.NET Security

by: Lynn Greiner

Docid: 00011392

Publication Date: 0801

Publication Type: TUTORIAL

Preview

Distributive computing and universal access are two axioms synonymous with the World Wide Web. Applications, universal access, and the Web, however, require the ability to provide for a secure environment to protect information, people, systems, and devices through software that is also interoperable across all Web platforms. The .NET Framework and .NET security specifications are Microsoft's answer to ubiquitous and secure access for mission-critical applications. Built on an XML Web services model, .NET's Global XML Web Services Architecture (GXA), Microsoft along with IBM and VeriSign have published a set of standards called WS-Security that address not only connectivity to the .NET platform but also contain set of standard Web protocol to define interoperability to non-.NET XML Web services platforms and applications.

Report Contents:

Executive Summary

[return to top of this report]

Microsoft's Web services strategy to connect information, people, systems, and devices through software was released in 2002 under the name of .NET. Designed primarily as a methodology for developers to integrate applications and users across the Microsoft platform, .NET has evolved into a standard that is Operating System (OS) and application agnostic using Extensible Markup Language (XML) as a common thread across disparate platforms. Connecting people information and people to a

distributed computing platform with universal accesses, however, poses many security challenges as well. .NET has answered these security challenges with a set of application and connectivity security specifications, which provide the measure of security required in a Web services and Web application environment.

Built on an XML Web services model, .NET's Global XML Web Services Architecture (GXA), Microsoft, IBM and VeriSign have published a set of standards called WS-Security that address not only connectivity to the .NET platform but also contain set of standard Web protocol to define interoperability to non-.NET XML Web services platforms and applications.

Description

[return to top of this report]

Released in 2002, the Microsoft .NET and higher order .NET Framework are add-in software components to the Microsoft Windows operating system that provides a large body of pre-coded multi-language solutions to common program requirements, and manages the execution of programs written specifically for the framework. Conceived as a web services strategy built on Extensible Markup Language (XML), the .NET intent is to connect information, people, systems, and devices through software. Utilizing the Microsoft platform, .NET technology provides the tools and processes to quickly build, deploy, manage, and use connected, security-enhanced solutions with Web services. .NET-connected solutions enable businesses to integrate their systems more rapidly and in a more agile manner and help them realize the promise of information anytime, anywhere, on any device.

For example, siloed applications such as point-of-sale (POS) systems may have only been tracked to and reportable for accounting or cash receipts. In order to track inventory, ordering and shipping, separate applications or duplicate data entry may be required to complete all enterprise application requirements. .NET was one of the first architectures to provide a method by which the POS data could transcend accounting and cash receipts to allow real-time inventory updating. Orders for restock could automatically be placed and shipments tracked to reduce costs due to over/under stocking levels and minimizing or eliminating manual processes. Using web services the once-siloed applications can share information over the Internet, through systems that are operating system or back-end software agnostic because of XML.

.NET Security

All systems should be designed to provide and address security needs at the application and network layers. The key goal of Microsoft .NET is to enable everyone – from developers to IT professionals to end users – to securely manage who, and what, accesses their data. In the aforementioned POS example, by its very nature the legacy method of siloed applications that were arduous and required manual and duplicate inputs did have more than a modicum of security. The silos of data were protected from cross functional and universal access by their separation. However, as markets went virtual with ubiquitous access, the challenge is to provide the same level of siloed security connecting to applications and accessible from intranets and the Internet.

Application Security. At the application layer, .NET and the .NET Framework were designed to include

a broad and flexible range of security options that can be implemented by developers, administrators, and users. At a high level, the core security components of the .NET Framework include:

- Role-based security, which provides a unified model for authenticating and authorizing users based on identity and roles
- Evidence-based and code access security by which administrators can dictate what resources may be accessed by certain types of code. Typically the evidence is the source of the assembly (whether it is installed on the local machine, or has been downloaded from the intranet or Internet).
- Cryptography which includes functions for encryption, digital signatures, hashing, and random number generation.

These "off-the-shelf" security components of the .NET Framework address the levels of security needed to meet criteria based on how users and applications will be handling data within their organizations. But as universal access to data and distributed applications have evolved, application layer security can only be one part of the whole.

Network Security. According to a 2005 Gartner survey, eight of the top 10 security threats identified by IT professionals involved threats from outside their organizations. As most outside intrusions enter at the network layer, security must also be addressed at the point of network connectivity.

Fundamental to distributed application computing and interoperability across intranets and the Internet is XML Web services. XML Web services addresses interoperability requirements through the use of standard Web protocols such as XML, the Simple Object Access Protocol (SOAP), and Universal Discovery Description and Integration (UDDI). XML represents a standard way to present data, while SOAP allows different types of systems to “talk” with each other and directs how to use data within each system. In addition standards such as Web Service Description Language (WSDL), allow XML Web services a way to describe their interfaces in enough detail for applications to “talk” to them. UDDI enables the registration of XML Web services, allowing users to find the services quickly on the Internet.

Recognizing the ascension of XML Web services as the prima facie to universal distributive computing, .NET specifications were developed under the name of Global XML Web Services Architecture (GXA) as the basis of interoperability of .NET with other distributed applications. Built on XML Web services specifications and protocols, including SOAP and UDDI, GXA is evolving through industry partnerships in expanding and standardizing future specifications that are needed for XML Web services interoperability.

Where .NET, XML Web services and GXA meet in the realm of security from a connectivity perspective is through the publications by Microsoft, IBM and VeriSign, of a specification called WS- Security. WS-Security defines a standard set of SOAP extensions, or message headers, which can be used to implement integrity and confidentiality in Web services applications by providing standard mechanisms to exchange secure, signed messages in a Web services environment.

Parts of the WS-Security standard addressing connectivity that are either approved or proposed include:

- WS-Policy - defines how to express the capabilities and constraints of security policies.

- WS-Trust - describes the model for establishing both direct and brokered trust relationships (including third parties and intermediaries).
- WS-Privacy - defines how Web services state and implement privacy practices.
- WS-Secure Conversation - describes how to manage and authenticate message exchanges between parties, including security context exchange and establishing and deriving session keys.
- WS-Federation - describes how to manage and broker trust relationships in a heterogeneous federated environment, including support for federated identities.
- WS-Authorization - defines how Web services manage authorization data and policies.

.NET Passport

There are also user needs within the realm of Web services and security that must also be addressed. As applications cross functional and organization boundaries, the need for universal user access presents a set of challenges onto itself. The specter of having a siloed process for authentication and authorizations to applications and services does not fit into the model of ubiquitous and distributed computing. Users need a consistent yet secure method to gain the required authorizations needed to perform and integrate within the enterprise application domains while protecting sensitive user information.

.NET Passport is a universal Internet-based authentication service that provides a single sign-in (SSI) across multiple public sites and services. The SSI allows Web sites and services to provide an experience that is related to a consistent user identity. Passport account holders determine what, and how much, of their personal information is shared with member sites on the Internet, while giving Web sites and XML Web service providers a means to identify users across multiple services with a consistent set of credentials. .NET Passport relieves developers of the burden of having to anticipate and design security measures to meet every type of transaction, allowing them instead to focus on delivering value to users. At the same time, it gives users ultimate control over their information.

Current View

[return to top of this report]

Since its introduction in 2002 and as an integral part of the Microsoft Office 2003 Professional Suite of applications, NET technologies, Web services and security have become the cornerstone on which many enterprise applications have been built. Some of the well-known names now utilizing the .NET Framework for distributive computing, application and connectivity security include, Honeywell, GlaxoSmithKline, Sony, Dollar Rent A Car, Farmers Insurance, and the United States Postal Service.

With millions of users worldwide of the various Microsoft Operating Systems and applications, .NET has been become a well accepted platform from which to base and launch the enterprise distributive

computing model. Enhancements to the .NET Framework and collaborative support from IBM and VeriSign have helped the .NET Framework and .NET WS-Security to keep pace with developer demands, however, the activity to meet the deadlines and expectations for the planned releases of Vista and Office 2007 have temporarily moved to the head of the line. Not to say that there has not been a flurry of recent activity with respect to .NET security.

In November of 2006, Microsoft and Novell announced an agreement that will create a joint research facility at which Microsoft and Novell technical experts will architect and test new software solutions and work with customers and the community to build and support these technologies. At this center, developers will focus on several major technical areas. Specific to Web services and .NET is an effort to enable Windows to run on top of Linux, and Linux to run on top of Windows. Where Web services and in particular WS-Security shake out will be centered on making it easy to manage mixed Windows and SuSE Linux Enterprise environments and ways to confederate Microsoft Active Directory with Novell eDirectory, and vice-versa. For XML Web Services, GXA and WS-Security, the agreement is a possible final bridge to any gap of interoperability that my still exist. It is perceived as a win-win for users and developers, however, how this shakes out in the industry is a question waiting for an answer.

Outlook

[return to top of this report]

NET enhancements to Vista and Office 2007 may also be influenced as a result of the Microsoft/Novell agreement. Bridging the gap between Microsoft Office and the Novell supported OpenOffice.org, specifically, in ways to translate and improve interoperability between Microsoft's Open XML and OpenOffice's ODF (OpenDocument formats). What is significant about this effort is in the fact that last vestiges of propriety are considered to be a the application layer. With IBM and VeriSign in the Microsoft camp and with Novell buoyed only by an encampment of loyal Linux open systems users, there is a sense of déjà vu reminiscent of the early 1990s blood bath over LAN OS dominance that has had Microsoft and Novell as bitter rivals for over a decade.

Why Novell and why now, is in part as result of the successes of Linux and Novell's SuSE Linux Enterprise Server in Europe, Asia and most recently, the United States. Just as in world economics, for country to play in the greater world economic structure, entities must adapt to and accept the rules of market based and open economic policy. Likewise in the distributive application and computing environments, it is becoming easier proposition to play nice with the competition rather than isolate and destroy. With a common thread of XML Web services, SOAP, UDDI and WSDL, the .NET Framework's GXA and WS-Security extensions may placate developers and others entranced on either side of the battle as to which system is superior, to seek kinder/gentler common ground in strategic application development and deployment.

The details are still sketchy as to how all of the cessation of hostilities will translate into Vista, Office 2007 and beyond with respect to the .NET Framework and .NET WS-Security. As the late great Harry Truman once said, "The only thing new in the world is the history you don’t know," and in deference to President Truman also possibly the history that we did not learn from. Using the LAN OS wars in the 1990s as an example, the current agreement between bitter rivals may only be a harkening of the calm before the storm. Given the enormity of Microsoft and IBM on one side, offering an olive branch to a smaller and more vulnerable Novell, may be comparable to Goliath paling up to David. It may work to

Microsoft's or Novell's advantage or blowup in their face. Microsoft's history dictates that they intend to gain more than they will lose.

Recommendations

[return to top of this report]

Ubiquity, universal access and distributive computing are symbiotic concepts. In order for enterprises to play and compete in the world of eBusiness, having a strategy that embraces the concepts of a applications and data from anywhere and by others from outside of your organization are the building blocks for eSuccess. Where as choice in how an enterprise is achieve its objectives is always a good, thing, with respect to universal and distributive computing, this may not be the case.

The primary objective of any private enterprise is to make money. One of the ways for enterprises to make money is entice their clientele to buy more and more of their products and not the products of their competitors. The agreement between Microsoft and Novell to recommended the others products and services over their own if there is a needs deficiency, just doesn't smell right. Why this should be of concern to the enterprise is with respect to capital invested, Return on Investment (ROI), compatibility and most importantly security.

In the interim as .NET security transitions into Vista, Office 2007 and the codicils of the Microsoft/Novell agreement, the enterprise should assume a neutral position. Deployment based on the basics of XML Web Services and the associated application and connectivity security protocols will still allow enterprises and their developers ample opportunity to proceed with distributive application development and launch with little to no risk. Plans to implement WS-Security extensions may need to be put on hold as adoption by standards organizations is pending plus waiting for the devil in the details of Vista, Office 2007 and Microsoft/Novell agreement surface. .NET in its current form and feature set will continue to emerge and for the Microsoft only shops, are a safe bet. However, what was once considered a Microsoft world has tilted a bit on its axis. Prudence may be the best policy in keeping an open mind and open architecture for .NET and XML Web services.

About the Author

Lynn Greiner is Vice President, Technical Services for a division of a multi-national corporation, and also an award-winning computer industry journalist. Ms. Greiner is a regular contributor to Faulkner Information Services and a member of the Advisor Panel.

Web Links

[return to top of this report]

IBM: http://www.ibm.com/
Microsoft: http://www.microsoft.com/
Microsoft .NET: http://www.microsoft.com/net

VeriSign: http://www.verisign.com/

[return to ***top*** of this report]