài nguyên, đặc điểm chất lượng dịch vụ và quy trình kinh doanh end-to-end. Cốt lõi của mô hình là các khái niệm về sự khẳng định chính sách, xác định hành vi, đó là việc yêu cầu một hoặc nhiều hơn một, của một đối tượng chính sách. Ngữ nghĩa của việc xác nhận chính là các miền cụ thể. Cách tiếp cận được thông qua bởi WS-Policy là xác định khẳng định tên miền cụ thể trong thông số kỹ thuật riêng biệt. Chính sách khẳng định có thể được xác định trong thông số kỹ thuật công cộng như WS-SecurityPolicy và WS-PolicyAssertion hoặc bởi các thực thể sở hữu các Web Service. Đáng chú ý là sự khẳng định này có thể làm hài lòng bằng cách sử dụng SOAP Message Security, WS-Security hoặc bằng cách sử dụng cơ các cơ chế khác trong phạm vi bảo đảm thông tin SOAP. Ví dụ: bằng cách gửi tin nhắn trong một giao thức như HTTPs. Các đối tượng mà chính sách này áp dụng cho một thông điệp chính sách đối tượng (thông điệp SOAP) và tiêu chuẩn WS-PolicyAttachment mà thực thể hoặc tổ chức WSDL và UDDI áp dụng[3].

WS-Policy định nghĩa các điều kiện theo một yêu cầu có thể đáp ứng, tương ứng, khẳng định chính sách của các web servuce đó, giải pháp thay thế chính sách và cuối cùng là toàn bộ các chính sách:

- Một sự khẳng định chính sách được hỗ trợ bằng cách yêu cầu khi và chỉ khi người yêu cầu đáp ứng được các yêu cầu tương ứng để khẳng định.
- Một chính sách được hỗ trợ bằng cách yêu cầu khi và chỉ khi có yêu cầu hỗ trợ ít nhất là một trong những lựa chọn thay thế trong chính sách đó.

  Khung chính sách được bổ sung bởi ba tiêu chuẩn[5]:

  - WS-Policy Assertion: xác định cấu trúc của một chính sách khẳng định
  - WS-Policy Attachment: định nghĩa làm thế nào để chính sách liên kết với web service hoặc bằng cách trực tiếp nhúng nó trong WSDL, định nghĩa hoặc gián tiếp liên kết thông qua UDDI.WS-PolicyAttachment cũng xác định làm thế nào để thực hiện liên kết các chính sách cụ thể với tất cả hoặc một phần của một kiểu cổng WSDL khi tiếp cận từ thực hiện cụ thể.

- WS-Security Policy:xác định một tập các khẳng định chính sách tương ứng với tiêu chuẩn bảo mật thông điệp SOAP, đó là thông điệp khẳng định tính toàn vẹn, tin tưởng bảo mật khẳng định, và tin tưởng an ninh khẳng định. Một chính sách WS-Security tiếp cận thông qua WSDL hoặc UDDI, cho phép người gửi yêu cầu để xác định xem WS-Security là tùy chọn hay bắt buộc đối với web service bất kỳ. Nếu nó là bắt buộc, người yêu cầu có thể xác định kiểu bảo mật mã hóa mà web service cung cấp. Người yêu cầu cũng có thể xác định xem họ cần phải ký tên vào thông điệp hay không và những phần nào để đăng nhập. Cuối cùng, yêu cầu xác định có thể mã hoác các thông điệp và nếu có là những thuật toán sử dụng.

4.3.5. eXentisble Rights Markup Language (XrML)

Kỹ thuật và các công cụ được sử dụng để cung cấp bảo mật hệ thống, chẳng hạn như tường lửa phục vụ việc truy cập vào mạng và hệ thống kiểm soát truy cập hạn chế truy cập dữ liệu được lưu trữ, không thể thực thi các quy định kinh doanh mà cách mọi người sử dụng và phân phối dữ liệu bên ngoài hệ thống.

Việc kiểm soát và thực thi phân phối, sử dụng thông tin số đã được giải quyết bằng cách quản lý bản quyền số (Digital Right Management DRM). Thuật ngữ này thường được gọi bằng luật về quyền tác giả, chủ sở hữu nội dung khi tìm kiếm phương tiện đề kiểm soát sử dụng tài sản trí tuệ của mình.Hệ thống DRM về cơ bản thực hiện hai chức năng chính đó là giám sát và điều khiển truy cập:

- Chức năng giám sát, cho phép việc theo dõi những gì đang thực sự được chuyển giao qua mạng đến tay người nhận.
- Chức năng điều khiển truy cập và sử dụng kiểm soát những gì người dùng có thể hoặc không thể làm gì với nội dung kỹ thuật số chuyển giao cho máy tính của mình.

Các mô tả về hoạt động cho phép cho người dùng trên một nội dung kỹ thuật số là khái niệm tương tự như mô tả về các hoạt động trong chính sách kiểm soát truy cập. Các chính sách kiểm soát truy cập được gắn với các nội dung kỹ thuật số của nó trong một hộp an toàn, để các nội dung kỹ thuật số đi kèm với mô tả của chính sách điều khiển truy cập áp dụng cho nó. Mục đích DRM thi hành việc truy cập cụ thể và chính sách kiểm soát sử dụng kết hợp với các nội dung kỹ thuật số.

XrML là một ngôn ngữ XML mà xác định làm thế nào để mô tả các quyền, lệ phí và điều kiện để sử dụng nội dung kỹ thuật số, với tính toàn vẹn thông điệp và tổ chức chứng thực. XrML đã được hình thành để hỗ trợ thương mại trong các nội dung kỹ thuật số, đó là việc xuất bản và bán sách điện tử, phim kỹ thuật số, kỹ thuật số âm nhạc, trò chơi tương tác, phần mềm máy tính và sáng tạo khác được phân phối dưới dạng kỹ thuật số. XrML dự định hỗ trợ truy cập và đặc điểm kỹ thuật của việc sử dụng điều khiển các đối tượng an toàn kỹ thuật số trong trường hợp trao đổi tài chính.

Đặc điểm cốt lõi kỹ thuật XrML cũng xác định các tập thường được sử dụng, quyền hạn cụ thể, đặc biệt là các quyền liên quan đến quyền khác, chẳng hạn như vấn đề, thu hồi, ủy quyền. Phần mở rộng cho các XrML có thể định nghĩa về quyền cho việc sử dụng các ứng dụng cụ thể. Ví dụ: nội dung XrML gia hạn xác định quyền thích hợp cho việc sử dụng sản phẩm kỹ thuật số (sử dụng và in quyền). Một thực tế tài nguyên đại diện cho các đối tượng trong đó một bên có thể được cấp cho một người đứng đầu. Một nguồn tài nguyên có thể là một công việc kỹ thuật số, chẳng hạn như âm thanh hoặc tập tin video, hoặc hình ảnh, dịch vụ, chẳng hạn như là dịch vụ email, hoặc thậm chí mẩu thông tin có thể được sử dụng bởi một địa chỉ email, thuộc tài sản nào khác hay thuộc tính.

4.3.6. Giao thức bảo mật SSL

4.3.6.1: Tổng quan về SSL

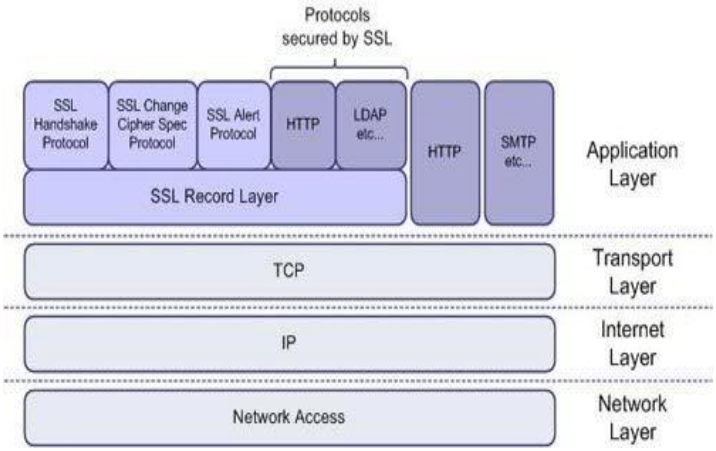
SSL là một sự xuất hiện bổ sung của VPN (Virtual Private Networks).Nó được thiết kế cho giải pháp truy cập từ xa và không cung cấp những kết nối site-to-site. SSL VPNs cung cấp vấn đề bảo mật truy cập đầu tiên những ứng dụng web.

SSL VPNs hoạt động ở tầng phiên của mô hình tiêu chuẩn OSI. Và bởi vì máy khách là một trình duyệt web nên những ứng dụng chúng hỗ trợ trình duyệt web,mặc định, nó sẽ làm việc với một giải pháp VPN. Vì thế những ứng dụng như Telnet, FTP, SMTP, POP3, multimedia, hệ thống điện thoại di động IP, điều khiển desktop từ xa, và những cái khác không làm việc với SSL VPNs bởi vì chúng không sử dụng trình duyệt web cho giao diện đầu cuối người dùng của họ. Tất nhiên, nhiều nhà cung cấp cũng sử dụng cả java hoặc ActiveX để nâng cao SSL VPNs Thêm vào đó để phân phối những thành phần SSL VPNs khác, chẳng hạn như thêm vào những chức năng bảo mật cho việc xóa hết những dấu vết từ một hoạt động của một khách hàng trên máy tính của họ sau khi SSL VPNs đã được kết thúc. Cisco chỉ sự bổ xung SSL VPN như là WebVPN.

SSL được coi là giao thức bảo mật trong lớp vận chuyển (Layer Transport) có tầm quan trọng cao nhất đối với sự bảo mật của các trình ứng dụng trên Web. Và đó là một giao thức đa mục đích được thiết kế để tạo ra các giao tiếp giữa hai chương trình ứng dụng trên một cổng định trước (thông thường là socket 433) nhằm mã hóa toàn bộ thông tin đi/đến, mà ngày nay được sử dụng rộng rãi cho giao dịch điện tử như truyền số hiệu thẻ tin dụng, mật khẩu, số bí mật cá nhân (PIN) trên Internet. Giao thức SSL được hình thành và phát triển đầu tiên vào năm 1994 bởi nhóm nghiên cứu Netscape dẫn dắt bởi Elgammal và ngày nay đã trở thành chuẩn bảo mật thực hành trên mạng Internet. Phiên bản SSL hiện nay là 3.0 và vẫn đang tiếp tụ được bổ sung và hoàn thiện. Chức năng chính là bảo vệ bằng mật mã lưu lượng dữ liệu HTTP.

4.3.6.2 Cấu trúc của một giao thức bảo mật SSL

Cấu trúc và giao thức SSL tương ứng được minh họa trong hình dưới đây.SSL ám chỉ một lớp (bảo mật) trung gian giữa lớp vận chuyển và lớp ứng dụng.SSL được xếp lớp lên trên một dịch vụ vận chuyển định hướng nối kết và đáng tin cậy. Về khả năng, nó có thể cung cấp các dịch vụ bảo mật cho các giao thức ứng dụng tùy ý dựa vào TCP chứ không chỉ HTTP. Thực tế, một ưu điểm chính của các giao thức bảo mật lớp vận chuyển nói chung và giao thức SSL nói riêng là chúng độc lập với ứng dụng theo nghĩa là chúng có thể được sử dụng để bảo vệ bất kỳ giao thức ứng dụng được xếp lớp lên trên TCP một cách trong suốt.SSL có một định hướng máy khách-máy chủ mạnh mẽ và thật sự không đáp ứng các yêu cầu của các giao thức ứng dụng ngang hàng[4].



Hình 4.8: Cấu trúc của SSL và giao thức SSL

⇨ Tóm lại: SSL cung cấp sự bảo mật truyền thông vốn có ba đặc tính cơ bản:

- Các bên giao tiếp có thể xác thực nhau bằng cách sử dụng mật mã khóa chung.
- Sự bí mật của lưu lượng dữ liệu được bảo vệ
- Tính xác thực và tính toàn vẹn của lưu lượng dữ liệu cũng được bảo vệ

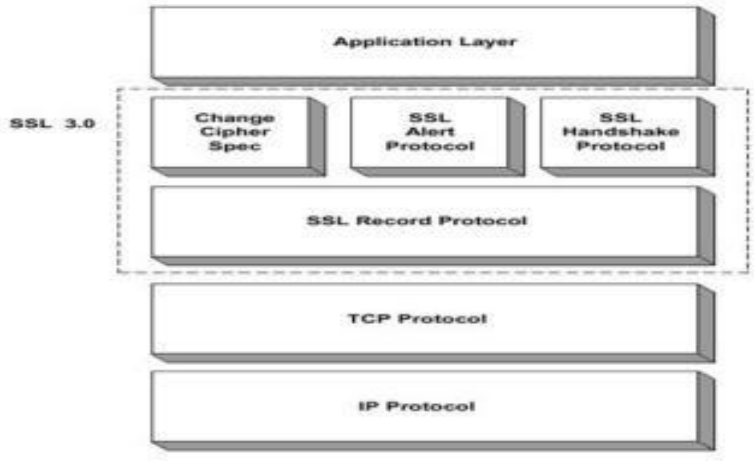
Để sử dụng SSL, máy khách và máy chủ đều phải sử dụng giao thức SSL:

- Sử dụng các số cổng chuyên dụng được dành riêng bởi Internet Asigned Numbers Authority (IANA).M ột số cổng riêng biệt phải được gán cho mọi giao thức ứng dụng vốn sử dụng SSL.
- Sử dụng số cổng chuẩn cho mọi giao thức ứng dụng và để thương lượng các tùy chọn bảo mật như là một phần của giao thức ứng dụng
- Sử dụng một tùy chọn TCP để thương lượng việc sử dụng một giao thức bảo mật

4.3.6.3: Các giao thức bảo mật SSL

❖ **SSL Record Protocol**

SSL Record Protocol [4] [5] nhận dữ liệu từ các giao thức con SSL lớp cao hơn và xử lý việc phân đoạn, nén, xác thực và mã hóa dữ liệu. Chính xác hơn, giao thức này lấy một khối dữ liệu có kích cỡ tùy ý làm dữ liệu nhập và tọa một loạt các đoạn dữ liệu SSL làm dữ liệu xuất (hoặc còn được gọi là các bản ghi) nhỏ hơn hoặc bằng 16,383 byte.



Hình 4.9: Các bước SSL Record Protocol

Các bước khác nhau của SSL Record Protocol vốn đi từ một đoạn dữ liệu thô đến một bản ghi SSL Plaintext (bước phân đoạn), SSL Compressed (bước nén) và SSL Ciphertext (bước mã hóa). Sau cùng, mỗi bản ghi SSL chứa các trường thông tin sau

- Loại nội dung.
- Số phiên bản của giao thức.
- Chiều dài.
- Tải trọng dữ liệu (được nén và được mã hóa tùy ý).
- MAC.

Loại nội dung xác định giao thức lớp cao hơn vốn phải được sử dụng để sau đó xử lý tải trọng dữ liệu bản ghi SSL (sau khi giải nén và giải mã hóa thích hợp). Số phiên bản của giao thức xác định phiên bản SSL đang sử dụng (thường là version 3.0). Mỗi tải trọng dữ liệu bản ghi SSL được nén và được mã hóa theo phương thức nén hiện hành và thông số mật mã được xác định cho session SSL.

Lúc bắt đầu mỗi session SSL, phương pháp nén và thông số mật mã thường được xác định là rỗng. Cả hai được xác lập trong suốt quá trình thực thi ban đầu SSL Handshake Protocol. Sau cùng, MAC được thêm vào mỗi bản ghi SSL. Nó cung cấp các dịch vụ xác thực nguồn gốc thông báo và tính toàn vẹn dữ liệu. Tương tự như thuật toán mã hóa, thuật toán vốn được sử dụng để tính và xác nhận MAC được xác định trong thông số mật mã của trạng thái session hiện hành. Theo mặc định, SSL Record Protocol sử dụng một cấu trúc MAC vốn tương tự nhưng vẫn khác với cấu trúc HMAC hơn. Có ba điểm khác biệt chính giữa cấu trúc SSL MAC và cấu trúc HMAC:

- Cấu trúc SSL MAC có một số chuỗi trong thông báo trước khi hash để ngăn các hình thức tấn công xem lại riêng biệt.
- Cấu trúc SSL MAC có chiều dài bản ghi.
- Cấu trúc SSL MAC sử dụng các toán tử ghép, trong khi cấu trúc MAC sử dụng moduloe cộng 2.

Tất cả những điểm khác biệt này hiện hữu chủ yếu vì cấu trúc SSL MAC được sử dụng trước cấu trúc HMAC trong hầu như tất cả thông số kỹ thuật giao thức bảo mật Internet. Cấu trúc HMAC cũng được sử dụng cho thông số kỹ thuật giao thức TLS gần đây hơn

Một số giao thức con SSL được xếp lớp trên SSL Record Protocol. Mỗi giao thức con có thể tham chiếu đến các loại thông báo cụ thể vốn được gửi bằng cách sử dụng SSL Record Protocol. Thông số kỹ thuật SSL 3.0 xác định ba giao thức SSL sau đây:

- Alert Protocol: được sử dụng để chuyển các cảnh báo thông qua SSL Record Protocol. Mỗi cảnh báo gồm 2 phần, một mức cảnh báo và một mô tả cảnh báo.
- Handshake Protocol: là giao thức con SSL chính được sử dụng để hỗ trợ xác thực máy khách và máy chủ và để trao đổi một khóa session.
- ChangeCipherSpec Protocol: được sử dụng để thay đổi giữa một thông số mật mã này và một thông số mật mã khác. Mặc dù thông số mật mã thường được thay đổi ở cuối một sự thiết lập quan hệ SSL, nhưng nó cũng có thể được thay đổi vào bất kỳ thời điểm sau đó

Ngoài những giao thức con SSL này, một SSL Application Data Protocol được sử dụng để chuyển trực tiếp dữ liệu ứng dụng đến SSL Record Protocol.

❖ **SSL Handshake Protocol**

SSL Handshake Protocol[4] là giao thức con SSL chính được xếp lớp trên SSL Record Protocol. Kết quả, các thông báo thiết lập quan hệ SSL được cung cấp cho lớp bản ghi SSL nơi chúng được bao bọc trong một hoặc nhiều bản ghi SSL vốn được xử lý và được chuyển như được xác định bởi phương pháp nén và thông số mật mã của session SSL hiện hành và các khóa bảo mật mã của nối kết SSL tương ứng. Mục đích của SSL Handshake Protocol là yêu cầu một máy khách và máy chủ thiết lập và duy trì thông tin trạng thái vốn được sử dụng để bảo vệ các cuộc liên lạc. Cụ thể hơn, giao thức phải yêu cầu máy khách và máy chủ chấp thuận một phiên bản giao thức SSL chung, chọn phương thức nén và thông số mật mã, tùy ý xác thực nhau và tạo một khóa mật chính mà từ đó các khóa session khác nhau dành cho việc xác thực và mã hóa thông báo có thể được dẫn xuất từ đó.

Các thuật toán mã hóa và xác thực của SSL được sử dụng bao gồm (version3.0):

- DES: chuẩn mã hóa dữ liệu (1977).
- DSA: thuật toán chữ ký điện tử, chuẩn xác thực điện tử.
- KEA: thuật toán trao đổi khóa.

- MD5: thuật toán tạo giá trị "băm".
- RC2, RC4: mã hóa Rivest.
- RSA: thuật toán khóa công khai, cho mã hóa và xác thực.
- RSA key exchange: thuật toán trao đổi khóa cho SSL dựa trên thuật toán RSA.
- SHA-1: thuật toán hàm băm an toàn, phát triển và sử dụng bởi chính phủ Mỹ.
- SKIPJACK: khóa đối xứng phân loại được thực hiện trong phần cứng Fortezza
- Triple-DES: mã hóa DES ba lần.

Cơ sở lý thuyết và cơ chế hoạt động của các thuật toán sử dụng về bảo mật trên hiện nay là phổ biến rộng rãi và công khai, trừ các giải pháp thực hiện trong ứng dụng thực hành vào trong các sản phẩm bảo mật (phần cứng, phần mềm).

Đã có những kết luận cho rằng SSL cung cấp sự bảo mật hoàn hảo ngăn việc nghe lén và những cuộc tấn công thụ động khác, và người thực thi giao thức này sẽ ý thức đến một số cuộc tấn công chủ động tinh vi hơn.

4.3.7. Khai thác tính năng bảo mật của bộ thư viện WSE

Có rất nhiều lựa chọn khác nhau có sẵn để giúp an ninh các Web Service và các tổ chức khác nhau có các tiêu chí khác nhau giải quyết vấn đề an ninh của họ. Và trong đồ án này, tôi xin lựa chọn nghiên cứu về WSE.

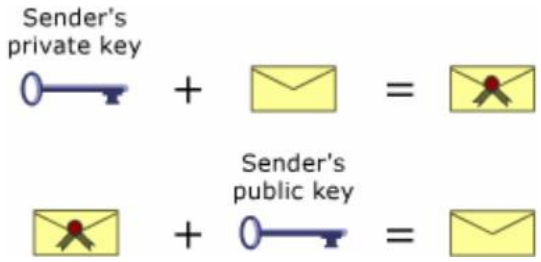
WSE 3.0 (Web Services Enhancements 3.0) là bộ thư viện lập trình trên nền .NET, hỗ trợ trong việc xây dựng các dịch vụ web theo những chuẩn mới nhất như WS-Security, WS-SecureConversation, WS-Trust, WS-Policy, WS-SecurityPolicy, WS-Addressing và MTOM. WSE hỗ trợ các thẻ nhằm bảo mật thông tin các Request giữa Máy khách và Máy chủ. Với bộ thư viện WSE, chúng ta có thể đưa các tính năng liên quan đến bảo mật này vào dịch vụ web trong lúc thiết kế bằng cách sử dụng mã lệnh, hay vào thời điểm triển khai thông qua việc sử dụng các tập tin chính sách. Hiện nay bộ thư viện này đang được sử dụng rộng rãi trên thế giới, điều này giúp hệ thống có tính tương tác cao khi đưa vào sử dụng [5].

4.3.7.1: Những tính năng bảo mật WS của WSE

WSE sử dụng các cơ chế được định nghĩa trong Web Service Security để đặt các ủy quyền chứng thực như một thẻ bảo mật vào trong các thông điệp SOAP. Sau đó sẽ thực hiện kiểm tra tính hợp lệ của những thẻ này trước khi chuyển quyền thực thi

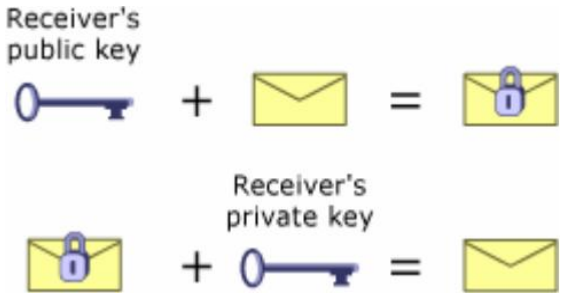
cho Web Service. WSE 2.0 hỗ trợ các loại thẻ sau: username/password, X.509 Certificate, Kerberos ticket, Security Context token và các loại security token do người dung định nghĩa. WSE còn cho phép các nhà phát triển xây dựng riêng cho mình các dịch vụ thẻ bảo mật. Các dịch vụ này có thể tạo ra các loại thẻ khác mà có thể dựng trong quá trình tương tác với các Web Service nào tin tưởng vào dịch vụ này. Thông qua việc hỗ trợ xác nhận số hay mã hóa các thông điệp SOAP sẽ tăng cường khả năng an toàn cho các Web Service.

Xác nhận một số thông điệp SOAP sẽ giúp cho đối tượng nhận thông điệp kiểm tra được thông điệp có bị thay đổi hay không.



Hình 4.10 : Xác nhận một số thông điệp

Mã hóa thông điệp SOAP sẽ đảm bảo cho chỉ những WS mong muốn mới có thể đọc được nội dung của thông điệp đó.



Hình 4.11 : Mã hóa một thông điệp

4.3.7.2: WSE hỗ trợ Policy

WSE hỗ trợ nhà phát triển đưa ra các yêu cầu về quá trình gửi và nhận thông điệp bằng cách dung các tập tin câu hình. Và cũng tương tự như thế, phía gửi cũng phải viết mã lệnh để lấy được yêu cầu này từ phía nhà cung cấp. Nay thì các yêu cầu này có thể được cung cấp thông tin qua các tập tin cấu hình.

Khi các cơ chế xác nhận Policy được chỉ định thì:

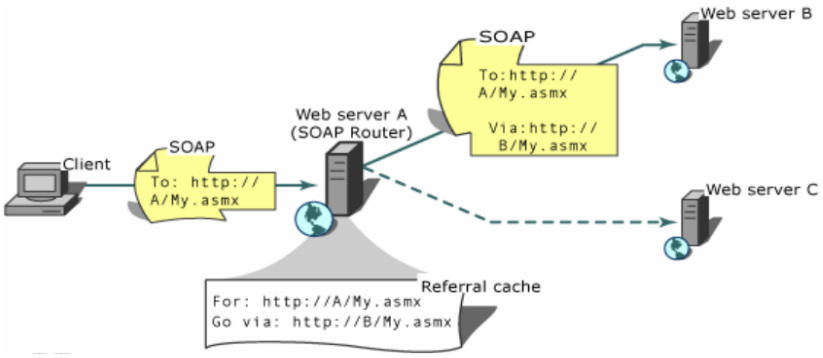
- Các thông điệp SOAP khi được gửi đi sẽ qua quá trình kiểm tra để đảm bảo chúng thỏa mãn các Policy assertion của phía gửi. Nếu không thỏa mãn, WSE sẽ đưa ra một ngoại lệ.
- Các thông điệp SOAP trước khi được nhận vào sẽ phải được kiểm tra xem có đáp ứng được các Policy assertion của phía nhận hay không? Nếu không, thông điệp đó sẽ sẽ được gửi trả về hay một ngoại lệ sẽ được đưa ra.
- WSE đã hỗ trợ sẵn một vài cơ chế xác nhận Policy (ví dụ: yêu cầu phần body của thông điệp phải được xác nhận – signed bởi một X.509 certificate). Ngoài ra, hệ thống Policy còn cho phép them những cơ chế xác nhận Policy khác do người dùng định nghĩa.

❖ **SOAP Messaging**

Đây là một tính năng nổi trội của WSE. SOAP messaging hỗ trợ nhiều nghi thức ở tầng vận chuyển HTTP, TCP, với giao diện bất đồng bộ hay đồng bộ. Đặc biệt, khi thực hiện việc gửi và nhận các thông điệp theo nghi thức TCP thi ta không cần phải có một WS.

❖ **Điều phối các thông điệp SOAP**

Ứng dụng WSE để xây dựng các ứng dụng phân tán mà kiến trúc phân tán của nó là trong suốt đối với người dùng. Ta sử dụng một máy tính trung gian và cấu hình nó chạy WSE router. Người dùng sẽ gửi yêu cầu đến WSE router thay vì trực tiếp đến WebServivce. WSE router sau đó sẽ chuyển thông điệp SOAP đến máy đang chạy WebService dựa trên thông tin cấu hình của router. Giải pháp này giúp hệ thống linh hoạt, bền vững hơn, và ta có thể thay đổi thông tin về các máy đích khi có sự cố xảy ra.



Hình 4.12: Điều phối thông điệp SOAP

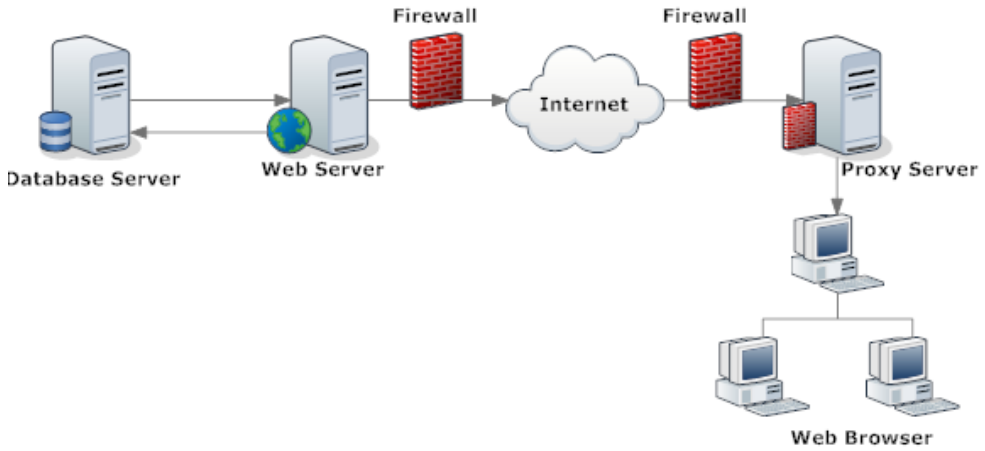
❖ **Gửi những đối tượng kèm theo các thông điệp SOAP**

WSE hỗ trợ nghi thức DIME (Direct Internet Message Encapsulation). Nghi thức này định nghĩa cơ chế để đính kèm những đối tượng khác trong thông điệp SOAP, cần thiết cho những Web Service có như cầu muốn gửi thông tin có kích thước lớn. Theo mặc định thì các thông điệp SOAP không thích hợp để gửi đính kèm các tập tin lớn.Định dạng thông điệp SOAP là theo XML nên khi thêm một tập tin vào đòi hỏi tập tin đó phải được chuyển đổi thành dạng XML. DIME giải quyết vấn đề này bằng cách định nghĩa một cơ chế đặt toàn bộ nội dung tập tin gốc nằm ở bên ngoài thông điệp SOAP, như vậy sẽ loại bỏ được việc phải chuyển đổi nội dung tập tin sang dạng XML.

CHƯƠNG 5: TRIỂN KHAI ỨNG DỤNG VÀ ĐÁNH GIÁ KẾT QUẢ

Từ các kiến thức về SOA và Web Service cũng như các kỹ thuật bảo mật Web Service ở các chương trên, chương năm sẽ đề xuất giải pháp để thực hiện bài toán đã đặt ra, triển khai và xây dựng hệ thống.

5.1. Mô tả hệ thống cần xây dựng



Hình 5.1: Hệ thống truyền dữ liệu cần xây dựng

Hệ thống nghiên cứu và xây dựng bao gồm hai máy tính (máy Database Máy chủ và máy Web Máy chủ).

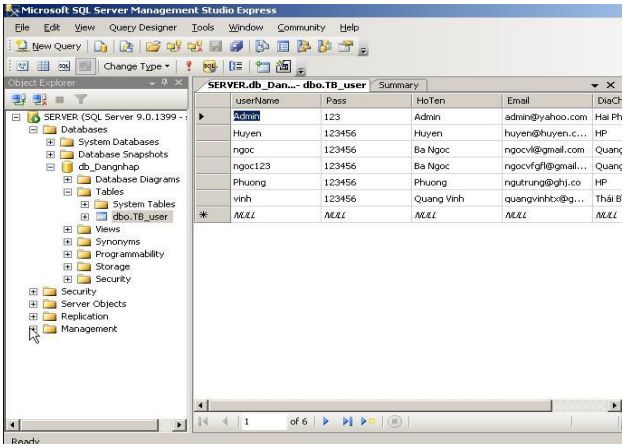
Máy DatabaseMáy chủ là nơi sẽ lưu trữ cơ sở dữ liệu của hệ thống và thực hiện tạo một Web Service có nhiệm vụ hiển thị cơ sở dữ liệu trên trình duyệt. Máy DatabaseMáy chủ được cài đặt hệ điều hành Windows Máy chủ 2003 Enterprise Edition để giúp cho việc thiết lập các cơ chế IIS cũng như FTP dễ dàng hơn.Web Service luôn luôn sẵn sàng nhận lệnh và khi có một lời gọi tới nó Web Service sẽ được thực thi và phục vụ cho lời gọi đó. Nhiệm vụ của Web Service dùng để thực hiện câu lệnh kết nối cơ sở dữ liệu và được public trên hệ thống mạng.

Máy WebMáy chủ là nơi sẽ thực hiện một lời gọi tới Web Service bằng một trình duyệt bất kỳ như FireFox, IE , Opera … Lúc này khi cần hiển thị cơ sở dữ liệu, WebMáy chủ sẽ thực hiện một lời gọi và nhờ có Web Service đã được public trên hệ thống mà các cơ sở dữ liệu sẽ được hiển thị mà không cần phải truy cập trực tiếp vào máy DatabaseMáy chủ.

5.2. Triển khai hệ thống

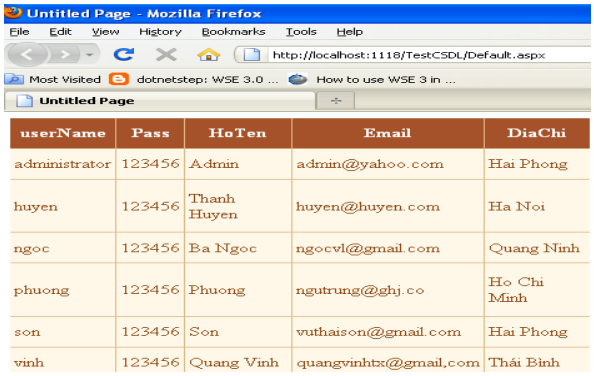
Để xây dựng một hệ thống hoàn chỉnh thực hiện chức năng hiển thị dự liệu cũng như việc bảo mật đòi hỏi phải có một thời gian dài. Trong thời gian qua tôi đã tiến hành nghiên cứu và sử dụng mô hình tương tác cơ sở dữ liệu giữa hai máy Database và máy Web, bước đầu được những kết quả và xây dựng xong chức năng hiển thị cơ sở dữ liệu mà hệ thống đặt ra và chạy thử.

Ngôn ngữ mà tôi sử dụng để xây dựng hệ thống là ASP.NET của bộ Visual Studio 2008 và Microsoft SQL Máy chủ 2005. Với những tính năng nổi bật của ngôn ngữ này và nhận thấy phù hợp với việc xây dựng và triển khai hệ thống cần xây dựng. Ngoài ra tôi sử dụng các Application có sẵn trong hệ điều hành Windows XP như Internet Information Service (IIS), Microsoft .NET Framework 3.5



Hình 5.2: Cơ sở dữ liệu User trên máy DatabaseMáy chủ

Máy WebMáy chủ thực thi việc gọi đến Web Service đã được public trên hệ thống để hiển thị cơ sở dữ liệu trên máy DatabaseMáy chủ bằng việc sử dụng một trình duyệt

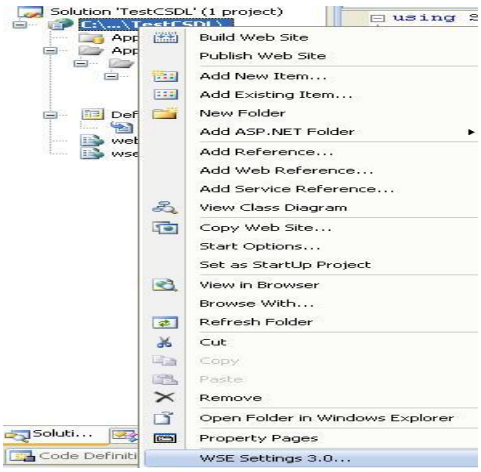


userName	Pass	HoTen	Email	DiaChi
administrator	123456	Admin	admin@yahoo.com	Hai Phong
huyen	123456	Thanh Huyen	huyen@huyen.com	Ha Noi
ngoc	123456	Ba Ngoc	ngocvl@gmail.com	Quang Ninh
phuong	123456	Phuong	ngutrung@ghj.co	Ho Chi Minh
son	123456	Son	vuthaison@gmail.com	Hai Phong
vinh	123456	Quang Vinh	quangvinhtx@gmail.com	Thái Bình

Hình 5.3: WebMáy chủ gọi tới Web Service để hiển thị dữ liệu

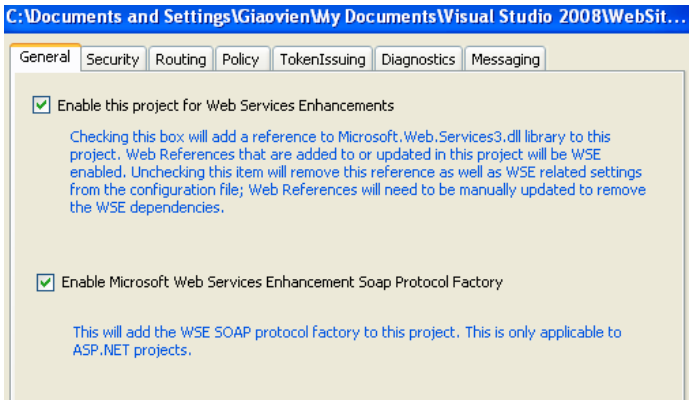
5.3. Tích hợp các thẻ bảo mật cho chương trình với công cụ WSE

Thực thi WSE với công cụ WSE phải chắc chắn rằng máy tính của chúng ta đã được cài đặt Visual Studio 2008 và bộ thư viện WSE 3.0. Sau đó thực hiện các bước để cấu hình và triển khai bộ thư viện WSE 3.0 lên chương trình.



Hình 5.4: Cấu hình WSE 3.0

Sau đó tôi sẽ cấu hình WSE 3.0 để tích hợp các thẻ bảo mật cho chương trình.



Hình 5.5: Triển khai WSE 3.0 cho chương trình hệ thống

Hai chức năng này cho phép chương trình sẽ được đặt trong môi trường bảo mật của WSE 3.0 với bộ thư viện Microsoft.Web.Service3.dll. Tại thẻ Security tôi thêm hai thẻ chính của WSE 3.0 là X509v3 Token Manager và Kerberos Token Manager. Ngoài ra tôi triển khai một Token trong Security Token Managers là Username Token Manager. Sau khi các thẻ trong thư viện WSE 3.0, chương trình đã được bảo mật. Những thông tin trong thông điệp Request mà bên WebMáy chủ gửi đến cho DatabaseMáy chủ để hiển thị dữ liệu cũng được mã hóa và bảo đảm. Và ngay cả trên đường truyền dữ liệu từ WebMáy chủ và DatabaseMáy chủ cũng đã được hỗ trợ bảo mật bởi Firewall được thiết lập mặc định trên các hệ điều hành.

```
<microsoft.web.services3>
  <security>
    <binarySecurityTokenManager>
      <add valueType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-pro
      <add valueType="http://docs.oasis-open.org/wss/oasis-wss-kerberos-token-profile-1.1#GS
    </binarySecurityTokenManager>
    <securityTokenManager>
      <add type="Microsoft.Web.Services3.Security.Tokens.UsernameTokenManager, Microsoft.Web
    </securityTokenManager>
  </security>
  <diagnostics>
    <trace enabled="true" input="InputTrace.webinfo" output="OutputTrace.webinfo" />
  </diagnostics>
</microsoft.web.services3>
```

Hình 5.6: Tích hợp thẻ Security vào trong WebService

5.4. Đánh giá kết quả chạy thử nghiệm chương trình

Qua thời gian chạy thử nghiệm cho thấy, chương trình thực hiện được chức năng cơ bản là hiển thị dữ liệu và đảm bảo được một số vấn đề an toàn cần thiết khi trao đổi dữ liệu.

Kết quả: Việc sử dụng thẻ Username Token Manager đã giúp cho các bên tham gia giao tiếp xác thực lẫn nhau và tránh bị giả mạo. Dữ liệu trên đường truyền được mã hóa bởi hai cơ chế bảo mật X509v3 Token và Kerberos Token luôn được đảm bảo về tính bảo mật cao.

CHƯƠNG 6: KẾT LUẬN

6.1. Tổng kết

Web Service đã và đang được triển khai và áp dụng trong nhiều lĩnh vực đời sống như ngân hàng, chứng khoán, trao đổi dữ liệu … và ngày càng trở lên phổ biến. Cùng với sự phát triển của nó là những đòi hỏi về tính an toàn, khả năng bảo mật. Bằng việc sử dụng các kỹ thuật đảm bảo an ninh Web Service sẽ giúp cho người sử dụng Web Service trở nên an tâm hơn.

Việc chọn cơ chế an toàn cho Web Service phải đòi hỏi sao cho người dùng không cảm thấy quá phức tạp hay gò bó mà phải tạo nên sự trong suốt với người dùng. Do đó, nên chọn các cơ chế an toàn mà Web Service phụ thuộc vào loại dịch vụ đó và những tính năng mà dịch vụ này cung cấp. Bên cạnh đó còn một điểm cần quan tâm đó là sự an toàn không chỉ phụ thuộc vào những giải thuật, những tiêu chuẩn, và những cơ chế an ninh Web Service mang lại, mà nó còn tùy vào thái độ của các công ty có hiểu rõ tầm quan trọng của an toàn thông tin khi triển khai các ứng dụng, giao dịch trên mạng hay không cũng rất cần thiết.

6.2. Kết quả đạt được của đồ án tốt nghiệp

Sau thời gian nghiên cứu tài liệu, tìm hiểu các chương trình mã nguồn mở, tôi đã hoàn thành xong đồ án tốt nghiệp với bài toán ban đầu đặt ra là "*Bảo mật Web Service*". Với việc lựa chọn chương trình trao đổi dữ liệu giữa hai máy tính trong mạng và đảm bảo anh ninh cho việc truyền dữ liệu. Đồ án đã đạt được một số kết quả sau:

Phân tích bài toán và tính cấp thiết của việc đảm bảo an toàn cho các trang Web Service. Đưa ra hướng phát triển cho bài toán.

Nghiên cứu về kiến trúc hướng dịch vụ SOA, Web Service và các thành phần. Mối quan hệ ứng dụng kiến trúc SOA vào xây dựng Web Service và tích hợp chúng theo chuẩn.

Tìm hiểu thực trạng bảo mật Web Service hiện nay, các công nghệ đảm bảo an ninh Web Service như công nghệ bảo mật SSL và bộ thư viện WSE.

Triển khai ứng dụng truyền dữ liệu giữa máy tính trong một mạng cục bộ và tích hợp thẻ bảo mật trong bộ thư viện WSE để bảo mật thông tin cho các bên tham gia.

6.3. Những hạn chế

Để xây dựng được một hệ thống hoàn chỉnh có thêm nhiều chức năng và đảm bảo tuyệt đối những yêu cầu đặt ra, phải cần rất nhiều thời gian. Trong thời gian nghiên cứu và triển khai đồ án, tôi cũng đã cố gắng đạt được những kết quả nhất định, tuy nhiên vẫn còn nhiều hạn chế:

Chương trình khá đơn giản chỉ với chức năng hiển thị dữ liệu, cũng như việc thiết kế dữ liệu chưa thực sự tốt.

Không được đưa ra áp dụng thực tế nên sẽ có khả năng nhiều lỗi mà người nghiên cứu không thể phát hiện ra

Về bảo mật, chưa tìm hiểu hết được các loại thẻ bảo mật Web Service, việc sử dụng bộ thư viện Web Service Enhancement vẫn chỉ dừng lại ở việc tích hợp vào chương trình mức cơ bản nhất, vẫn chưa đưa ra thực tế và sử dụng các phương pháp tấn công để kiểm tra độ bảo mật trên mức cơ bản của chương trình.

Nếu có điều kiện, trong tương lai tôi sẽ cố gắng tìm hiểu thêm về những mặt hạn chế của đồ án này và cố gắng khắc phục để tạo ra một chương trình hoàn chỉnh và có thể áp dụng vào thực tế.

TÀI LIỆU THAM KHẢO

Tài liệu Tiếng Việt

[1]Đại học Khoa học tự nhiên – Đại học Quốc gia Hồ Chí Minh (2005),

Nghiên cứu kiến trúc hướng dịch vụ (Service Oriented Architecture) và ứng dụng.

Tài liệu Tiếng Anh

[2]Doug Tidwell – James Snell – Pavel Kulchenko, Publíher: O'Reily(2001),

Programming Web Services with SOAP.

[3]Elisha Bertino – Lozenzo D.Martino – Federica Paci – Anna C.Squicciarini,

Seurity for Web Services and Service Oriented Architecture.

[4]Freier, A.O, Karlton, Kocher, Scout(1996), *The SSL Protocol Version 3.0*

online: http://wp.netscape.com/eng/ssl3/draft302.txt

[5]Hogg,Jane(2006),*Microsoft, Web Service Security, Scenarios, Patterns, and Implementation Guidance for Web Service Enhancements (WSE 3.0)*

[6]Judith Hurwitz, Publisher: For Dummies, *Service Oriented Architecture*.

[7]Jeaning Hall Gailey, *Understanding Web Services Specifications and the WSE*

[8]OASIS Standard Specification, *Web Service Security Kerberos Token Profile.*

[9]*XML Key Management Specification (XKMS) (W3C Note, 30 Marc 2001),* online: http://www.w3.org/TR/xkms