VISUAL GUARD

Role Based Access Control

- What is Visual Guard?
- Authentication
  Verifying the identity of the user
- Authorisations
  What a user can do in an application?
- Auditing and Reporting
  Visual Guard ready-to-use Auditing and Reporting features
- Visual Guard Security Tools
  Developer, Administrator and Auditor Tools
- How does it work?
- Technical Specifications

What is Visual Guard?

Novalys has been developing authentications and permissions solutions for business applications for 15 years. Based on this experience, we designed Visual Guard .Net, a solution that addresses dotnet application security issues, no matter how complex your environment.

Visual Guard is a Role Based Access Control System.
It adds security features to your projects.

- Manage users, memberships, roles and password policy
- Define user permissions (what end-users can do in the application)
- Authenticate users with Active Directory or database accounts
- Implement Single Sign-On
- Keep track of sensitive transactions (application logging and auditing)

Visual Guard .Net :

- Provides a user-friendly administration console, designed for non-technical staff
- Centralize the security of all your applications in one tool
- Does not require coding to define permissions
- Integrates easily in any dotnet application
- Supports Winforms, Webforms and Webservices, and any dotnet component.

Authentication

Authentication in RBAC consists of verifying the identity of the user through a two step process:

- Identification: stating who you are;
- Authentication: proving who you are.

Two type of needs:

- You need to create a list of user account/password from scratch.
- You already have an authentication system and would like to re-use it at application level.

The solution:

Visual Guard .Net supports login/passwords authentication.
A user account declared within Visual Guard is available for all your dotnet applications.

Authentication modes supported:

- Windows Accounts (local account or Active Directory account)
- Database Accounts (created by the DBA for users to access the database)
- Visual Guard Accounts (created and managed by Visual Guard)

You are using another authentication mode? (smartcard, etc...) contact us

Windows accounts/Active Directory

If you use the Windows authentication mechanism, passwords are created, stored and administrated in Active Directory.

- Visual Guard console enables to search for a user in Active Directory. When the user is found, Visual Guard .Net stores his security identifier (SID) in the repository. You will then be able to assign Visual Guard roles and permissions to this user.

  You can declare several users at the same time (as many as you want).
- You can automatically import some/all windows accounts from Active Directory into the repository: VG provides an API that allows an external program to create or modify new account

Single sign-on

If you use Active Directory to manage user accounts, you may want to implement a single sign-on process: once a user is logged on in a Windows session, any application opens without asking for further credentials.

Visual Guard supports Single Sign-On configurations for Windows accounts:

- Visual Guard .Net automatically gets the ID of the current windows user.
- VG checks if this user has access to the application.
- VG loads the corresponding role and apply its permissions to the application.
- The same process will repeat each time the user launches an application.
- As a result, the user does not enter his credentials when accessing an application.

Database Accounts

Visual Guard .Net supports authentication based on DBMS account for Oracle 8 and higher and Sql Server 2000 and higher.
Visual Guard .Net allows you to re-use user accounts and passwords defined in these DBMS.
More information about Visual Guard .Net integration

Visual Guard .Net Accounts

Visual Guard .Net has its own membership provider to manage user accounts and passwords. Credentials are stored in Visual Guard repository.

Password policy

Visual Guard .Net allows you to define a Password Policy for Visual Guard .NET accounts.

For instance, you can define:

- Minimum password length
- Minimum number of non alphanumeric characters
- The possibility to use again former passwords.
- Enforce unique email address for each user
- Enforce the modification of each password every x days
- Maximum number of consecutive failed attempts to enter password. User-id will then be disabled until reset through an authorized process.
- Number of grace logins (possibility to login x times after the account is blocked).
- And more…

Regular expressions are available to customize the verification of the password. You can set a list of compulsory characters (for instance at least one capital character, one small character, and at least a figure), a list of valid/invalid characters etc...

Authorizations

Authorizations define what a user can do in an application: Basically, you define what the user is allowed to see, do and modify in your applications based on his role. Different words are used to define authorisation: permissions, rights, restrictions, privileges….

There are two ways of defining authorizations:

- The most secure way is to forbid everything by default, and then grant permissions to open possibilities. This way, if you forget to define a permission, the user won't be able to do something he should, rather than accidentally do something he shouldn't.
- The faster way is to allow everything by default, and then you assign restrictions to forbid some actions. This way is faster because typically there are fewer restrictions than permissions.

As a result you end up with a complex role based access solution, costly to maintain and difficult to update.

The need

By default, applications include the code defining their permissions. Each time you define a permission, you go through the entire development process again (specification, coding, testing, deployment,…).

This is a sharp issue since:

- Applications are updated every 2 or 3 months, whereas permissions require much frequent updates.
- Achieving the application's functional requirements given the technical limitations of your security system can be very time consuming, indeed can turn out to be impossible
- Complex permissions are often identified when the applications are in production, requiring an immediate fix.

The solution: make permissions independent from the code

With Visual Guard .Net, you do not write code in the application to define permissions. Your code is dynamically modified in runtime:

- You can create or modify permissions without going through the entire development cycle of coding, testing, deploying, waiting for feedback…
- You can define permissions any time, even when the application is in production. They are effective immediately.

What type of permission can Visual Guard manage

There is no limitation to what permission you can implement with Visual Guard. Any change you want to make in your dotnet application and any restriction are possible.
You can create permissions on graphical components or business objects as well as objects which manage the access to the Database.

For instance, you can:

- Hide or disable fields, menu options, tabs, controls,…
- Switch a form into "read only"
- Filter data in a list
- Grant access to webservice
- Modify business rules…

Visual Guard can secure any dotnet components, for instance :

- GUI objects
- Non visual objects
- Dynamic objects
- SQL Statements

How does Visual Guard .NET defines permissions

Visual Guard uses the reflection mechanism provided by the .Net framework to modify the application. This allows managing permissions totally independent from the code.

Visual Guard provides several solutions to define permissions:

- **Properties Actions**: Visual Guard can list all the objects (graphical and non graphical) and their properties. The developer uses a Wizard to identify the object related to the permission and assign a new value to one of its properties (like "visible" = "false" if you want to hide a control). The definition of such permission is stored in Visual Guard repository. The application code remains unchanged. Visual Guard modifies the application at runtime according to this permission.

- **Script Actions**: a script action is composed of code that you write. This script is stored in Visual Guard repository and applied directly to your application. Again, the application code remains unchanged.

- **Testing permission in your application**: you can also define the permission in your application: you can write code in the application to verify whether a permission (or a role) is granted to the current user, and execute the permission if the test is successful.

- **Limiting a Method access**: you can define for which role a Method is accessible.

- **Limiting a Folder access**: you can define for which roles a folder is accessible (ASP.NET applications only).

The following page present the different solutions provided by VG to adapt your application to business requirements:
http://www.visual-guard.com/support/index.php?option=com_content&task=view&id=40&Itemid=43

When can you define and grant permissions

By default, we suggest a two-step process:

- Step 1: the development team use Visual Guard developer tools to define permissions. Each permission is given a functional name (allow editing customers, hide personal info...) to make it easy for administrators to understand it.

- Step 2: administrators (non technical persons) use Visual Guard administration tools to manage user accounts and grant them roles and permissions.

Auditing and reporting

The need

- Keeping track of who did what in your applications
- Monitoring sensitive/financial transactions.
- Reviewing existing user accounts, their roles and permissions.

Such features may be necessary to comply with:

- Internal business rules, specific to your company
- Legal requirements (Sarbanes-Oxley Act…).
- Certification requirements (ISO, CMMI, ITIL…)

The solution

Visual Guards provides the following ready-to-use features:

- **Reporting**: you can automatically generate detailed report about existing applications, user accounts, roles, permissions, etc.

- **Application auditing**: you can log any event in your application and review this log anytime. Visual Guard provides several types of filter to focus on the transactions/events you are currently reviewing.

- **Administration console auditing**: Visual Guard logs automatically several actions available in the console, such as:
  - Who logged / failed to log in to the console
  - Who granted a given permission
  - Console user account locked or unlocked

Visual Guard Security Tools

Developer tools

Visual Guard provide developer tools to:

- Create and modify permissions within a few clicks,
- Verify automatically the consistency between dotnet applications and permissions
- Search in the repository to find existing user accounts, permissions...
- Deploy new versions of a repository along with a new version of the application.
- Manage several versions of a repository when deploying the application,

Security Action wizard
When defining permissions, you will probably create property actions or script actions. The Security Actions Wizard assists you to create permissions in a few clicks, without adding code in your application. It offers detailed options to implement permissions that fit your ever-changing business environment as perfectly as possible with increased responsiveness.

Permissions description is stored in the repository. It is immediately available if the application is in production. No need to go through the entire development cycle (changing the code, testing, deploying…).

Permissions consistency
When a permission is related to a component, any change in this component may affect the permission and generate regression bugs. Therefore, each new version of the application requires a full verification of all the permissions.
Visual Guard .NET can verify automatically that permissions and application code are perfectly matching.

Deployment wizard
Each new version of the application may come with new permissions and a new repository. You need to deploy this repository, without disturbing the data already entered by administrators in the previous repository.

Visual Guard .Net provides a wizard to deploy new permissions in a production repository. Visual Guard will merge automatically the new and previous permission sets, without loosing existing user accounts, roles and permissions.

Security Data Versioning
When a new version of an application is being deployed, some users are using the new version while others are still using the old one. Both the existing and new permission repositories have to be available during the migration process.

Visual Guard .Net can manage several versions of the permission repository. It allows a progressive deployment of the new application and its repository. Users can access both existing and new permission repository, according to their version of the application. You can switch smoothly from one version to another without blocking end user or rushing the process.

Global Search
When maintaining the permissions, you may have to find permissions related to a specific user or business keyword. For example, if you modify a control in your project, you need to list all permissions related to this control in order to adapt them.

Visual Guard .Net provides a global search features to browse the repository and find all items related to a given keyword. You save time while maintaining the permissions by getting immediately a comprehensive and reliable result.

Administrator tools

The administration forms
You need to manage user accounts, roles and permissions on a daily basis with accuracy and responsiveness.

Visual Guard .Net provides 2 options for such administration purposes:

- **Visual Guard Console**: a user-friendly application designed for non-technical users.
- **Visual Guard API**: you can develop your own administration form and call Visual Guard API. It will provide ready-to-use features to manage user accounts and grant them roles and permissions. Such form will be integrated in your application, with your default look and feel, so that user with administration roles can use it easily. Visual Guard API supports both winforms and ASP.NET administration forms.

Who can be Visual Guard administrator?
Because Visual Guard administration forms do not require any technical skills, you are free to delegate the administration to whoever is in the best position to do it. Administrators may be security people, system administrators, managers in charge of a department or a remote site..,

This choice is no longer related to technical ability. You can comply with company security policy and optimize internal business processes.

You can even define several levels of administrators, each one with more or less permissions to use the administration forms. For instance, you may have a central, super-administrator who can create permissions and roles, and local (remote sites ?) administrators who can only create user accounts and grant them existing roles, without the possibility to change these roles.

As a result, you get more flexibility: administrators are totally independent. They can change permissions, roles and user accounts anytime, and these changes are immediately effective. On the other end, you release the development team from daily administration.

Shared roles
You may need to define roles that should be available across your entire organization, whatever the number of applications you have (exisiting or future).

Visual Guard .Net provides Shared Roles grouping permissions for several applications. You do not have to manage a role for each application anymore.

Auditor tools

An Auditor role is available when using Visual Guard Administration console. This role restricts the features available to the one you need as an auditor, such as:

- Exploring the repository in read-only mode
- Generating detailed reports about the existing permissions, roles and user accounts
- Reviewing application logs to monitor specific or sensitive transactions
- Reviewing administration logs to track who gave which authorizations to whom.
- …

How does it work?

What's in Visual Guard

Visual Guard Repository

The repository stores user accounts, passwords, roles and permissions. It can centralize the permissions of all .NET applications (winform, webform, web services), as well as all user accounts, in a single and secured place.

Note: you may let Visual Guard .Net create and manage its own user accounts or you can re-use existing accounts created in Active Directory or in your Database.

Visual Guard .NET supports the following type of storage:

- VG supports SQL Server and Oracle for users, permissions and roles storage
- VG provides proprietary encrypted files for users, permissions and roles storage

- In case you choose to re-use Windows Accounts for Single Sign-On & authentication purposes, credentials are stored in Active Directory

You do not have to change your application database when integrating Visual Guard .NET. Visual Guard provides a wizard to create its own repository. You may choose between creating the corresponding Visual Guard .Net tables in your application database or in an independent database.

Visual Guard runtime

Visual Guard runtime is:

- Composed of .NET assemblies
- Integrated and deployed with your application
- Communicating with the repository to verify user identity and retrieve its permissions
- Adjusting dynamically your application to the user permissions

Integrating Visual Guard for .NET

Integration is fast and easy:

1. Adding Visual Guard run-time to your project and activate Visual Guard security services (just a few lines of code required).
2. Implementing Visual Guard login form (or use your own login form).
3. Create a security repository and declare your project in this repository.
4. Compile and deploy the application.
5. Defining, for each application, the related permissions with Visual Guard Console. The initial permission list is usually defined by the development team before the application goes to production. But permissions can be created and modified anytime, even after deployment. They are immediately effective for the application.
6. Managing user accounts, granting them roles and permissions (usually done by administrators. Of course it can also be done by the development team.

For more information about the integration process, click here...
Would you like to try Visual Guard with your own application? click here...

What happens at runtime?

- The end-user provides its credentials (except if a Single Sign-On process is activated)
- Visual Guard verifies the user identity
- Visual Guard connects to the security repository and retrieves the user permissions.
- Visual Guard run-time adjusts the project according to these permissions.

Example: when a form opens, Visual Guard may hide controls and filter a list (same features supported for ASP.NET, winform and webservices).
For more information, you can read Visual Guard Getting Started Guide

Visual Guard Technical Specifications

Applications supported

Visual Guard supports all Dotnet applications:

- **Winforms** (written with VB.NET or C#)
- **Webforms** (ASP.NET and ASP.NET 2.0)
- **Web services** (2.0 and 3.0 WCF)

Visual Guard .Net secures all Dotnet components (GUI, non-GUI, dynamic...)
Visual Guard .Net supports .Net framework 1.1, 2.0, 3.0 and 3.5

It also supports PowerBuilder versions 5 to 11

Visual Guard Repository

You can create the repository in:

- **Oracle** (8i or higher)
- **MS SQLServer** (2000 or higher)
- Visual Guard.NET encrypted files (for small applications).
- For other databases, contact us...

User authentication

Visual Guard .Net supports user authentication based on:

- Windows Accounts (Single Sign-On available for winforms, webforms and webservices)
- **Active Directory** (Active Directory User Accounts and Groups)
- Database Accounts (username and password stored in a DBMS)
- Visual Guard Accounts (username and password defined with Visual Guard).