5250 Application Firewall

Match Your Application Software Exactly to Your Work Management Policy

By Olivier Crouslé

Olivier Crouslé

In this article Olivier Crouslé browses over the 3 steps of security for iSeries and presents a multi-tier security approach down to the internal level of application software.

Three Points of View on Security

Security on iSeries – could it be a wider subject? Security is a stronghold of this platform, but it's often a nightmare, for what it does as well as for what it misses. To begin, I suggest looking at it from the point of view of the three main positions involved in it: the security expert, the final user, and between them – the application manager.

For the security expert, there are two steps to reach a highly secure iSeries environment. For production data and programs, his/her security strategy includes the control of user access and the assignment of ownership and authorities, according to the local policy. "Please, Boss, give me a security policy to implement, or endorse mine", may be his motto. His objective is to control access to the system at minimal working level, as well as to keep trace ability against, for example, a disgruntled employee or an anonymous Internet hacker. So, first is access to the machine, technically the easiest step. Second is access to the objects, more linked to the functional logic, thus much more difficult to effectively implement. Actually there is a third step. That is, in a security expert's mind, related more to the application manager's responsibility. It is to manage what is available through the regular menu-driven business application.

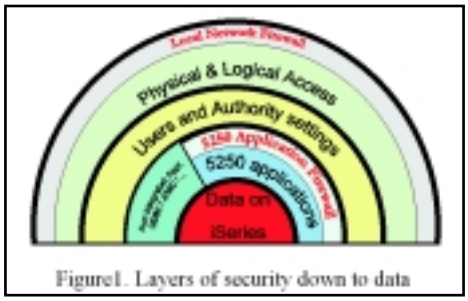


Figure1. Layers of security down to data

For the end-user, everything made available to him is virtually allowed. He is accustomed to many checking procedures that prevent him from making any mistakes, because Murphy's Law is more universal than Newton's law: in a production context and stress, if it may occur, it will. Moreover, when he is authorized on a screen to update a record and sees an underlined field he expects to be authorized to it. Then, with only *USE authority on the object matching this field, a confusion is created: either he calls for support when he does not succeed in modifying this record; or worse, he believes that he modified it when in fact he didn't. In such a case, security is felt a plague. You have enforced perfect object/field security, but now you have created a management problem.

In order to get sharp and efficient business production, the application manager must avoid any accidental mistakes and misunderstandings. He too, needs a good definition of each position and the required authority on the global information system. But his objective goes further – to reach the perfect fit between user's job/positions, business rules, and application software tools in order to gain on productivity and quality. He is the one interested in the "application firewall" approach that we will tackle in this article.

Intrinsic Security

Regarding the first 2 steps, frequent speakers for COMMON or TEC events, such as Carol Woodbury and Wayne O. Evans, have well explained most of its mysteries and wizardries. Don't neglect their level of expertise, because it is easier every day for end users to shortcut your security.

A few months ago, I did a full installation of OS/400 V5R1 along with Client Access Express V5R1 on workstations already equipped with Microsoft Excel 2000. You would be surprised to discover, as I did much more recently, two new wonderful icons in a dedicated Excel toolbar with a marvelous, not to say powerful, wizard to transfer data to and from your iSeries. For a developer, that is much better than DFU, or even WRKDBF, for creating a testing set of data. Most users are authorized to reach some data only through a menu-driven application, and if their object authority is implemented more loosely at the database level, you've opened the door to confidential data with direct access… Imagine leaving that in the hands of your final user. Such an experiment may convert you to the "Adoption of Authority" technique, like the famous exit programs which limit ODBC/JDBC/FTP access to your production database or prevent useless servers from starting automatically.

In today's global economy, with competition in a worldwide market, access to information is often a key factor. So, the first step is physical access to the iSeries, just like the door inside of your buildings, or through one of its multiple electronic ports.

> **Access protection from "anywhere" to your LAN**, is the job of the classical firewall server, as well as VPN and other encryption software, to virtually extend your entrusted LAN to "everywhere".
>
> **From your LAN to your iSeries**, is what your iSeries security configuration is for. Do you know which TCP or Host servers are started? Which ones are filtered with exit programs? What are the security related system values and password rules?

Great! Among 6.5 billion human beings and a few million computers in the world, you have now set security with every tool available to ensure you have granted some rights to everyone or everything that needs access to your machine and regained control on the channels used to access it.

The second step is object authorities settings. A regular user usually can change a record in a file, but is not allowed to delete the entire file. He is just allowed the use a program. Some very small iSeries shops exceptionally give *ALLOBJ rights to every authorized users because nobody around has the time, skill or money to adjust required authorities. When these companies grow up, the probability for mistakes soars. A third-party package or a good experience with the GO SECTOOLS menu may help. Rigor and clearly defined business rules are required to avoid such security settings from becoming a maze.

Application Firewall

Eventually, it is at a third step that the "application firewall" may be very useful for fine-tuning. What do you do when you have the perfect application database already available, but too many fields in it? Do you have an application with too largely available option "4=delete" or too many "F6=Create record" possibilities? Usually a package arrives with many tools to define users. To suit specifically the enterprise needs, add-ons are often developed for packages. The other way is the full in-house development that is perfect just for that. But the crucial point is how their evolution is managed, how to keep a specific development when a package upgrade is required and how fast you can implement a management decision.

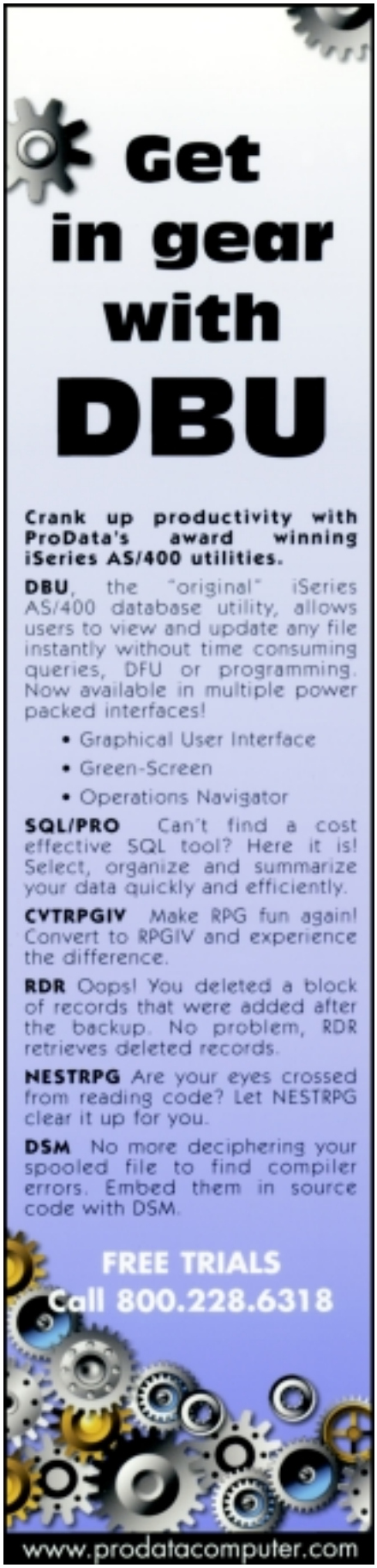
Constraints are time, money, skills availability and reorganization decision. For changes, will you bend your management rules to match the existing application, or will you always prefer to fit your application to your needs? How much time do you need to get a project accepted, processed then completed to implement the new rules? Or won't you be tempted to stick to the "status quo", while missing some productivity gains and eventually market share?

From Theory to Practice

Assume you have a human resources department application.

> Employees moving during the summer must notify you of their new address, phone number and so on.
>
> The Internet Service Provider of your home-working contractor's went bankrupt, so your partner must hastily notify you of his new electronic coordinates.

Such updates used to be sent by paper mail, then by e-mail to a clerk in the human resources department. Now, with the recent trend of wider systems openness, upper management has decided that everybody will be granted the responsibility of maintaining their own personal data, with limited access to their own records. Obviously, you are left with this task to implement. Easy. Your existing application has on its screen everything you need to update. You could just add an option to every user menu which calls the previous program on their own record... But wait! It allows wages input for raise implementation too! "BONANZA, we are allowed to choose our salary ourselves!"



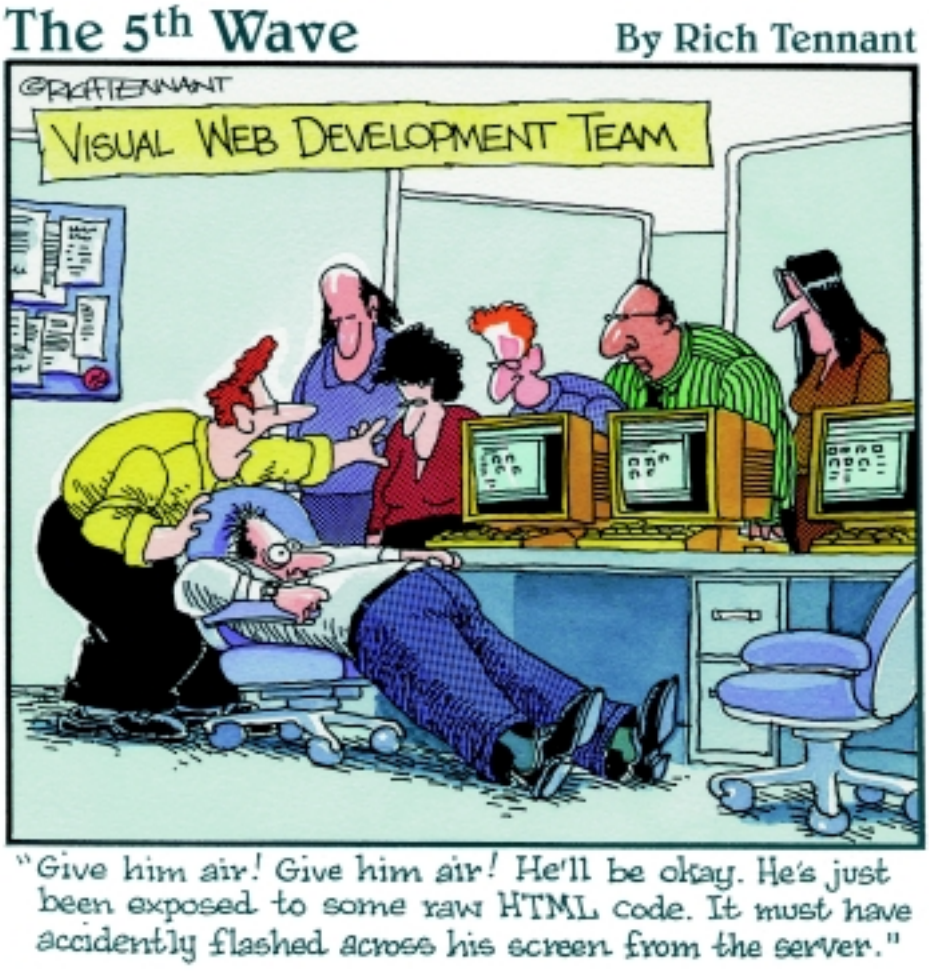
Alternatives:

"Classic": Duplicate or modify the code of the existing application.

"Modern": Allow direct access to the database with a glossy new Windows or JavaWeb application with a SQL/ODBC/JDBC connection!

"Shortcut": Work directly on the user interface to filter the existing application and sweep its exceeding features. That's the "application firewall" approach!

Most screen scrapers allow such an approach. They catch the 5250 application screen, either on an intermediate server or directly on the user workstation, then allow some customization. However, be cautious because if their only action is to change attribute to a "non-display" one (X'37'...), some emulators may consider black over black data is still data valuable to print! In any case, confidential data leaves the iSeries or at least is transferred in its IFS. You are sent back to reengineer security from step one with a new ASCII server, or worst, directly on the final workstation. Finally, only 2 products can manage to prevent field display from going outside of the QSYS library on the iSeries without any change to your current business application source code. The first one is the Webfacing tool from IBM. You need the DDS source for analysis and to invest in Websphere Application Server. The 5250 datastream will not be generated anymore and a pure customizable Java user interface is generated instead. The other one is a non-Java, very fast 5250-datastream filter acting really as an "application firewall": Cogiscreen by ASP-re (www.cogiscreen.com). This product works directly inside of your iSeries on the 5250 datastream, so it works even for dumb terminals and before any subsequent 5250 emulator or screenscraper. It can disable list options and function keys, and withdraw an output field completely from the display. It may even withdraw an input field without disturbing the existing program, because an unchanged field never sends back any datastream.



© The 5th Wave, www.the5thwave.com

A New Breed of Tools for the Application Manager

Let's take for example a purchase department application that gathers data from 3 other departments. A reorganization decides that from now on, every department will manage its purchases. But every department is only concerned by one third of the fields displayed on the input screen. It is decided it may see but must not alter what is relevant to the others. Here, it would be a "protect" function for display only that would turn off each excedentary input field for each department.

You can do that with Cogiscreen in just a few drag-and-drops at design time, and with only one command with a "profile" parameter to be inserted in the user's initial program. Such a quick filtering of existing application to decide who is prevented from getting access to parts of a genuine application with no source programming – that is the goal of the application firewall.

It should not replace object-level security settings, as I remind them to you. But it is a perfectly complementary approach because it allows you to reuse and maintain existing applications with more confidence and less confusion.

Although not self-sufficient to prevent any breakthrough, it eases the way to define regular procedures to access data, and grants a fine tuning level which was missing for business applications. In this sense, an application firewall can greatly improve functional security and kick up business productivity. T G

Olivier Crouslé *is consultant at ASP-re Inc. With a PhD from the Université Lyon 1-France, Olivier had joined ASP Group in 1998 where, for the past 2 years, he works in the R&D cell of ASP group in Montreal, Qc. His more recent assignment has been on the Cogiscreen family products for iSeries www.cogiscreen.com. Olivier can be reached at 514-392-0007 or by e-mail at oc@asp-re.com.*