REQUEST FOR INTERPRETATION BY THE NYS BOARD OF ELECTIONS

Requestor(s)	**SysTest Labs Inc.**

Request Date	**10/03/2008**

Requestor Contact Information *(Name, telephone, fax, mailing address, & email address)*	**R. Reed** **303-575-6881** **216 16th St** **Suite 700** **Denver, CO 80202** **rreed@systest.com**

NYS Election Law, Guideline, or Other Issue to be Clarified *(cite specific reference)*	VVSG Vol I, Section 5.2.3a Software Modularity and Programming: Each module shall have a specific function that can be tested and verified independently of the remainder of the code. In practice, some additional modules (such as library modules) may be needed to compile the module under test, but the modular construction allows the supporting modules to be replaced by special test versions that support test objectives. VVSG Vol II, Section 5.4.2 Assessment of Coding Conventions: Has all assert() statements coded such that they are absent from a production compilation. Such coding may be implemented by ifdef()s that remove them from or include them in the compilation. If implemented, the initial program identification in setup should identify that assert() is enabled and active as a test version Section 6209.2 G. Any submitted voting system's software shall not contain any code, procedures or other material which may disable, disarm or otherwise affect in any manner, the proper operation of the voting system, or which may damage the voting system, any hardware, or any computer system or other property of the State Board or county board, including but not limited to 'viruses', 'worms', 'time bombs', and 'drop dead' devices that may cause the voting system to cease functioning properly at a future time.

Statement of Ambiguity	VVSG Vol II 5.4.2 and VVSGVol I 5.2.3aare the only sections of the VVSG where there are any distinctions made between “production compilation” and “test versions” of software. While these particular sections reference the use of conditional compile statements, the same rationale could be extended and applied to any potential security exposure through the introduction of “conditionally compiled” code “test” code or “unused” code. Section 6209.2 identifies “any submitted voting system’s software”, but does not identify if this is the “production compilation” for the Trusted Build or the “complete source code” delivery to the VSTL which could include any “conditionally compiled” code, “test” code or “unused” code. The Ambiguity impacts the review process for the code. • If from Section 6209.2 and VVSG Vol II 5.4.2, it may be categorically stated that there is a difference between “production compilations” of code used for the Trusted Build and “test versions” of code, and that test versions of code are allowable in production compilations, , then the focus isn’t just on the source code review for -“conditionally compiled”, “unused”, or “test-only” code. Instead, the focus is also on the review of all of the supporting files in the build process and the combinations of compiler switch settings. This will require a complete review of all compilation switches and a confirmation that all “conditionally compiled”, “unused”, or “test” code is NOT being compiled as part of the “production compilation”. • If Section 6209.2 means “any delivered software prior to compilation” must be “production” level code, then “conditionally compiled”, “unused”, or “test-only” code can not be delivered as part of the submitted voting systems software. The focus would be to review the source code to validate that all “conditionally compiled”, “unused” and “test-only” code is not present in the production level code, and the compiler switch review would confirm that all code provided is compiled and used in the Trusted Build for the voting system.

Facts Supporting Ambiguity	In an independent analysis of compiler flag settings by SysTest prompted by NYSTEC questions on compiler switches, an execution trace for a given compiled module (i.e. cfload) was followed from initiation to compilation to determine if a security patch was bypassed. The result was that the flags for cfload were

	all set correctly and the security patch was not bypassed. The more significant finding was that through the manipulation of compiler switches in nested makefiles, the end result of the compiled code *could* be impacted, and that compiled binaries could be produced that did not exactly reflect what was provided in the source code through compiler switch manipulation. The current VVSG and NYSBOE requirements (NY Laws, 6209 and BMD) do not require the settings for compile flags to be set as “release-level”. They do not specifically prohibit “conditionally compiled” or “unused code. They do not specify that all compiler flags must be traced from initiation to final compiled binaries to validate all make files and build files compile the source code exactly as reviewed.

Proposed Interpretation	<u>SysTest Labs is suggesting one of the following Proposed Interpretation Options:</u> Option 1 6209.2 should be interpreted to mean any software code delivered prior to compilation shall be “production-level code”. This means that source code that is delivered to SysTest Labs would only be the code that is used in the production compilation of the Trusted Build, and it would be in a “production-level” state without “debugging code”, “test” code, “unused code”, or “conditionally compiled” code . This option would allow for diagnostics that could be used in the code if accessible and usable through the hardware on which the code is installed. Reasons: This would limit the volume of code that is being delivered and used for the “production-level compilation” of the Trusted Build to only that code that is absolutely used by the voting system. This would also improve the quality of the code. Positive impact: • Removes any identified “conditionally compiled” code, “unused” code, or “test” code from the source code prior to compilation. • it limits the code being maintained in the TDP, • it limits the number of compiler switches, • it decreases the amount of time spent performing the source code review and the build process analysis. • Any submitted code will be compiled during Trusted

	Builds in an equivalent setting for "release" mode • Would need to require the manufacturers to provide a list of all compiler switches and how they are intended to be used so that SysTest Labs may verify this through the examination of their build items. Negative Impact: • Manufacturers would need to take the time to review their own code and remove code that does not meet "production-level" standards. These are complex systems created through mergers and acquisitions and clean-up of the code would be costly to the manufacturers. • Manufacturers would have to create NYS-specific code releases, meaning that ongoing support would be expensive and the additional versions of code would be increasingly hard to maintain through configuration management. • Re-release of all impacted code. This would require a full source code review on code that is nearing defect fix completion. <u>Option 2:</u> Interpret 6209.2 as meaning "production compilation" for Trusted Build is the "submitted voting system's software as delivered". This means that the manufacturer could deliver anything as part of their Source Code, but the source code review would need to include a full review of the entire build process to confirm no "conditionally compiled", "unused", or "test" code is used as part of the production compilation Trusted Build. Positive Impact: • Source code is already reviewed • Would need to require the manufacturers to provide a list of all compiler switches and how they are intended to be used so that SysTest Labs may verify this through the examination of their build items. • Manufacturers would not need to create multiple versions of the code as the differences would be in the compiler switches. Easier to maintain, lower costs. Negative Impact: • Requirement for significant additional up-front time on the source code review prior to Trusted Builds to perform build analysis and confirm the compiler switches. • The amount of time spent to fully review the compiler switches will push out testing schedules.

	• There is a higher risk of less secure code built into the Trusted Build as the result of the resetting or unsetting of compiler switches, as illustrated by the cfload example if the switches had been set incorrectly. <u>Compiler Switch Review Process for Option 2:</u> The proposed process for performing the compiler switch review is included below should Interpretation Opion 2 be selected: Review of pre-compiler statements (including flags and conditionals) will be performed in the following manner on the source code and supporting materials (as defined by the Certified Voting System Software and Source Code Escrow Requirements, submitted by NYSTEC to the NYSBOE on January 21, 2008. Step 1 - All “# if” statements will be identified through keyword search in the source code only, and listed Step 2 – In the source code, the “# if” occurrences will be analyzed on a case-by-case basis to determine if the pre-compiler instructions lead to any functionality changes which can impact security in any way. • The only occurrences of pre-compiler instructions that will be deemed acceptable are those settings which deal specifically with error messages written to the log and system logging messages. • Unacceptable outcomes, which will lead to the generation of a discrepancy are: 1. Any changes to the functionality of the module 2. Any debug messages which present messages to the user rather than through writing to logs Step 3 – If the pre-compiler flags are not set in the source files then all of the supporting materials (build files, compiler artifacts) will then be analyzed with the list produced in Step 1 for every occurrence of the same pre-compiler flag. If the flag exists, a discrepancy will be generated for the removal of the flags in the supporting materials.

Please submit “Request for Interpretation” to:

NYS BOARD OF ELECTIONS
ELECTION OPERATIONS UNIT
ATTN: R. Warren
40 STEUBEN ST
ALBANY, NY 12207

OR:

election_ops@elections.state.ny.us

NOTE: Interpretations by NYSBOE will be provided in a separate, attached, document.