Microsoft .NET - An Overview

Network security is a complicated subject, historically only tackled by well-trained and experienced experts. However, as more and more people become "wired", an increasing number of people need to understand the basics of security in a networked world."(1) "Vulnerability assessment tools are coming into widespread use, but the methods that they use are not well understood."(26) Networking security tools have been in existence for several years and have been growing in their scope and capability. The quality of an Ope...

Microsoft .NET – An Overview

By Rob McBee

Global Information Assurance Certification
GIAC Security Essentials Certification (GSEC)
Administrivia Version 2.2
Practical Paper Requirements – Version 1.4
Option 1 – Research on Topics in Information Security
September 14, 2002

Table of Contents

1 Abstract

“Network security is a complicated subject, historically only tackled by well-trained and experienced experts. However, as more and more people become “wired", an increasing number of people need to understand the basics of security in a networked world.”[1]

“Vulnerability assessment tools are coming into widespread use, but the methods that they use are not well understood.”[26] Networking security tools have been in existence for several years and have been growing in their scope and capability. The quality of an Operating System will be judged by how it is able to handle these tools. The Windows .NET Server operating system will provide many important new security features and improve on the one’s included in the original Windows 2000 Server.

According to Ed Parry, news editor for SearchWin2000, “Windows .NET Server represents the near future -- it's slated for release before the end of the year. You may wonder if you can't live without it. Or you may wonder if you need it at all.”[21] I will hopefully, give you some insight into this decision.

2 Introduction

What is Windows.NET and why is it better than Windows 2000? This is a question I am often asked. Unfortunately, everyone that asks me this question wants a one-sentence definition. .NET means different things to different people. .NET to developers is different than .NET to users is different than .NET to businesses and so on. The following analysis was taken from "Security in the Microsoft .NET Framework" written by Foundstone, Inc. and their partner, CORE Security Technologies:[12]

> Our analysis revealed that, used properly, the .NET Framework gives developers and administrators granular security control over their applications and resources; provides developers with an easy-to-use set of tools to implement powerful authentication, authorization, and cryptographic routines; eliminates many of the major security risks facing applications today due to flawed code (such as buffer overflows); and shifts the burden from having to make critical security decisions-such as whether or not to run a particular application or what resources that application should be able to access-from end users to developers and administrators.[12]

3 What is .NET

In early 2000, Microsoft announced its .NET initiative - previously code named Next Generation Windows Services or Windows 2002 Server. According to Jay Munro from ExtremeTech, he said the .NET initiative is, “a comprehensive distributed, Internet-based computing platform comprised of new development tools, runtime services, operating system features, servers, and Internet protocols. The primary .NET goals are to enable simplified development and delivery of distributed Web-based services, allow creation of powerful new B2B and B2C transactional capabilities, and enrich the user computing experience both locally and across the Web.”(7)

In Microsoft's words, .NET is "a shift in focus from individual Web sites or devices connected to the Internet, to constellations of computers, devices and services that work together to deliver broader, richer solutions". (22)

.NET is a technology evolution built from the ground up for the Web. It is not a bolt-on solution. It’s a new *platform* that can provide seamless integration of multiple applications and devices. It can be used to take advantage of the abundant computing and communications resources provided on the Internet to “enable smart, service-aware systems and richer user experiences.”(31)

.NET is based on standards and a unified programming model. Web standards are open and non-proprietary, they are built to be first-class constructs and they drive interoperability and integration of new existing systems. .NET uses open architecture and it provides a robust, open platform for developing the next generation of Web Application, XML Web services and Windows Applications. .NET is also multi-language because it works in the language of your choice.

.NET will allow you to create programs that transcend device boundaries and harness the connectivity of the Internet in your applications. Furthermore, it is viable for all your applications, as you no longer need to think about two separate infrastructures—one for Web applications and another for internal or desktop applications.

4 The .NET Platform

The .NET Platform is a set of development tools and operational systems used to build, expose and consume XML Web services. It has four main components:

- Tools – To build applications and XML Web Services (.NET Framework and Visual Studio.NET)
- Client Software – The software that powers smart devices, allowing users to interact and experience the .NET platform (ASP.NET, XML, SOAP, Web Services and Microsoft's Mobile Internet Toolkit)
- Server Infrastructure -- On which to build, host and deploy those applications and services (Windows DNA, MSMQ, COM+, IRC and IIS)
- Services – A core set of .NET building block services (Microsoft Passport™, HailStorm)

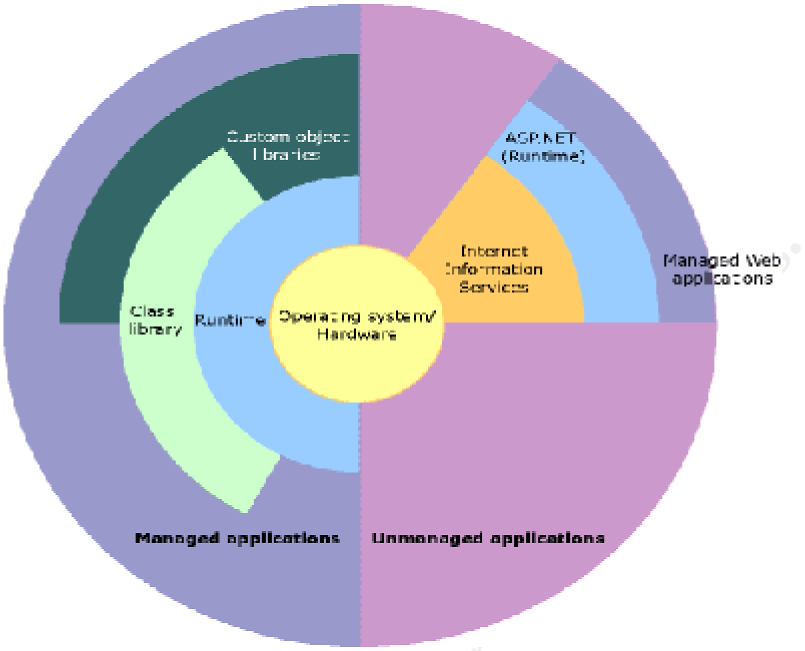
4.1 Tools

- The .NET Framework
- Visual Studio .NET

4.1.1 The .NET Framework

"The .NET Framework is an environment for building, deploying and running XML Web services and other applications."[(27)] It is also a security solution using the concept of "managed code", based on security rules. The .NET Framework is the infrastructure for the .NET Platform. It is a computing platform designed to streamline application development for the Internet. The .NET Framework has two main components, the CLR and the .NET Framework class library.

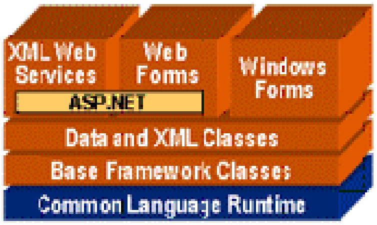
The .NET Framework in context:[14]



Source: Microsoft Corporation[14]

4.1.1.1 The Common Language Runtime (CLR)

The CLR is the foundation of the .NET Framework. Think of the Common Language Runtime as an agent that manages code at execution time, providing services such as memory management, thread management, security issues and component dependencies, while enforcing strict safety and accuracy of the code. The concept of code management is the fundamental principle of the CLR. Code that targets the CLR is known as managed code while code that does not target the Runtime is known as unmanaged code.



Source: Microsoft Corporation[16]

Another component of the CLR is the Common Type System. The Common Type System defines how types are declared, used and managed in the CLR. It also is an important part of the Runtime's support for cross-language integration.
The Common Type System performs the following functions:

- Establishes a framework that enables cross-language integration, type safety and high performance code execution.
- Provides an object-oriented model that supports the complete implementation of many programming languages.
- Defines rules that languages must follow to help ensure that objects written in different languages can interact.

4.1.1.2 The .NET Framework Class Library

According to Microsoft, "The .NET Framework Class Library is a comprehensive, object-oriented collection of reusable classes that will help you develop applications ranging from traditional command-line or Graphical User Interface (GUI) applications to applications based on the latest innovations provided by ASP.NET and Web services."(10) With these libraries, developers can write programs that will execute in the CLR and implement many new security features including permissions, authentication and cryptographic protocols. With these new restrictions, the CLR is able to "manage" the code come execution time by performing "just-in-time" compilation (JIT). JIT translates the managed code in native code before executing, ensuring that the code is checked before running.

4.1.2 Visual Studio .NET

In today's business environment applications need to be delivered quickly, maintained easily and upgraded frequently, so maximizing developer productivity is a crucial component of success. With that in mind, the best choice is to choose a language that is simple to use, powerful and productive.

Visual Studio .NET gives you a comprehensive integrated development environment (IDE) and many great tools like the Forms Designer for Windows Forms and Web Forms, the XML Designer and plenty of wizards. You can work in the language of your choice and create the broadest range of applications and interfaces including browser, mobile devices and rich Windows clients. With Visual Studio, developers will be able to use a familiar programming approach for a broad range of user interfaces, including browser and mobile clients. Building a Web application in the .NET language of your choice will be as easy as selecting New, Project.

Developer productivity is a vital ingredient for project success, as today's fast-paced world demands that application delivery be quickened. Visual Studio .NET was designed to help developers build their solutions faster with a host of new productivity tools and enhancements such as drag-and-drop form creation combined with features such as IDE, enhanced IntelliSense, Microsoft Help 2.0 and dynamic help, will help to provide organizations with the ability to minimize time to market and stay ahead of the competition.

4.1.2.1 Integrated Development Environment

The Integrated Development Environment of Visual Studio .NET saves development time and energy with a vast number of features including its ability to support many design tools and languages and to allow designers to create their own tools:

- Share a single toolbox, debugger and task window across Visual Basic, C++, and C# projects.
- Build solutions that span multiple languages and multiple projects with the shared Solution Explorer
- Reuse code using new cross-language inheritance
- Easier to write code
- Easier to share knowledge
- Leverage existing skills
- Customize the environment
- Macros and more powerful add-ins

4.1.2.2 IntelliSense®

IntelliSense, with its increased context sensitivity, provides an array of options that make language references accessible. When coding, it is no longer necessary to leave the text editor to perform searches on language elements, which means you can keep your context, find the necessary information and insert language elements directly into your code. IntelliSense automatically performs schema validation in the editor for different HTML targets like Internet Explorer and Netscape. You can specify a specific HTML version for your code or create HTML 3.2 by default. At a Microsoft's Developer's conference I went to a few months back, the Instructor said that, "IntelliSense is so advanced that it will complete your typing."

4.1.2.3 Microsoft Help 2.0 & Dynamic Help

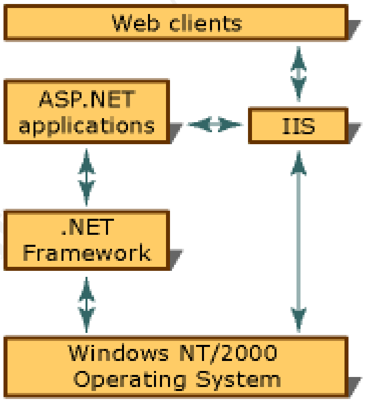
- Improves filtering, F1, searching and performance
- Attributed, consistent topics
- Provides immediate related help information and the ability to track content

4.2 Client Services

- ASP .NET
- XML
- SOAP and Web Services
- Microsoft's Mobile Internet Toolkit

4.2.1 ASP .NET

Active Server Pages .NET, ASP .NET, is a Web development platform that enables developers to build enterprise Web Applications. ASP .NET provides a new programming model and infrastructure that provides an extensible security system for Web Applications that enables you to integrate it with existing systems or to create your own system. The following diagram shows the relationships among the security systems in ASP.NET:(19)



Source: ASP Alliance(19)

With most Web sites, it is necessary to selectively restrict access to some portions of the site. You can think of a Web site as analogous to an art gallery.
The gallery is open for the public to browse, but parts of the facility, such as the business offices, are restricted to people with certain credentials. When a user enters a company's web site to purchase something, there credit card information is stored in some from of database. This information must be secured from public access. ASP .NET security features address these types of issues.

Another form of security that ASP .NET does is it authenticates user credentials. It does this in conjunction with IIS. ASP .NET can authenticate user credentials, such as names and passwords, using any of the following authentication methods:

- Windows: Basic, Digest or Integrated Windows Authentication (NTLM or Kerberos)
- Microsoft Passport authentication
- Forms-based authentication
- Client Certificates

ASP .NET architecture also provides Role-based Security to allow you to make authorization decisions based upon those users membership in various roles. This makes it easy to create an application that exposes certain things to particular users and other things to all users.

ASP .NET also provides a rich development environment that features drag and drop development and a powerful event driven architecture. It is a powerful upgrade to ASP and Web development in general that exists within the .NET Framework.

According to Microsoft, "ASP .NET is a compiled .NET Framework-based environment; you can author applications in any .NET Framework compatible language including Visual Basic®, C# and JScript. Additionally, the .NET Framework platform is available to any ASP .NET application. "(28)

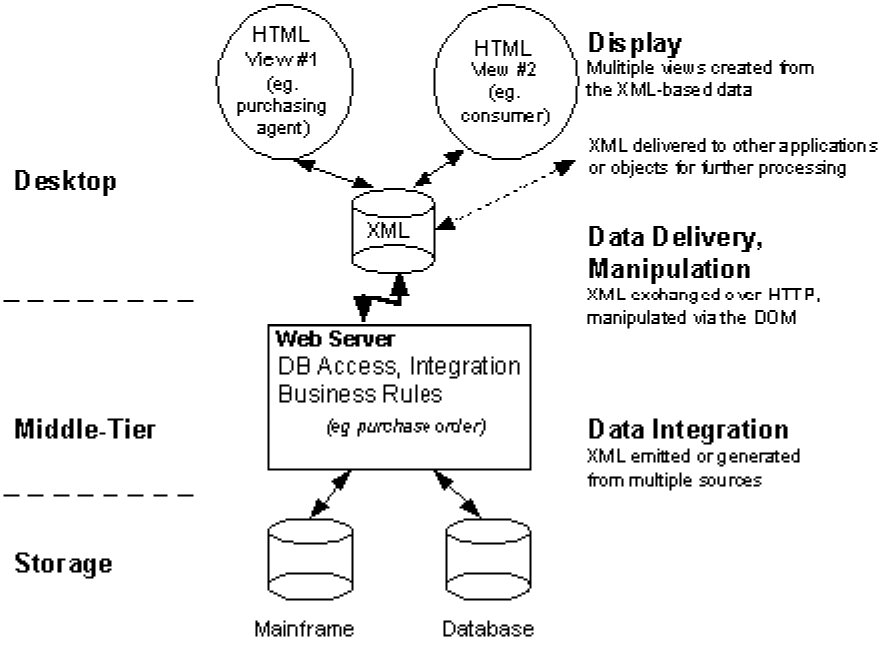
You can write your ASP .NET applications in Notepad if you like, but it's easier to use Visual Studio .NET with its powerful GUI interface and rich set of tools for developing ASP .NET applications. It also includes controls that encapsulate common HTML user-interface elements such as text boxes and drop-down menus. These controls run on the server, but push their user interface as HTML to the browser. The resulting HTML from ASP .NET applications is browser neutral, providing a rich architecture for developing HTML 3.2 and higher applications.

In ASP .NET, your code either can be stored on the same page as the tags or on a separate page known as the code behind page. This code separation allows applications to be cleanly structured and to perform much faster than the traditional ASP.

ASP .NET contains a complete request processing system for Web Applications that is modular and pluggable, which means you can change the components.
The request processing system also provides features such as caching, security and session state. Having these services "baked-in" allows you to focus on developing your core application without having to worry about creating the services provided by ASP .NET.

4.2.2 XML

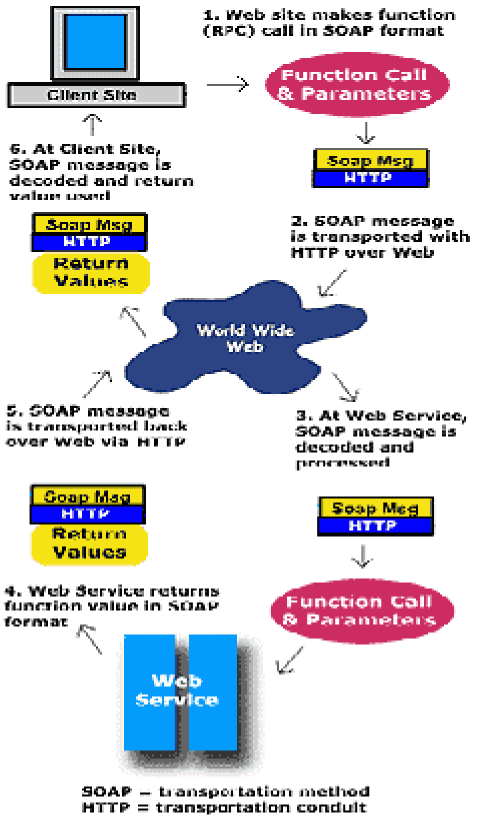
The XML standard for data representation will expand the Internet/Intranet in much the same way the HTML standard for display did a few years ago. XML is a language that provides a format for describing structured data. It also ensures that structured data will be uniform and application independent, providing interoperability using flexible, open, standards-based format. It allows for applications to be built more quickly, they are easier to maintain, and can easily provide multiple views on the structured data. The power and beauty of XML is that it maintains the separation of the user interface from the structured data. HTML specifies how to display data in the browser; XML defines the content. With XML data, you can use style sheets, such as Extensible Style Language (XSL) and Cascading Style Sheets (CSS), to present the data in the browser.


HTML View #1 (eg. purchasing agent)
HTML View #2 (eg. consumer)
Display
Mulitiple views created from the XML-based data
XML delivered to other applications or objects for further processing
Desktop
XML
Data Delivery, Manipulation
XML exchanged over HTTP, manipulated via the DOM
Web Server
DB Access, Integration
Business Rules
(eg purchase order)
Middle-Tier
Data Integration
XML emitted or generated from multiple sources
Storage
Mainframe
Database

Source: Microsoft Corporation[24]

4.2.3 SOAP and Web Services

One of the key elements in Microsoft's .NET architecture is the ability to create Web services that can be accessed by other applications over HTTP. Simple Object Access Protocol (SOAP) provides a lightweight mechanism for exchanging structured data between application components using XML. The Microsoft SOAP Toolkit (available here[25]) provides tools required to implement Web services with SOAP. The SOAP Toolkit includes a tool that will extract the type library for an existing COM component and turn it into a Service Description Language (SDL) contract, which represents that component's capabilities as a service. It will also produce necessary files to make the service available for consumption. Once you have a contract, the SOAP Toolkit includes an extension for any COM-enabled development tool (such as Visual Studio) that will automatically turn the contract into a proxy you can program against, just like a local COM component.


1. Web site makes function (RPC) call in SOAP format
Client Site
Function Call & Parameters
Soap Msg
HTTP
6. At Client Site, SOAP message is decoded and return value used
Soap Msg
HTTP
Return Values
2. SOAP message is transported with HTTP over Web
World Wide Web
5. SOAP message is transported back over Web via HTTP
3. At Web Service, SOAP message is decoded and processed
Soap Msg
HTTP
Return Values
Soap Msg
HTTP
4. Web Service returns function value in SOAP format
Function Call & Parameters
Web Service
SOAP – transportation method
HTTP = transportation conduit

Source: Purdue University[8]

Other technologies that work hand-in-glove with XML and SOAP are UDDI and WSDL. UDDI or Universal Discovery Description and Integration, functions like the "yellow pages" of Web services. It's a specification for information registries of Web Services. In that distributed registry, businesses and services are described in a common XML format. The structured data in those XML documents is easily searched, analyzed, and manipulated. WSDL, or Web Services Description Language, describes a Web service's functions. If you are providing a service, you need to be able to describe it to the world, and if you want to use a service, you need to be able to describe what you're looking for. That's WSDL. While UDDI lets you find the Web services that you are interested in, WSDL lets you interact with the Web service.

Taking all of these technologies together, we have the infrastructure we need for Web services to work. Service providers can describe themselves, service requesters can describe what they're looking for, and service brokers have an automated way of determining which requester-provider pairs are a good match. Once a match has been found, there are standard ways of finding the methods available for interacting with the service, along with any binding information necessary.

SOAP, WSDL, and UDDI are woven throughout Microsoft's .Net suite of products, and in the products of many other software vendors. These standards are real and are already in use today.

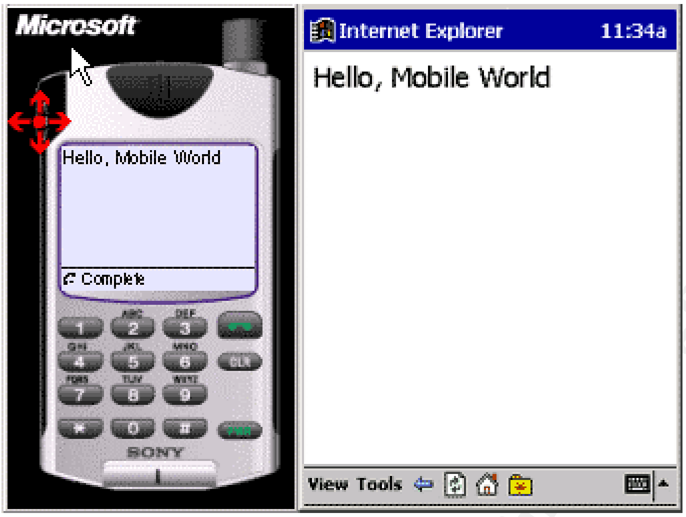
4.2.4 Microsoft's Mobile Internet Toolkit

The Microsoft Mobile Internet Toolkit insulates developers from the details of wireless technology. Thus, developers can quickly and easily build a single mobile Web application that delivers appropriate markup for a wide variety of mobile devices.

According to Microsoft, "The Mobile Internet Toolkit contains the following:"(18)

- Mobile Web Forms controls that generate a markup language for different devices
- Mobile Internet Designer that works with the Visual Studio .NET IDE to provide a drag-and-drop mobile development environment.
- Rich browser capabilities that extend ASP.NET device capabilities to mobile devices
- QuickStart tutorial with sample code
- Developer documentation·
- Device adapter code samples

The following is a sample from Microsoft of what the Mobile Internet will be about:[18]



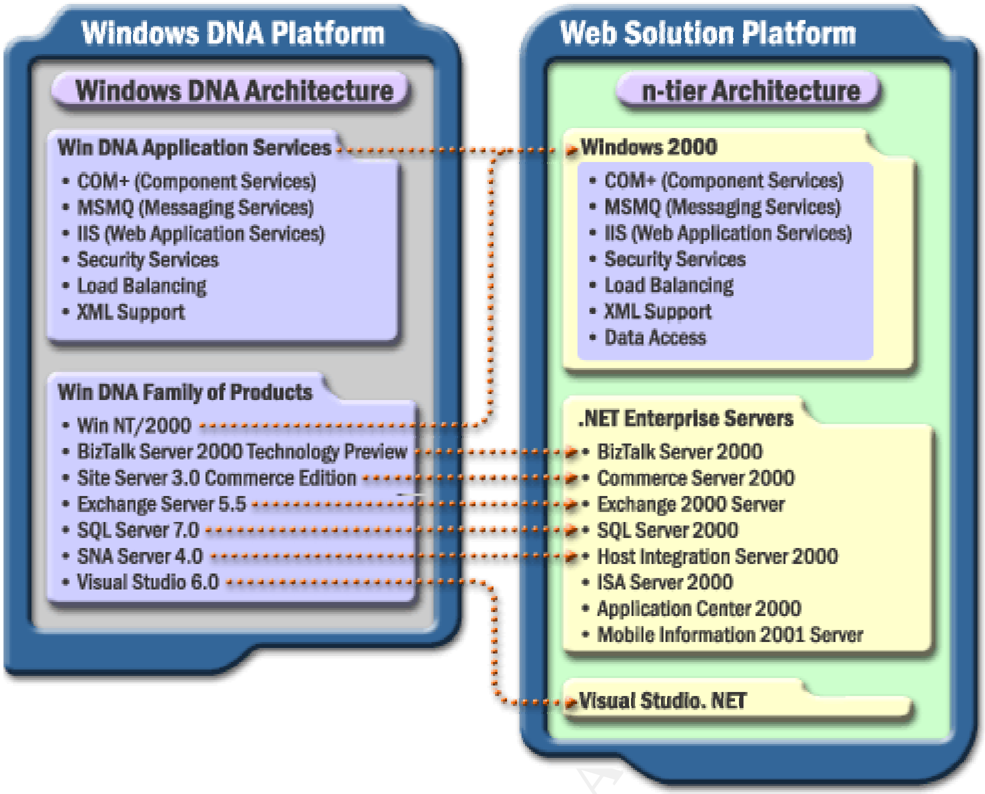
Source: Microsoft Corporation[18]

4.3 Server Infrastructure

Microsoft introduced Windows DNA (Distributed interNet Application) to provide a scalable architecture for distributed, enterprise-ready web applications. By utilizing the n-tier computing model of Windows DNA, developers have been able to use the web to reduce their deployment costs and achieve levels of scalability not possible under previous architectures, such as the client/server model.

Today, the Windows DNA programming model has evolved into Microsoft's Web Solution Platform. This new model handles literally all the traditional "plumbing" that application's require, allowing developers to focus on the business needs of a software solution. This platform will enable a new set of Internet applications and services, which is known as .NET.

The chart below shows the evolution of Windows DNA to .NET: (24)



Source: Microsoft Corporation(24)

Inter-Process Communication (IPC) is the interaction between applications and/or the components of an application. With Microsoft Windows DNA applications, this is typically COM/DCOM. COM is a complex technology and must be used properly, as it can be one of the major causes of the “DLL Hell” problem. MTS provides management for transactions, scalability, connections, and threading. Another common method of application IPC is through Message-Oriented Middleware (MOM) products, via a publish-subscribe or point-to-point paradigm. Microsoft offers MSMQ in this space. MSMQ is often used in applications to provide for asynchronous processing in the background, where the application can launch several tasks at once and use event processing to “listen” for messages returned from its background processing.

Web applications written against Microsoft’s Internet Information Server (IIS) will most likely involve HTML, Dynamic HTML (DHTML, or client-side scripting), and Active Server Pages (ASP). ASP is an open application environment in which you can combine HTML, server-side scripts (VBScript and JavaScript), and server-side COM components to create dynamic web applications. In Microsoft’s Web Solution

Platform, ASP is evolving into ASP.NET. In ASP.NET, the benefits of ASP's dynamic programming power will be improved by replacing its scripting engine with the .NET JIT (Just-In-Time) compilers, allowing multiple languages to be used in ASP files, and integrating XML throughout.

Successful global component (COM or Win32) sharing requires that any shared component function exactly like previous versions of that component. Unfortunately, complete backward compatibility is next to impossible. It is not possible to test all configurations in which a shared component may be used. Both new and old applications end up using the same component; and over time, fixing and improving the component becomes increasingly difficult, and produces more and more undesired side effects. This leads to a state known as application fragility ("DLL Hell").

One solution for reducing application fragility is to selectively isolate applications and components. Under this scenario, applications may all have access to the same component, but multiple versions of that component can be used at runtime. This is also known as side-by-side sharing. With side-by-side sharing, applications can use the specific version of a component for which they were designed and tested, even if another application requires a different version of the same component. This arrangement allows developers to build and deploy more reliable applications because developers are able to specify the version of the component they will use for their application, independent of the other applications on the system.

Although there is more than one way to run side-by-side components, the recommended method is a strategy called COM/DLL Redirection. This strategy works best when new client applications are being deployed to computers that already support several other applications, or where the new client application needs to be made more resilient to changes in shared components caused by the installation of other applications. DLL/COM redirection requires that when the application is deployed, the application executable and all side-by-side components be installed into the application directory (or a subdirectory thereof) rather than the system directory. Additionally, a ".local" file is deployed to the application directory to modify the Windows binding behavior, so that the application binds to the side-by-side component(s) rather than the globally shared version. "DLL/COM redirection is activated on an application-by-application basis by the presence of a ".local" file. The ".local" file is an empty file in the same directory as the application's .exe file, with the same name as the application's .exe file with ".local" appended to the end of the name. For example, to activate DLL/COM redirection for an application called "myapp.exe," create an empty file called "myapp.exe.local" in the same directory where myapp.exe is installed. Once DLL/COM redirection is activated, whenever the application loads a DLL or an OCX, Windows looks first for the DLL or OCX in the directory where the application's .exe file is installed. If a version of the DLL or OCX is found in the directory where the application's .exe file is installed, the application uses it regardless of any directory path specified in the application or the registry. If a version of the DLL or OCX is not found in the directory where the application's .exe file is installed, the normal search path or server path is used."[30] Side-by-side components installed in this manner should *not* be registered in the system's registry.

While most components can easily be installed side-by-side, they may not run side-by-side. This happens because some components use global state (such as settings stored in the registry), assuming that there will be only one version of the component on the computer at any time. Additionally, the component may make assumptions about the specific directory in which it is installed when locating other resources that it needs.

4.4 Services

- Microsoft Passport™
- HailStorm (.NET My Services)

4.4.1 Microsoft Passport™

Microsoft Passport uses browser and Web mechanisms for user authentication over the Internet to communications based on Kerberos, an authentication system designed to enable two parties to exchange private information across an otherwise open network. Passport now uses the HTTP and an encrypted cookie to authenticate a user. It supports single sign-on through the user's browser to an enterprise's Web server software. The future Passport will use Kerberos directly from the user's OS to the enterprise's server OS. Since Microsoft's PC and the server OS, starting with Windows 2000, have support for Kerberos built in, by the end of 2002, most PCs will be able to support this capability. However, non-PC devices that don't run Windows OS will be at a disadvantage unless they add Kerberos support.

4.4.2 HailStorm

HailStorm is a "code-name" for a set of user-centric XML Web services that puts users in control. It manages and protects user info and offers consistent and personalized access from any application, device, service or network. HailStorm's platform approach creates incredible industry opportunities. Users don't generally interact with the service directly except for administrative functions, but access happens through the different applications, devices and services. It helps manage, protect and securely share data, e.g. calendar: How to deliver information to a PC or a cell phone? How to share a subset of your calendar with co-workers, friends and family or your dentist? Rumor has it that Microsoft will rename it to .NET My Services.

HailStorm interacts via open protocols:

- Any OS, language or network
- Windows, Windows CE, UNIX, Mac, Palm, etc.
- No Microsoft software required

The following is a quote from Bill Gates, Microsoft’s Chairman and Chief Software Architect about the release of HailStorm, "HailStorm is a key milestone to deliver on the Microsoft mission to empower people through great software, any time, any place and on any device," said Bill Gates, Microsoft chairman and chief software architect. "We believe this innovation will take individual empowerment to a new level, create unprecedented opportunity for the industry and trigger a renewed wave of excitement."(17)

5 Security

According to Foundstone, Inc. and CORE Security Technologies, “The security architecture of the .NET Framework is composed of a number of core elements.”(12) They are as follows:

- Evidence-based security
- Code access security
- Role-Based security
- Cryptography

5.1 Evidence-Based Security

Evidence-Based Security is broken up into three subsystems, including:

- Evidence
- Permissions
- Policy

5.1.1 Evidence

Evidence is information that serves as input to the CLR's mechanism for making decisions based on security policy, and it indicates that code has a particular characteristic. Common forms of evidence include digital signatures and the location from which code originates, but evidence also can be custom designed to represent other information that is meaningful to the application.

Trusted application domain hosts can present evidence about an application domain that enables the CLR to decide what permissions to grant the application domain.

This information enables the CLR to evaluate machine and user policy and return the set of permissions to grant to the application domain. If the host does not have permission to provide evidence, the application domain will get the permissions that have been granted to the host.

The CLR gets evidence about assemblies from trusted application domain hosts or from the loader. Some evidence, such as from where the code originates, has to come from the application domain host because only the host knows. Other evidence, such as an assembly's digital signature, is inherent in the code itself so it can come from the loader or a trusted host. Typically, when code is loaded each assembly's digital signature is validated by the runtime. If the digital signature is valid, the host passes the signature information as evidence to the runtime's policy mechanism. In addition, an assembly or a host can provide custom evidence as a resource that is part of the assembly. Administrators and developers can define these types of evidence and extend security policy to recognize and use it.

5.1.2 Permissions

The CLR allows code to perform only those operations it has permission to perform. To enforce restrictions on managed code, the CLR uses permissions. The primary uses of permissions are as follows:

- Code can *request* the permission it needs to access resources or perform operations
- The CLR can *grant* permission to code based on characteristics of the code's identity, on what permissions were requested and on how much the code is trusted
- Code can *demand* its callers have permission

5.1.3 Policy

There are three levels on which security policy is specified: machine policy, user policy and application domain policy and each policy level has its own settings. The runtime intersects the settings for these three levels to determine the permissions that an assembly is allowed. The allowed permission set for application domains is determined by intersecting only the settings for the user and machine policy levels.

By default, both the user policy and the application domain policy specify that the set of permissions is allowed. Because the policy levels are intersected when determining the allowed permission set, the machine policy settings determine default security policy.

After the application domain policy is set, all subsequently loaded assemblies will be granted permissions under the new policy (machine policy, user policy and application domain policy). Previously loaded assemblies will get grants under the pre-existing policy (i.e., machine and user policy only). Assemblies loaded into the application domain before the policy is in place will not have permission grants re-evaluated under the newly set application domain policy.

If the host is trusted, it can provide evidence to the runtime about assemblies loaded into the application domain. If a domain host does not have sufficient permission (i.e., the SecurityPermission[32] for controlling evidence), the security enforced on the host is used to determine the security enforced on the assembly.

"Typically, after an application domain is created the host loads the first (main) assembly into the application domain and calls into it to begin execution. When code in the first assembly references code in another assembly, the loader resolves the reference, loads the appropriate assembly into the application domain and supplies the evidence about the assembly to the runtime."[29]

5.2 Code Access Security

Code Access Security is a feature of the CLR. Its job is to verify the code's identity and its origins. In the traditional server, all applications were trusted, yet in a hosted environment, applications from diverse sources may be managed in the same process space by the CLR. Additionally, complex applications have varied security needs. Under CLR management, applications are granted different levels of access and the CLR won't execute unverifiable code.

5.3 Role-Based Security

According to Foundstone, Inc. and CORE Security Technologies, "Role-Based Security defines the way the .NET Framework establishes identity, and permits or denies that identity to access resources. These two processes are frequently referred to as authentication and authorization, the linchpins of secure application design for Web applications."[12]

5.3.1 Authentication

According to Microsoft, "Authentication is the process of accepting credentials from a user and validating those credentials against some authority. If the credentials are valid, one speaks of having an authenticated identity. Authentication can be accomplished by either system or business logic, and is available through a single API."[3] .NET can obtain this "authenticated identity" in a number of ways:

- Forms-based or Cookie Authentication – where a user would supply credentials in the way of a form that would be pushed back to a server and checked against a SQL database that would return a result. If the credentials were accepted .NET would issue a cookie back to the user that contained the credentials.

- Passport Authentication -- Passport uses HTTP and an encrypted cookie to authenticate a user. In the future Microsoft is planning on using Kerberos to provide the authentication.
- IIS – IIS provides several authentication methods including: Basic Authentication, NTLM, Kerberos, Digest Authentication, and X.509 Certificates (with SSL)
- Windows Authentication – Windows, like ISS also has a few authentication methods like Kerberos, NTLM, and X.509 Certificates.

5.3.2 Authorization

Authorization is the process of determining whether a requesting identity will be granted access to a given resource. ASP.NET offers two types of authorization services: file authorization and URL authorization. File authorization is determined by the ACL's that are used based from the user's identity making the request. URL authorization is the logical mapping between the URI namespace and the users and roles being requested.

5.4 Cryptography

Today, security between computers is achieved mainly through cryptography. Both symmetric (shared secret) and asymmetric (public-key) cryptography work together in a PKI to provide the overall security.

In the 70's, Diffie & Hellman introduced the world to asymmetric ciphers, in which the key for encryption and the key for decryption were related but conspicuously different. So different, in fact, that it would be impossible to publicize one (public key) without danger of anyone being able to derive or compute the other (private key). The keys in a key pair of a public-key cipher are different, but they are also related. One must decrypt what the other encrypts. The idea that one of the keys in this pair (key pair) can be revealed publicly was so radical and appealing that this whole method of protecting data quickly became known as public-key cryptography. Asymmetric encryption is accomplished with an asymmetric algorithm, requiring an asymmetric public key for encryption, and a corresponding asymmetric private key for decryption. Anyone can encrypt clear-text using someone's public key, but only the holder of the corresponding private key can decrypt the cipher-text.

The following are examples of the different types of encryption:

- Asymmetric – RSA and DSA public key
- Symmetric – DES, TripleDES, and RC2 private key
- Hashing – MD5 and SHA1

6 Strengths of .NET

- Provides tighter integration of tools and technologies than in previous versions of the Microsoft computing platform and other vendor offerings.
- Provides the highly productive, standards-based managed runtime environment for Web applications, smart client applications, and XML Web Services.
- Agile architecture promotes application integration across the internet and intranet.
- Leverages existing applications through application interoperability. Allows for conversion of existing application components to XML Web Services.
- Benchmarks and industry studies have shown significant improvement in application reliability, security, deployment and performance.
- Supports multiple languages, even within the same application.
- CLR facilitates component/class reuse, as base classes are inheritable from any language.
- CLR contains structured exception handling, security model (including evidence-based and role-based security), memory, thread and process management, and garbage collection (to reduce/eliminate memory leak scenarios).
- Handles much of the application "plumbing", allowing application developers to focus on business logic.
- Contains a rich component class library and built-in controls for common tasks, such as shopping carts, tree views, and tab views.
- ASP.NET is now JIT compiled using the CLR. It has been benchmarked over 3 times faster than legacy ASP. It introduces improved session state management and fault tolerance.
- Improved ADO model supports XML standards, disconnected data sets, and data caching.
- Allows for XCOPY deployment, eliminating "DLL Hell" with new applications.
- .NET is a natural progression from the current/legacy Microsoft web platform.

7 Weaknesses of .NET

- The .NET technology is a significant change/upgrade from the previous Windows DNA model, and will require learning on multiple levels.
- The .NET Framework is currently only available on the Windows platform.
- The .NET Framework is a new, immature technology. It will take time to understand its full potential.
- Much data on performance, reliability and scalability is extremely scarce or unavailable industry-wide.

8 Summary

As you can see, Microsoft is embracing the reality that more security is needed in an Operating System. .NET is a technology evolution built from the ground up with security in mind. With .NET, Microsoft is attempting to create a platform that will give both developers and administrators security control over their resources at a granular level. They are doing this by allowing their customers to develop more powerful code while decreasing the associated risk that goes along with it, such as buffer overflows. Compared to previous Operating Systems, .NET appears to be a step in the right directions. As with each OS before it, I'm sure .NET will have its share of problems along with security issues, but from the outside it finally looks like they are starting to take security more seriously. Is this the OS for you, only you can answer that question, but for me, I might just give it a try!

9 References

1. Curtin, Matt. Interhack. “Introduction to Network Security” March 1997.
URL: http://www.interhack.net/pubs/network-security (July 2002).

2. Kirk, Steve, And Boylan, John. “NET Application Architecture Panel Discussion: PDC 2001” Revised February 2002.
URL: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnbda/html/bdadotnetarch102.asp (July 2002).

3. Microsoft Corporation. “Microsoft .NET Framework Security Overview”
URL: http://msdn.microsoft.com/vstudio/techinfo/articles/developerproductivity/frameworksec.asp (August 2002).

4. Baltazar, Henry, And Dyck, Timothy. EWeek. “Windows Gets Security Boost” December 3, 2001.
URL: http://www.eweek.com/article/0,3658,s=1184&a=19301,00.asp (August 2002).

5. Connolly, P.J. InfoWorld. “Microsoft casts server .Net, but will customers bite?” December 7, 2001.
URL: http://www.infoworld.com/articles/tc/xml/01/12/10/011210tcmsnet.xml (July 2002).

6. O’Farrell, Neal. SearchSecurity. “Security training: A call to arms” December 14, 2001.
URL: http://searchsecurity.stage.techtarget.com/tip/1,289483,sid14_gci786584,00.htm (August 2002).

7. Munro, Jay. ExtremeTech. “The Microsoft .NET Development Environment” June 8, 2001.
URL: http://www.extremetech.com/article2/0,3973,473512,00.asp (August 2002).

8. Zhuang, Ting. Purdue University. “What is Microsoft .NET”
URL: http://icdweb.cc.purdue.edu/~zhuang/550portal/main.htm (July 2002).

9. Kuptz, Jerome. Lycos, Carnegie Mellon University. “.Net Framework Overview” January 11, 2001.
URL: http://hotwired.lycos.com/webmonkey/01/02/index3a_page2.html?tw=programming (August 2001)

10. Microsoft Corporation. “A Powerful Platform” June 21, 2002.
URL: http://www.microsoft.com/windowsxp/tabletpc/productinfo/overviews/Platform.asp (August 2002)

11. Microsoft Corporation. “.NET and Security” January 14, 2002.
URL: http://microsoft.com/net/develop/security.asp (August 2002)

12. Foundstone, Inc. And CORE Security Technologies. “Security in the Microsoft .NET Framework”
URL: http://www.microsoft.com/technet/treeview/default.asp?url=/technet/itsolutions/net/evaluate/fsnetsec.asp (July 2002)

13. Microsoft Corporation. “What is the Common Language Specification?”
URL: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpguide/html/cpconwhatiscommonlanguagespecification.asp (August 2002)

14. Microsoft Corporation. “Overview of the .NET Framework”
URL: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpguide/html/cpovrintroductiontonetframeworksdk.asp (August 2002)

15. Microsoft Corporation. “A Guide to Reviewing the Microsoft .NET Framework and Microsoft Visual Studio.NET beta 1”
URL: http://www.aictc.com/softwareDevelopment/VisualStudioNetFramework.pdf (August 2002)

16. Microsoft Corporation. “What is Microsoft .NET?” July 9, 2002.
URL: http://msdn.microsoft.com/netframework/productinfo/overview.asp (July 2002)

17. Microsoft Corporation. “Microsoft Announces "HailStorm," a New Set of XML Web Services Designed to Give Users Greater Control” March 19, 2001.
URL: http://www.microsoft.com/presspass/features/2001/mar01/03-19hailstorm.asp (August 2002)

18. Microsoft Corporation. “Microsoft Mobile Internet Toolkit Reviewer's Guide” April 23, 2002.
URL: http://msdn.microsoft.com/vstudio/device/mit.asp (September 2002)

19. ASPAlliance. “ASP.NET Web Application Security”
URL: http://www.aspalliance.com/aspxtreme/webapps/aspnetarchitecture.aspx (August 2002)

20. Microsoft Corporation. “Why ASP.NET?”
URL: http://www.asp.net/whitepaper/whyaspnet.aspx (August 2002)

21. Parry, Ed. SearchWin2000. “.NET Server: Does it belong in your future?”
URL: http://searchwin2000.techtarget.com/originalContent/0,289142,sid1_gci817291,00.html (July 2002)

22. Microsoft Corporation. “Why .NET? Getting Beyond Technology Roadblocks” January 14, 2002.
URL: http://www.microsoft.com/net/basics/whynet.asp (September 2002)

23. Microsoft Corporation. “System Security Cryptography Namespace”
URL: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpref/html/frlrfsystemsecuritycryptography.asp (August 2002)

24. Microsoft Corporation. Broken Link. Unable to get information. (July 2002).

25. Microsoft Corporation. “SOAP Toolkit 3.0”
URL: http://msdn.microsoft.com/downloads/default.asp?URL=/downloads/sample.asp?url=/msdn-files/027/001/948/msdncompositedoc.xml (September 2002)

26. Shostack, Adam. And Black, Scott. BindView Development. “Towards a Taxonomy of Network Security Assessment Techniques” July 2, 1999
URL: http://razor.bindview.com/publish/papers/taxonomy.html (September 2002)

27. Microsoft Corporation. “Microsoft .NET Glossary”
URL: http://members.microsoft.com/partner/products/net/microsoftnet/DotNetGlossary.aspx (September 2002)

28. Microsoft Corporation. “About ASP.NET”

URL: http://www.gotdotnet.com/team/asp/ (September 2002)

29. Microsoft Corporation. “Application Domain Hosts”
URL: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpguide/html/cpconapplicationdomainhosts.asp (September 2002)

30. Andrews, Chip. Clarus Corporation. “Activating DLL/COM Redirection” September 20, 2000.
URL: http://msdn.microsoft.com/library/techart/sidebyside.htm (September 2002)

31. Miller, Jim. Microsoft Corporation. “Microsoft .NET: The Vision”
URL: http://research.microsoft.com/programs/europe/events/dotnetcc/Version1/20010903-MSRCambridge.ppt (September 2002)

32. Microsoft Corporation. “.NET Framework Class Library, SecurityPermission Class”
URL: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpref/html/frlrfSystemSecurityPermissionsSecurityPermissionClassTopic.asp (September 2002)

Last Updated: October 2nd, 2012

Upcoming SANS Training

Click Here for a full list of all Upcoming SANS Events by Location

SANS CyberCon 2012	Online, VAUS	Oct 08, 2012 - Oct 13, 2012	Live Event
SEC 579: Virtualization and Private Cloud Security @ Bangalore	Bangalore, IN	Oct 08, 2012 - Oct 13, 2012	Live Event
SOS: SANS October Singapore 2012	Singapore, SG	Oct 08, 2012 - Oct 20, 2012	Live Event
SANS Gulf Region 2012	Dubai, AE	Oct 13, 2012 - Oct 25, 2012	Live Event
SANS Seattle 2012	Seattle, WAUS	Oct 14, 2012 - Oct 19, 2012	Live Event
SANS Baltimore 2012	Baltimore, MDUS	Oct 15, 2012 - Oct 20, 2012	Live Event
SANS@ Grid Security Conference 2012	San Diego, CAUS	Oct 16, 2012 - Oct 16, 2012	Live Event
SANS South Africa 2012 - Cape Town	Cape Town, ZA	Oct 26, 2012 - Oct 27, 2012	Live Event
SANS Chicago 2012	Chicago, ILUS	Oct 27, 2012 - Nov 05, 2012	Live Event
SANS South Africa 2012	Johannesburg, ZA	Oct 29, 2012 - Nov 06, 2012	Live Event
SANS Bangalore 2012	Bangalore, IN	Oct 29, 2012 - Nov 03, 2012	Live Event
FOR526 Beta	Denver, COUS	Nov 05, 2012 - Nov 09, 2012	Live Event
SANS Korea 2012	Seoul, KR	Nov 05, 2012 - Nov 13, 2012	Live Event
SANS Tokyo Autumn 2012	Tokyo, JP	Nov 05, 2012 - Nov 10, 2012	Live Event
SANS San Diego 2012	San Diego, CAUS	Nov 12, 2012 - Nov 17, 2012	Live Event
SANS Sydney 2012	Sydney, AU	Nov 12, 2012 - Nov 20, 2012	Live Event
SANS London 2012	London, GB	Nov 26, 2012 - Dec 03, 2012	Live Event
SANS San Antonio 2012	San Antonio, TXUS	Nov 27, 2012 - Dec 02, 2012	Live Event
European SCADA and Process Control System Security Summit 2012	Barcelona, ES	Dec 05, 2012 - Dec 11, 2012	Live Event
SANS Cyber Defense Initiative 2012	Washington, DCUS	Dec 07, 2012 - Dec 16, 2012	Live Event
SANS Egypt 2012	Cairo, EG	Dec 08, 2012 - Dec 20, 2012	Live Event
SANS Forensics Prague 2012	OnlineCZ	Oct 07, 2012 - Oct 13, 2012	Live Event
SANS OnDemand	Books & MP3s OnlyUS	Anytime	Self Paced