# *Understanding Lotus Notes Security & Execution Control List (ECL) Settings*

Kurt E. Sanders GSEC Practical V1.4
May 15th, 2002 Option 1

## Abstract

**E**xecution **C**ontrol **L**ists (ECLs) are generally “not well understood” and a relatively “hidden” security option within the Lotus Notes client. When activated and configured properly with your corporate clients, the ECL can provide an excellent security defense to preventing the disastrous effects of defective or malicious code/active content. When overlooked, the Lotus Notes client will freely execute all active content sent without hesitation.

Lotus Notes has enjoyed being in the shadow of Microsoft® products, like Outlook™ and Internet Explorer™ when it comes to wide spread, corporate targeted, malicious code attacks. Being in the shadow has created a false sense of security for Lotus Notes system administrators and will eventually be exploited. Microsoft provides a basic execution control mechanism for preventing macros in the Office suite and digitally signed active content within Internet Explorer, but not at the granularity of control that Lotus Notes ECL's provides.

Now is the time to make sure that your Lotus Notes ECL's are properly set on your companies workstations. This article will prepare you, your system administrators and your clients for the next generation of hackers who will design “Email Neutral” viruses that can run freely on both Lotus Notes and Microsoft Outlook workstations. It will also cover how Lotus Notes V5 and V6 ECL's work, their importance in user workstation security, identify several limitations, and finally, how your company can centrally deploy and manage them effectively for your business.

## Table of Contents

## Summary

ECLs are only effective if they are implemented properly. While the changes in releases since 5.0.2 serve as gentle reminders about the presence and purpose of ECLs, it is up to Domino administrators to manage them effectively. This involves careful planning for who and what is trusted; thorough implementation of updated client ECLs; and ongoing maintenance of the Administration ECL, to reflect changes in trusted signers.

ECL's are a powerful workstation security tool when centrally managed from the

Domino server. ECL's can be refreshed with a known standard of authorized signatures which will relieve the end users of having to "Trust" active content sent to their workstation for execution. However, administration of the authorized signatures in the ECL's requires a comprehensive plan for managing development code.

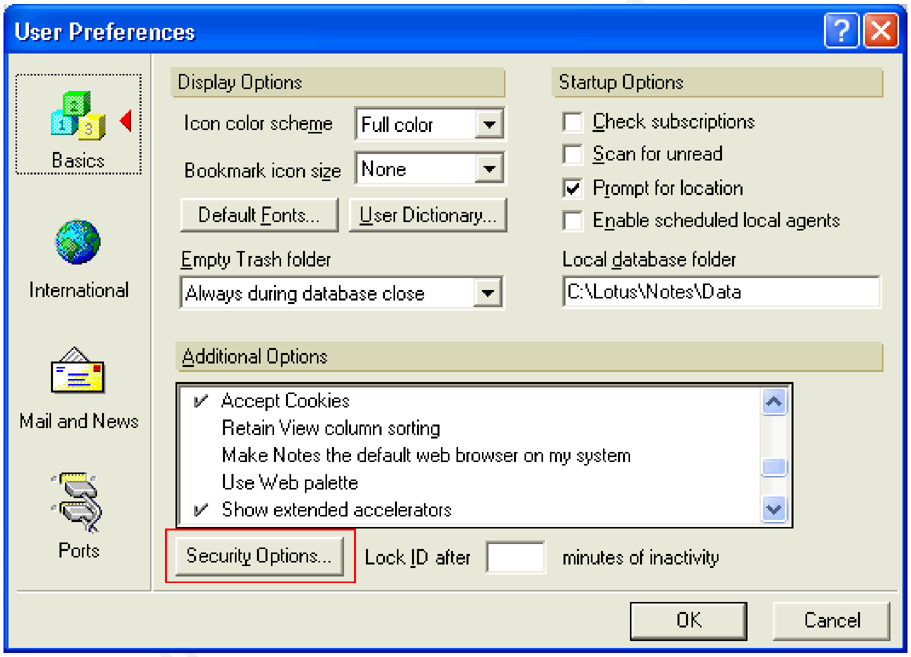
The Lotus Domino Release 6 Client will support Client ECL Logging and related operations. The ECL entries are logged in the Client log (log.nsf) in "Miscellaneous Events". Administrators, in release 6, will be able to push Admin ECLs to their clients dynamically, as needed. This solves the issue of those instances when clients get the default ECL (rather than the Admin ECL) during setup because they are disconnected from the directory, and provides for a more timely delivery of updates.

As always, there is no substitute for reminding end users to not launch or run executable code or any kind from unknown/untrusted sources. Users need to be reminded to carefully read the security warnings from the Notes ECL or from Internet Explorer ActiveX controls and contact the company support center when uncertain as to what they are allowing run on their workstation.

Lastly, In Notes, attachments can be viewed instead of Launched or Detached. This provides an excellent way to look at the content in most cases with a high degree of safety.

## When did the "Execution Control List" Arrive to Lotus Notes

ECLs were first introduced in 1996 to protect Lotus Notes™ 4.5 clients against the execution of potentially malicious code, whether it was executed from a local application, sent via email (*Can Lotus Notes Mail be attacked using LotusScript?)* as a embedded form or an Action button. The client ECL management features are located in the User Preferences dialogue screen under the button "Security Options". In order to enable this feature, Lotus shipped all design elements with the installation code as digitally signed by the development staff at Lotus as "Lotus Notes Template Development/Lotus Notes". The ECL was designed to limit the actions of formulas and scripts when they run on a workstation, based on the rights assigned for that signature.



Display Options

Icon color scheme: Full color

Bookmark icon size: None

Empty Trash folder: Always during database close

Startup Options

☐ Check subscriptions
☐ Scan for unread
☑ Prompt for location
☐ Enable scheduled local agents

Local database folder: C:\Lotus\Notes\Data

Additional Options

✓ Accept Cookies
Retain View column sorting
Make Notes the default web browser on my system
Use Web palette
✓ Show extended accelerators

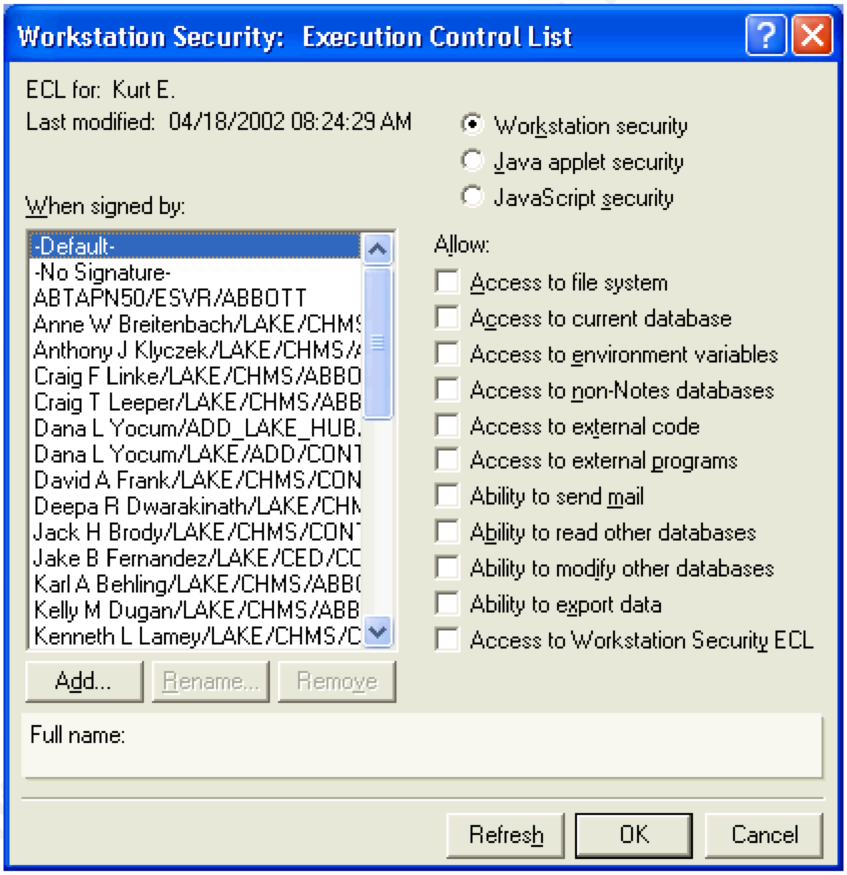
Lock ID after [ ] minutes of inactivity

**Figure 1 - User Preferences**

The (ECL) is a powerful part of the system administrator's security toolbox, yet it is frequently under-utilized at best, and overlooked at worst. Waiting quietly in the background on every client workstation, like a watchdog, the ECL is designed to protect user workstations against malicious code from unknown or suspect sources. The ECL determines whether the signer of the code is allowed to have its code run on a given workstation, and defines the extent to which the code has access to various workstation functions and is gated by the workstation security ECL.

## How Do ECL's Function?

For the purposes of this article, the term "active content" is used to refer to items that are verified and screened by the ECL. This includes formulas, scripts, agents, design elements in databases and templates, documents with stored forms, actions, buttons, hot spots, as well as malicious code (such as viruses and Trojan horses) -- in short, anything that can be executed on a user workstation.



Workstation Security: Execution Control List

ECL for: Kurt E.
Last modified: 04/18/2002 08:24:29 AM

● Workstation security
○ Java applet security
○ JavaScript security

When signed by:

- -Default-
- -No Signature-
- ABTAPN50/ESVR/ABBOTT
- Anne W Breitenbach/LAKE/CHMS
- Anthony J Klyczek/LAKE/CHMS/A
- Craig F Linke/LAKE/CHMS/ABBO
- Craig T Leeper/LAKE/CHMS/ABB
- Dana L Yocum/ADD_LAKE_HUB.
- Dana L Yocum/LAKE/ADD/CONT
- David A Frank/LAKE/CHMS/CON
- Deepa R Dwarakinath/LAKE/CHM
- Jack H Brody/LAKE/CHMS/CONT
- Jake B Fernandez/LAKE/CED/CC
- Karl A Behling/LAKE/CHMS/ABB(
- Kelly M Dugan/LAKE/CHMS/ABB
- Kenneth L Lamey/LAKE/CHMS/C

Allow:

□ Access to file system
□ Access to current database
□ Access to environment variables
□ Access to non-Notes databases
□ Access to external code
□ Access to external programs
□ Ability to send mail
□ Ability to read other databases
□ Ability to modify other databases
□ Ability to export data
□ Access to Workstation Security ECL

Full name:

**Figure 2 - Workstation Security: ECL**

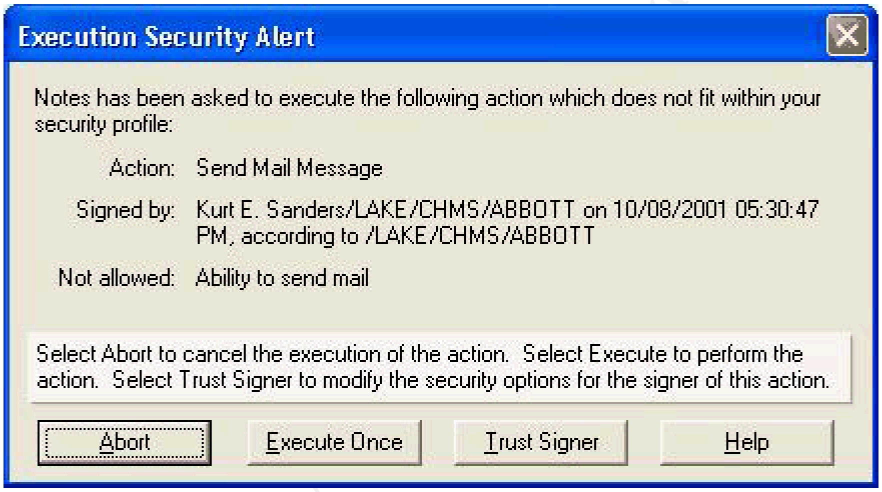
ECLs list trusted authors of active content (See example Figure 2). In Notes, database design elements, formulas, scripts, and other active content are signed with the ID of the user who created it or last modified it. In order for active content to be trusted, and thereby allowed to run on the workstation, the signer must be listed in the ECL.

For each signer listed in the ECL, workstation security settings can be enabled for access to protected operations, such as the ability to access the workstation file system or external programs. For a description of the workstation security options, see the Workstation access options sidebar. Although this article concentrates specifically on workstation security ECLs, descriptions of Java and JavaScript security ECL options are also provided in the sidebar.

Note the list of signers in the ECL dialog shown in Figure 2. You can see that the "Default" entry (highlighted) does not have any workstation security options enabled. This setting creates a “Prompt” level for any active content that does not support or have a digital signature assigned.

When active content runs on a user workstation and attempts a potentially harmful operation, several things happen. Lotus Notes verifies the code is signed, looks up the signer of the code in the client's ECL, and then checks the signer's ECL settings to determine whether the action is allowed. If the signer of the code is listed in the client's ECL and the appropriate setting is enabled, the code is executed. This code signing verification procedure does take some of the speed of execution for the active content at the trade-off of security.

If the active content attempts an action that has not been enabled for its particular signer in the ECL, or if the signer is not listed in the ECL, an Execution Security Alert (ESA) is generated (See Figure 3). The ESA specifies the attempted action, the item's signer, and the ECL access option that is not allowed.



Execution Security Alert

Notes has been asked to execute the following action which does not fit within your security profile:

Action: Send Mail Message

Signed by: Kurt E. Sanders/LAKE/CHMS/ABBOTT on 10/08/2001 05:30:47 PM, according to /LAKE/CHMS/ABBOTT

Not allowed: Ability to send mail

Select Abort to cancel the execution of the action. Select Execute to perform the action. Select Trust Signer to modify the security options for the signer of this action.

**Figure 3 - Workstation Security: ECL**

In the example dialogue box above (See Figure 1), I have trusted several developers that have created active content applications that I have selectively allowed the code to run on my workstation. A shortcoming of the Lotus Notes ECL is that when you accept a person's active content in one application, you have allowed that same person to have the same level of access for any other active content they will send to you in the future. The "trust" explained next, is not on a application by application basis, but on a author by action basis. When users see an ESA dialogue box, they are presented with a detailed alert and three options as shown on the right.

It is pointed out in the Execution Security Alert, that an action which is authorized is being requested by an active content component. It describes the action, the author's identity and what action is being requested to perform. Note that you cannot get access to view the running code to determine who they are sending an e-mail to, or what information is being sent. This, in my opinion, would be a valuable enhancement.

The three options (less the obvious HELP button) that are presented are as follows:

**Abort** -- Cancel the execution of the action in question. Leaves the end user with no other message.

**Execute Once** -- Perform the action, but doing so does not modify the ECL configuration. If the same action is attempted by the same signer in the future, the ESA appears again. This is the preferred choice if you trust the developer for this single program action execution.

**Trust Signer** -- Performs the action for the signer and modifies the ECL configuration, adding permission for the signer to execute the action anytime. This is like providing your credit card credentials and expiration code to a friend. You want to be certain that you really "Trust" the person in every similar action they will perform from now and for any future active content activities.

The ESA shown above was taken from my workstation that uses the ECL options shown earlier in this article. The active content is this case is a **mail message** that includes a button that performs a **Mail Send**. Note that while the active content is signed, the signer, in this case myself, is not trusted in the ECL so the action is disallowed. (The "No Signature" entry in the ECL signer list covers both unsigned code and code that is signed by an identity or organization that can't be authenticated.). If the user were to click "Trust Signer", the signer would be added to the ECL, and the **"Mail Send"** action would be enabled for that signer.

As I stated above, trusting unsigned content is extremely risky, and creates a security hole that allows potentially harmful code, malicious or otherwise, to access user workstations. Trusting signed active content from other organizations is also risky, as merely having a signature doesn't make an item trusted. Before adding an active content author to your ECL, you must decide if you trust the author has created and tested their "Safe Code".

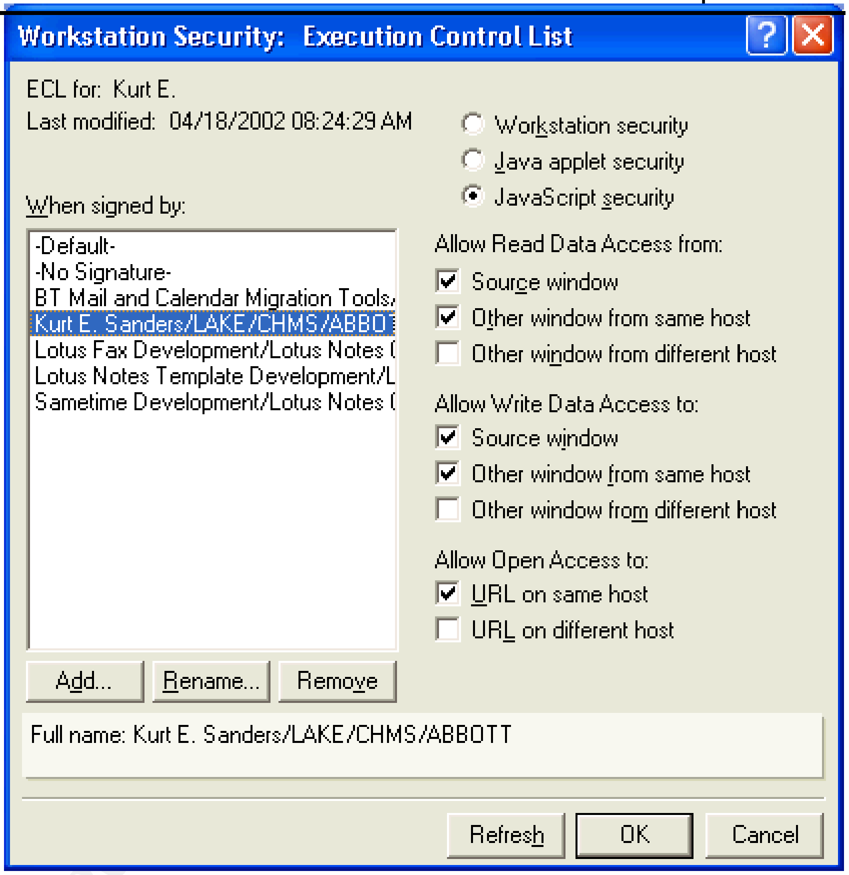
A shortcoming in the "Trust" is allowing your end users to directly modify their workstations ECL's. This is a potentially dangerous action left unchecked. End users sometimes view these confusing "Error" dialogue boxes as simply nuisances. It is important to review the ability to restrict end users from having the ability to access their workstations ECL's. Allowing your end user population to modify their ECL's will allow most all actions to be performed without prompting after a short period of time, effectively diluting the ECL's security role. This is what Email hackers are hoping for!

Administrators can also reset the ECL to disable all workstation protection (in effect, restore the pre-5.0.2 defaults) before deploying end-user ECLs during client setup. This means that users would stop getting ESAs, as restoring the default settings has the same effect as allowing users to always "Trust Signer." Users can also edit their ECLs, once the client has been setup, to restore the pre-5.0.2 default settings. In both cases, however, this leaves user workstations open to potential security problems.

## Java Security

The options that are presented for ESA are different depending on the active content that is presented to the workstation (See Figure 4).

As you can see, the granularity of security is more robust within the Lotuscript language content. One would expect that Lotus will strengthen these offerings in the Rnext release since Java plays such an important role in the emerging development platform for the hacker community.



ECL for: Kurt E.
Last modified: 04/18/2002 08:24:29 AM

○ Workstation security
○ Java applet security
● JavaScript security

When signed by:
-Default-
-No Signature-
BT Mail and Calendar Migration Tools/
Kurt E. Sanders/LAKE/CHMS/ABBO
Lotus Fax Development/Lotus Notes C
Lotus Notes Template Development/L
Sametime Development/Lotus Notes C

Allow Read Data Access from:
☑ Source window
☑ Other window from same host
□ Other window from different host

Allow Write Data Access to:
☑ Source window
☑ Other window from same host
□ Other window from different host

Allow Open Access to:
☑ URL on same host
□ URL on different host

Full name: Kurt E. Sanders/LAKE/CHMS/ABBOTT

**Figure 4 - Java Security**

## Stored Form Vulnerability

There is a standard Notes feature that a form is programmable. Therefore, for example, anyone with sufficient knowledge can create a form with "Store form in document" enabled, place some programming in, for example, the Form's PostOpen event, create a document, and mail it to another Notes user.

If the receiving database (the mail db, but it could be any db) has "Allow use of Stored Forms" enabled, and if the user's Execution Control List (ECL) permits it (or if the user allows execution to continue after receiving an Execution Security Alert), then the PostOpen code, which could potentially contain destructive instructions, will execute when the document is opened.

## Centrally Distributing ECLs from a Domino Server[1]

There are two kinds of ECLs:

1. Administration ECL, which resides in the Domino Directory (names.nsf).
2. Workstation ECL, which is stored in the workstation's Desktop file (desktop.dsk or desktop5.dsk).

In most cases, the Administration ECL is the template for all workstation ECLs. During the installation of the first server in the domain, the Administration ECL is created with default settings. Subsequently, whenever a new client is set up, a copy of the Administration ECL is created locally on the user workstation. The current Notes user ID is also added to the local ECL, with all access allowed. For example, when Betty Doe's Notes client is being set up, Betty Doe is automatically added to the client ECL signer list. If the home server is unavailable at setup time, such as when a user is disconnected, a default ECL is created.

ECL's are not static and they are designed to be reconfigured to meet changing security requirements. Administration ECL's can be edited through the Domino Administrator client. There are several ways to update user ECL's, by:

1. Editing the user ECL dialog.
2. Clicking Trust Signer (although this is not always desirable; see below).
3. Refreshing with an updated version of the Administration ECL. (For a description of this procedure, see "Recommendations for deploying tighter ECLs in the Release 5 release notes".

## ECL Enhancements for R5 and R6 Releases

### Version 5

Until release 5.0.2 of Notes/Domino, ECLs and signatures were provided as tools for administrators and users to implement as security policy dictated. In release 5.0.2, IBM/Lotus began concentrating efforts on fine-tuning ECLs to provide the optimum balance between security and usability.

IBM/Lotus will continue to provide resources in this area of ECL definitions and central management to match new security capabilities in Lotus Notes to the new security holes being identified. The author expects that IBM/Lotus will build in a central reporting application that workstations could report back various security violations (Invalid Access, Invalid Logons, etc) for review by the security administrator(s).

### Version 6

Administrators can push Admin ECLs to their clients dynamically, as needed in Lotus Notes V6 (Notes, Domino, and Domino Designer 6 Pre-Release 2 Release Notes). This solves the issue of those instances when clients get the default ECL (rather than the Admin ECL) during setup because they are disconnected from the directory, and provides for a more timely delivery of updates.

The Lotus Domino Release 6 Client now supports Client ECL Logging and related operations. The entries are logged in the Client log (log.nsf) in “Miscellaneous Events”.

1. The results of the Execution Security Alert (ESA) dialogs are logged. In addition there are additional ESA details that are logged. These details include info about the code that caused the ESA such as: the design type, design title, NoteID, Database Title and Path.
2. ECL modifications on the Client are logged. This log entry includes info on which ECL was modified, the ECL entries changed, added or deleted and the rights that were granted or revoked. This logs all ECL modifications resulting from such operations as dynamic ECL update, programmatic ECL refresh (@ECLRefresh function), setup ECL refresh/creation and manual ECL changes made in the ECL Editor or through the Security Panel.

## Default ECL Settings

A major change in R5 was the change in the ECL default settings. Previously, default ECL settings favored a more open configuration. They enabled all access options for the following signatures:

1. Default -- Trusts code signed with any signature
2. No Signature - Trusts unsigned or unauthenticated code
3. (UserName) - Trusts code signed with the user's ID (user ID that was added when the client ECL was first set up)
4. Lotus Notes Template Development - All Notes templates are signed with this ID, and this signature is trusted by default

If administrators failed to supply a Administration ECL with different settings, users would not get any ECLs; however, this meant that workstation security was, for all intents and purposes, nonexistent.

For release 5.0.2 and above, the default settings were now set "tight" instead of open, meaning that the access options for signatures not known to be trustworthy have been disabled. The new default ECL settings do not allow access to protected operations for unsigned or untrusted formulas and code. Consequently, secure ECL defaults are implemented for new domain and client installations, as well as for domains that never modified their original default Administration ECL.

Note that the secure ECL defaults are applied automatically only during setup of new client ECLs. To implement the secure defaults for existing (pre-5.0.2) clients, Administration ECLs should be updated with the secure settings and the

@RefreshECL function can be used to "push" updated Administration ECLs to existing clients.

When using the new default ECLs in R5.0.2 and above, users will be seeing ESAs with far greater frequency than ever before. Both active content that is signed and trustworthy, and that with untrusted or no signatures, will produce warnings unless remedial action is taken, either by updating the Administration ECL or clicking "Trust Signer."

## Execution Security Alert Choices

There was a small user interface change made to the Execution Security Alert for 5.0.2, as well. Prior to 5.0.2, if users opted to Trust Signer, they were also prompted to trust a signer's entire organization. This option was removed in 5.0.2, because while it might be necessary to trust a signer in order to run something on the workstation, it is not necessary to enable the same options for the signer's entire organization.

## What's in a Digital Signature

Most design elements that have executable code associated with them (for example, buttons, fields, formulas) can be digitally signed and have their signatures checked at time of execution (for example, when a button on a form is clicked). This enhancement makes sure that an organization can associate code with any of the many options available in the Lotus Designer, and not worry about users needing to leave a hole in their protection by granting "no signature" any access rights.

## Define Proactive Signature Policies

A signature policy is essentially a system for administrators to plan for, and configure in the ECL, those signatures that are trusted to sign active content, those that are not, and to what extent the trusted signatures can access protected workstation operations. Not only does a signature policy promote sound security practices, but ideally, it minimizes or negates the need for users to deal with ECLs.

Implementing a signature policy in your organization requires some time investment on the part of both the administrator and the organization; there is maintenance overhead for such tasks as centralizing signing, keeping administration and workstations ECLs updated, and so on. However, the benefits to be realized are significant should you wish to proceed down this pathway.

It is good information systems practice. ECLs protect user workstations from problems caused by active content, malicious or otherwise. It's possible to be exposed to code that was written with no malicious intent, but can still do damage because of coding errors. More and more, we see active content coming to our end users workstations from outside our internal development area. Lotus Notes allows end users to easily develop “Simple Actions” in the area of productivity, for example, “Click to add me to your calendar” or “Click to add me to your mailing list”. Setting up safeguards through a signature policy, such as only trusting certain users to sign/write code, reduces your exposure to both malicious and buggy code, and minimizes down time and support calls.

Having a pre-defined signature policy in place reduces the chances of making mistakes (such as trusting an unsigned formula), compared to when signatures are trusted ad hoc, such as when users react to ESAs. In addition, the existence of a signature policy is frequently a good vehicle for setting down end-user security policy and practices.

A well-implemented signature policy works in tandem with corporate security practices to protect corporate information assets. It encourages a conscious approach to enabling access to those assets.

There are two strategies to think about when considering a signature policy:
Managing and deploying user ECLs
Trusting active content

## Managing user ECLs

There are several options for managing and deploying user ECLs that range from minimal to maximum security, and may or may not require the implementation of a signature policy. An excellent reference table published by the Lotus Developer Domain Website provides clarity to all the available ECL settings for each allowable action. Whether and how you decide to implement a signature policy in your organization depends on several factors; namely,

1. Time and effort required for maintaining it,
2. Size and
3. Sophistication of the user community,
4. Nature of the business, and
5. Extent to which users communicate externally.

One way to manage ECLs is by not managing them. This is the least secure method of all. User ECLs are set so that everyone, even unidentified signers, is trusted. User impact is minimal, since, as a result, users will never get ESAs. So, while you as an administrator will rarely be bothered by someone who needs to have their ECL updated, there is a greater risk for damage by malicious code. This kind of scenario is appropriate in organizations with small user communities that have physical security and no connections to the outside world.

The next, more secure option for managing ECLs is the "ad hoc trusting" method, where who to trust is determined by examining what ESAs arise in regular use, and users are instructed by their system administrator about who to trust. As these decisions are made, the Administration ECL is updated, and user ECLs are refreshed accordingly.

The next couple of ECL management strategies require the use of signature policies. The first, which manages to incorporate a high degree of security and flexibility, relies on a set of policies and procedures. It includes guidelines for who is be trusted and who is not. There are procedures for keeping the Administration ECL up-to-date, and refreshing user ECLs regularly as the Administration ECL is updated. Users are given clear instructions for reporting ECL warnings, and there are firm policies about never trusting signers ad hoc, or clicking "execute once." Consequently, when ESAs do occur, it is either because of a mistake -- for instance, someone distributed code using a non-approved ID, or a database design element happens to be unsigned -- or because it is an actual security problem.

The most stringent signature policy is that which does not allow users to modify their ECLs. This means that they cannot edit their own workstation ECL, nor can they run unsigned or disallowed code. Should they get an ESA, the only option is to abort the operation. Administrators can set this option in the Administration ECL, by disabling the "Allow users to modify" option. When the Administration ECL is copied to user workstations, the option disallows users from editing their ECLs. This type of signature policy works best for companies in which users run a small, tightly controlled set of applications.

## Trusting active content

An important aspect of a signature policy is defining a methodology for trusting signers, which takes into account signed content that comes from both within and outside the organization.

For active content that comes from external sources (for example, third-party Notes applications), and that will be deployed in an organization, administrators need to make sure that all signers associated with this code are trusted. You have these options:

1. Add the signatures provided by the software vendor to your list of trusted signatures on your Administration ECL.
2. Sign all new databases with an approved internal ID, using the Admin Tools - Sign utility for signing databases. An admin utility is provided by Lotus that can take a database template and sign all the design elements with a new

signature. In Figure 1, we have elected to use a central Notes server (ABTAPN50) for enhancements to the mail template.

For active content that is created internally, Lotus offers the following approaches:

1. Create special signing IDs, which exist for the sole intent of signing databases, templates, and code for ECL purposes, and
2. Give the IDs rights to run restricted agents and be included in Administration ECL. The IDs exist apart from admin IDs, and usage should be limited to those individuals authorized to sign content.

In this scenario, it is extremely important to control access to the signing IDs. When authorized individuals leave the organization, their signing ID should be disabled. Similarly, new individuals who are given signing authority would get a new signing ID.

Have a separate organizational unit within a organization for users who must sign templates and applications, and then create an ID in that organizational unit for each of those users (for example */Template Developers/Company Name). Users who create templates and applications should only use the IDs issued through the new organizational unit when signing their templates and applications. The Administration ECL can then be configured to trust any user in that special organizational unit.

Avoid using wildcard naming conventions on trusted signatures (such as */Company Name) for an entire organization. Wildcarding, in this instance, means that all users within that organization are trusted. This is not recommended, primarily because most users don't, or don't need to, create active content; moreover, having such a policy in place makes any stolen ID potentially harmful.

## Notes holes Unbarred at Def Con

According to the a presentation a Def Con(*K-062: Vulnerabilities in Lotus Notes Domino Aired at DefCon 8)*, there are more than 60 million Notes users worldwide, so even a tiny crack in Notes security has large implications -- and these are not such small cracks. We have yet to see a large scale attack at Lotus Notes from an external source, but we recognize that this area can be exploited using active content.

Most of the problems can be solved by updating clients and servers, correcting weak access control lists that are part of default installations, and installing patches as soon as they are available.

## Remove entries from a workstation ECL that do not appear in the

## admin ECL

To remove entries from a workstation ECL that do not appear in the admin ECL when refreshing using @RefreshECL, the flag to prevent users updating their own ECL settings must be switched on in the admin ECL. This IS logical. If a user is able to modify their own ECL settings then it is reasonable for them to have ECl entries that do NOT appear in the admin ECL. Hence @RefreshECL leaves them alone. However, once you restrict their ability, one expects that their ECL MUST match the admin ECL. Therefore, @RefreshECL strips out entries that do not match

## Setting ECL at user login

To reset the user workstation ECL with the Administration ECL the notes.net forum suggests to use the @RefreshECL formula or the Session.SetEnvironmentVar("ECLSetup","1",True) method from LotusScript. Both Methods are for use with a button in a E-Mail Message.

## Conclusion

ECLs are only effective if they are implemented properly. While the changes in R5 serve as gentle reminders about the presence and purpose of ECLs, it is up to Domino administrators and Security Managers to manage them effectively. This involves careful planning for who and what is trusted; thorough implementation of updated client ECLs; and ongoing maintenance of the Administration ECL, to reflect changes in trusted signers.

ECL's can provide a security checkpoint to contain active content from several development platforms. Lotusscript and Java are two such language areas. Currently, there is a lack of a central reporting feature in R5 that would help security administrators quantify the alerts at the workstations.

As a general rule, do not launch or run executable code or any kind from unknown/untrusted sources. Do not ignore security warnings from the Notes ECL or from Internet Explorer ActiveX controls.

Lastly, attachments can be viewed instead of launched or detached from Lotus Notes providing a secure level to work with the attachments that could contain active content.

## References

1 LDD User, "ECL Management & Deployment Tool", 04/10/2000
URL: http://www-10.lotus.com/ldd/sandbox.nsf/ecc552f1ab6e46e4852568a90055c4cd/8ead2c26ce06cfaf852568bd00717201?OpenDocument&Highlight=0,ecl

2 Farrow, Rik. "ITworld.com Security Watch -- Keeping an eye on network security" , 8/4/00
URL: http://www.itworld.com/App/4150/ITW1898/

3 US Dept of Energy CIAC, *"K-062: Vulnerabilities in Lotus Notes Domino Aired at DefCon 8", 8/2/2000*
URL: http://ciac.llnl.gov/ciac/bulletins/k-062.shtml

4 *Duan, JP. "Can Lotus Notes Mail be attacked using LotusScript?", 11/27/2001*
*URL:* http://service4.symantec.com/SARC/sarc.nsf/info/html/can.lotus.notes.mail.be.attacked.using.lotusscript.html

5 IBM, *"Lotus Notes and Domino Reduce the Risks of Virus Attacks",*
*URL:* http://www.lotus.com/developers/itcentral.nsf/wdocid/67BEC14A50E33DE9852568E400604A0A?OpenDocument

6 *IRIS Archives, "ECL access option risk levels"*
*URL:* http://www-10.lotus.com/ldd/today.nsf/f01245ebfc115aaf8525661a006b86b9/90062e265849aa75852568310079ad10?OpenDocument

7 Lotus Corporation, Various graphics for workstation control settings and dialogue boxes, 5/1/2002,
Release 5.05 9/02/2002

8 Lotus Corporation, "Restricting Execution Access", "Lotus Domino & Notes Client Workstation Help", 9/02/2002, Release 5.05

9 Berlind, David. "Collaboration is our advantage", 1/24/2002
URL: http://techupdate.zdnet.com/techupdate/stories/main/0,14179,2842256-4,00.html

10 Lotus Notes Net Documentation Library, "Notes, Domino, and Domino Designer 6 Pre-Release 2 Release Notes (English)",
http://doc.notes.net/domino_notes/Rnext/readmePR2.nsf

11 Ludlow, David, "New flaw discovered in Lotus Domino", 2/23/2001, URL:
http://www.vnunet.com/News/1118193