ISO/IEC JTC 1/SC 22/OWGV N 0073

Proposal to the ISO/IEC Project 22.24772: Guidance for Avoiding Vulnerabilities through Language Selection and Use

Date	21 June 2007
Contributed by	Larry Wagoner
Original file name	N0073.doc
Notes	Part 1 is document N0066, Part 2 is document N0067 and Part 3 is a revised version of N0068.

Part 1: Reason for the Proposal

In order to focus on the most usable and commonly understandable standard, we must tie to other standards. Common Vulnerabilities and Exposures (CVE) (http://cve.mitre.org) is a standardized dictionary of names for vulnerabilities. It is a community wide effort with representatives from commercial security organizations, government, and academia. CVE has been very successful in creating a standardized vocabulary for communicating information about vulnerabilities.

Common Weakness Enumeration (CWE) (http://cwe.mitre.org) is accomplishing a similar goal for software weaknesses. CWE is targeted at security developers and practitioners and is creating a common language for describing software security weaknesses in architecture, design or code. Many organizations have already declared their intent to be CWE compatible. CWE is not a mature product, but is improving considerably with each iteration.

CWE can be viewed as a dictionary, a classification tree, a pdf file, an XML file or an XSD schema. CWE has a high degree of granularity. For instance, there are 11 different kinds of buffer errors (stack based or heap based buffer overflows) that ultimately lead to 46 different types of buffer errors to include XSS and SQL injection (types of string errors that are types of buffer errors).

Within the CWE dictionary, there are several fields for each entry including an ID number, description, observed examples and applicable platforms. Applicable platforms states which languages the weakness can occur in. CWE contains weaknesses that are very common both in occurrence and exploitation to the very obscure and rarely, if ever, exploited.

There are several items that I propose related to CWE:

That we state any and all vulnerabilities in CWE language using CWE IDs and terminology.

That we select some reasonable number of the vulnerabilities to be addressed in our standard. We could probably cover a large number of vulnerabilities by selecting 75-100 of the items from CWE. This would cover a large percentage of real world vulnerabilities.

Advantages to the proposals:

1). We will be speaking the language that is being adopted by many commercial security organizations, government and academia.
2). We will be leveraging the work that has been done on the CWE. Although not mature, CWE is a reasonable basis in its present form.

Part 2: Derivation from Frequently Occurring Vulnerabilities to CWE

It would be impractical and simply a waste of time to attempt to address all of the weaknesses listed in CWE. Many are obscure, difficult to exploit and rarely occur in the real world. Therefore we should reduce the number to some reasonable number that would cover the bulk of those seen and exploited in the real world.

There are a couple of considerations that should be made as a list of the vulnerabilities to be addressed is developed. The obvious first consideration is to determine the most frequently occurring weaknesses. Fortunately some of this work has been done, although not to the granularity we need and not always reported in CWE notation. Two primary sources will be used to determine which weaknesses to focus upon. They are:

Christy, Steve, "Vulnerability Type Distributions in CVE," v1.0, 4 October 2006, http://cwe.mitre.org/documents/vuln-trends.html

OWASP Top Ten Project, http://www.owasp.org/index.php/OWASP_Top_Ten_Project

A second consideration is that some weighting should be made between ease of addressing and the potential and seriousness of exploitation. Many tools and even compiler warnings exist to detect code problems such as unused variables, dead code and memory leaks. These are more likely to be considered code quality issues than security issues. However, these shouldn't be ignored from a security standpoint. These indicate more serious structural problems with the code development process. If code quality is not up to a reasonable level, is there any hope that security will be? So some consideration will be made to ensure the easily detectable issues are addressed.

In order to derive a set of vulnerabilities to be addressed as a standard, the approach taken will be to use the paper by Steve Christy and then to use the OWASP Top Ten to verify and add support to his results. Another crosscheck will be to verify that the 54 entries rated in Likelihood of Exploit as High or Very High in CWE are covered. Not all entries have a Likelihood of Exploit entry and the basis for the ranking is not clear, but those 54 will be used as a crosscheck with the identified entries. Finally, as previously mentioned, a few additional code vulnerabilities will be added that should be straightforward to detect and address.

The first paper by Steve Christy does a very good job of analyzing the current trends. It analyzes three data sets: CVEs publicly reported in 2001 or later, CVEs associated with OS vendor advisories and open/closed source vendor advisories (derived view designed to remove overlapping CVEs from the second set). The table below summarizes the top twenty of his results:

Flaw Abbreviation	Flaw Name	Overall percentage of total flaws (2001-2006)	Percentage of total flaws (2006)	CWE Entries
XSS	cross site scripting	13.90%	21.50%	79, 80, 87, 85, 82, 81, 83, 84
buf	buffer overflow	13.30%	7.90%	119, 120
sql-inject	SQL injection	8.70%	14.00%	89
dot	directory transversal	4.70%	4.40%	22,23,36
php-include	PHP remote file inclusion	3.50%	9.50%	98
infoleak	information leak	3.30%	2.60%	205, 212, 203, 209, 207, 200, 215
dos-malform	DoS via malformed input	2.90%	2.00%	238, 234, 166, 230
link	symbolic link following	2.00%	0.50%	61, 64
format-string	Format string vulnerability	1.80%	1.00%	134
crypt	cryptologic error	1.60%	0.90%	310, 311, 347, 320, 325
priv	Bad privilege assignment	1.40%	0.90%	266, 274, 272, 250, 264, 265, 268, 270, 271, 269, 267
metachar	Unescaped shell metacharacters	1.30%	0.30%	78
perm	Assigns bad permissions	1.30%	1.10%	276
int-overflow	Integer overflow	1.00%	1.20%	190
dos-flood	DoS flood	0.80%	0.40%	400
pass	default/hard-code password	0.80%	0.40%	259
auth	weak/bad authentication	0.80%	0.70%	289, 288, 302, 305, 294, 290, 287, 303
webroot	storage of sensitive data w/insufficient access control	0.50%	0.90%	219, 433
form-field	CGI program inherently trusts form field	0.50%	0.50%	472
relpath	untrusted search path	0.40%	0.30%	426, 428, 114

Table 1Vulnerability Type Distributions in CVE overall results from 2001-2006

Table 2, below, refines the results from Table 1 by putting in the CWE entry, name of the entry and whether it is a parent or a leaf node. Blank lines separate the entries from Table 1. For instance, the first eight entries in the table correspond to the first entry, Cross Site Scripting (XSS), in Table 1. Table 2 contains 64 entries.

CWE Entry	***Description***	***Parent or Leaf node***
79	Cross Site Scripting (XSS)	parent
80	Basic XSS	child of 79 - leaf
87	Alternate XSS syntax	child of 79 - leaf
85	Doubled character XSS manipulators, e.g. '<<script'	child of 79 - leaf
82	Script in IMG tags	child of 79 - leaf
81	XSS in error pages	child of 79 - leaf
83	XSS using Script in Attributes	child of 79 - leaf
84	XSS using Script Via Encoded URI Schemes	child of 79 - leaf
119	Buffer Errors	parent
120	Unbounded Transfer ('Classic overflow')	child of 119 - parent
89	SQL Injection	leaf
22	Path Traversal	parent
23	Relative Path Traversal	child of 22 - parent
36	Absolute Path Traversal	child of 22 - parent
98	PHP File Inclusion	leaf
205	Behavioral Discrepancy Information Leak	child of 203 - parent

CWE Entry	*Description*	*Parent or Leaf node*
212	Cross-Boundary Cleansing Information Leak	child of 200 - parent
203	Discrepancy Information Leaks	child of 200 - parent
209	Error Message Information Leaks	child of 200 - parent
207	External Behavioral Inconsistency Information Leak	child of 205 - leaf
200	Information Leak (information disclosure)	parent
215	Information Leak through Debug Information	child of 200 - leaf
238	Missing Element Error	leaf
234	Missing Parameter Error	leaf
166	Missing Special Element	leaf
230	Missing Value Error	leaf
61	UNIX symbolic link (symlink) following	leaf
64	Windows Shortcut Following (.LNK)	leaf
134	Format String Vulnerability	parent
310	Cryptologic Issues	parent
311	Failure to encrypt data	child of 310 - parent
347	Improperly Verified Signature	leaf
320	Key Management Errors	child of 310 - parent
325	Missing Required Cryptographic Step	child of 310 - leaf
266	Incorrect Privilege Assignment	leaf
274	Insufficient Privileges	child of 265 - leaf

CWE Entry	*Description*	*Parent or Leaf node*
272	Least Privilege Violation	child of 271 - leaf
250	Often Misused: Privilege Management	leaf
264	Permissions, Privileges, and Access Controls	parent
265	Privilege/sandbox issues	child of 264 - parent
268	Privilege Chaining	child of 265 - leaf
270	Privilege Context Switching Error	child of 265 -leaf
271	Privilege Dropping/Lowering Errors	child of 265 - parent
269	Privilege Management Error	child of 265 - leaf
267	Unsafe Privilege	child of 265 -leaf
78	OS Command Injection	leaf
276	Insecure Default Permissions	leaf
190	Integer Overflow (wrap or wrap around)	parent
400	Resource Exhaustion (file descriptor, disk space, sockets,...)	leaf
259	Hard-coded Password	parent
289	Authentication Bypass by Alternate Name	leaf
288	Authentication Bypass by Alternate Path/Channel	leaf
302	Authentication Bypass by Assumed-Immutable Data	leaf
305	Authentication Bypass by Primary Weakness	leaf

CWE Entry	*Description*	*Parent or Leaf node*
294	Authentication Bypass by Replay	leaf
290	Authentication Bypass by Spoofing	parent
287	Authentication Issues	parent
303	Authentication Logic Error	child of 287 -leaf
219	Sensitive Data Under Web Root	leaf
433	Unparsed Raw Web Content Delivery	leaf
472	Web Parameter Tampering	parent
426	Untrusted Search Path	parent
428	Unquoted Search Path or Element	child of 426 -leaf
114	Process Control	leaf

Table 2Vulnerabilities in CWE notation

Table 3 further refines Table 2 by showing the ancestry and position in the tree of the entries in Table 2. This will give a perspective of where the entries in Table 2 reside in the classification tree of CWE. Higher levels of parentage is to the left. That is, for example, Data Handling is a parent of Input Validation, which is a parent of Path Traversal, which is a parent of Relative Path Traversal. For some entries, intermediate generations may have been skipped between the left most column and the right most. Entries in black are entries that appeared in Table 2. Entries in red have been added to provide context from the CWE classification tree.

GG-Parent	*G-Parent*	*Parent*	*Child*
19. Data Handling			
	20. Input Validation		
		22. Path Traversal	
			23. Relative Path Traversal
			36. Absolute Path Traversal
		63. Link Following	

GG-Parent	*G-Parent*	*Parent*	*Child*
			61. UNIX symbolic link (symlink) following
			64. Windows Shortcut Following (.LNK)
		114. Process Control	
		79. Cross Site Scripting (XSS)	
			80. Basic XSS
			81. XSS in error pages
			82. Script in IMG tags
			83. XSS using Script in Attributes
			84. XSS using Script Via Encoded URI Schemes
			85. Doubled character XSS manipulators, e.g. '<<script'
			87. Alternate XSS syntax
		74. Injection	
			89. SQL Injection
			98. PHP File Inclusion
			134. Format String Vulnerability
	118. Range Errors		
		119. Buffer errors	
			120. Unbounded transfer ('Classic overflow')
			190. Integer overflow (wrap or wraparound)
	137. Representation Errors		
		159. Common Special Element Manipulations	
			166. Missing Special Element
	189. Numeric Errors		

GG-Parent	*G-Parent*	*Parent*	*Child*
		78. OS Command Injection	
		190. Integer Overflow (wrap or wrap around)	
	199. Information Management Errors		
		200. Information Leak (information disclosure)	
			212. Cross-Boundary Cleansing Information Leak
			203. Discrepancy Information Leaks
			205. Behavioral Discrepancy Information Leak (child of 203)
			207. External Behavioral Inconsistency Information Leak (child of 205)
			209. Error Message Information Leaks
			215. Information Leak through Debug Information
		219. Sensitive Data Under Web Root	
227. API Abuse			
	228. Structure and Validity Problems		
		230. Missing Value Error	
		234. Missing Parameter Error	
		238. Missing Element Error	
254. Security Features			
	264. Permissions, Privileges, and Access Controls		
		265. Privilege/sandbox Issues	
			250. Often Misused: Privilege Management

GG-Parent	*G-Parent*	*Parent*	*Child*
			266. Incorrect Privilege Assignment
			267. Unsafe Privilege
			268. Privilege Chaining
			269. Privilege Management Error
			270. Privilege Context Switching Error
			271. Privilege Dropping/Lowering Errors
			272. Least Privilege Violation (child of 271)
			274. Insufficient Privileges
		275. Permission Issues	
			276. Insecure Default Permissions
	287. Authentication Issues		
		303. Authentication Logic Error	
		592. Authentication Bypass Issues	
			289. Authentication Bypass by Alternate Name
			290. Authentication Bypass by Spoofing
			294. Authentication Bypass by Replay
			302. Authentication Bypass by Assumed-Immutable Data
			305. Authentication Bypass by Primary Weakness
	310. Cryptologic Issues		
		311. Failure to encrypt data	

GG-Parent	*G-Parent*	*Parent*	*Child*
		320. Key Management Errors	
		325. Missing Required Cryptographic Step	
		259. Hard-coded Password	
	345. Insufficient Verification of Data		
			347. Improperly Verified Signature
398. Code Quality			
	399. Resource Management Errors		
		400. Resource Exhaustion (file descriptor, disk space, sockets,...)	
	417. Channel and Path Errors		
		288. Authentication Bypass by Alternate Path/Channel	
		426. Untrusted Search Path	
			428. Unquoted Search Path or Element
	429. Handler Errors		
		433. Unparsed Raw Web Content Delivery	
	471. Modification of Assumed-Immutable Data		
		472. Web Parameter Tampering	

Table 3Refinement of Table 2

Although the by Steve Christy is a good summary, we must cross check with the OWASP Top Ten Project. In Table 4, a mapping from the OWASP Top Ten is made to the elements already appearing in Table 3. If an element needs to be added to address the OWASP Top 10, it will appear in the third column.

OWASP Top 10	*CWE Mapping – Entries Already Appearing*	*CWE Mapping – Entries Needed to be Added*
A1 Unvalidated Input	20. Input Validation	
A2 Broken Access Control		285. Missing or Inconsistent Access Control
A3 Broken Authentication and Session Management		255. Credentials Management
A4 Cross Site Scripting	79. Cross Site Scripting (XSS)	
A5 Buffer Overflow	119. Buffer errors	
A6 Injection Flaws	74. Injection	
A7 Improper Error Handling	200. Information Leak (information disclosure)	
A8 Insecure Storage	311. Failure to Encrypt Data 320. Cryptologic Issues	
A9 Application Denial of Service	400. Resource Exhaustion (file descriptor, disk space, sockets,...)	
A10 Insecure Configuration Management	265. Privilege/Sandbox Issues 275. Permission Issues 276. Insecure Default Permissions 259. Hard-coded Password 200. Information Leak (information disclosure)	522. Insufficiently Protected Credentials

Table 4Cross check with OWASP Top 10

Incorporating the results of Table 4 (in red) into Table 3 yields Table 5. Table 5 is also condensed by moving G-Parents, Parents, and Children on the same line as their ancestors.

GG-Parent	*G-Parent*	*Parent*	*Child*
19. Data Handling	20. Input Validation	22. Path Traversal	23. Relative Path Traversal
			36. Absolute Path Traversal
		63. Link Following	61. UNIX symbolic link (symlink) following

GG-Parent	*G-Parent*	*Parent*	*Child*
			64. Windows Shortcut Following (.LNK)
		114. Process Control	
		79. Cross Site Scripting (XSS)	80. Basic XSS
			81. XSS in error pages
			82. Script in IMG tags
			83. XSS using Script in Attributes
			84. XSS using Script Via Encoded URI Schemes
			85. Doubled character XSS manipulators, e.g. '<<script'
			87. Alternate XSS syntax
		74. Injection	89. SQL Injection
			98. PHP File Inclusion
			134. Format String Vulnerability
	118. Range Errors	119. Buffer errors	120. Unbounded transfer ('Classic overflow')
			190. Integer overflow (wrap or wraparound)
	137. Representation Errors	159. Common Special Element Manipulations	166. Missing Special Element
	189. Numeric Errors	78. OS Command Injection	
		190. Integer Overflow (wrap or wrap around)	
	199. Information Management Errors	200. Information Leak (information disclosure)	212. Cross-Boundary Cleansing Information Leak
			203. Discrepancy Information Leaks
			205. Behavioral Discrepancy Information Leak (child of 203)
			207. External Behavioral Inconsistency Information Leak (child of 205)

GG-Parent	*G-Parent*	*Parent*	*Child*
			209. Error Message Information Leaks
			215. Information Leak through Debug Information
		219. Sensitive Data Under Web Root	
227. API Abuse	228. Structure and Validity Problems	230. Missing Value Error	
		234. Missing Parameter Error	
		238. Missing Element Error	
254. Security Features	255. Credentials Management	522. Insufficiently Protected Credentials	
	264. Permissions, Privileges, and Access Controls	265. Privilege/sandbox Issues	250. Often Misused: Privilege Management
			266. Incorrect Privilege Assignment
			267. Unsafe Privilege
			268. Privilege Chaining
			269. Privilege Management Error
			270. Privilege Context Switching Error
			271. Privilege Dropping/Lowering Errors
			272. Least Privilege Violation (child of 271)
			274. Insufficient Privileges
		275. Permission Issues	276. Insecure Default Permissions
		284. Access Control Issues	285. Missing or Inconsistent Access Control
	287. Authentication Issues	303. Authentication Logic Error	

GG-Parent	*G-Parent*	*Parent*	*Child*
		592. Authentication Bypass Issues	289. Authentication Bypass by Alternate Name
			290. Authentication Bypass by Spoofing
			294. Authentication Bypass by Replay
			302. Authentication Bypass by Assumed-Immutable Data
			305. Authentication Bypass by Primary Weakness
	310. Cryptologic Issues	311. Failure to encrypt data	347. Improperly Verified Signature
		320. Key Management Errors	
		325. Missing Required Cryptographic Step	
		259. Hard-coded Password	
	345. Insufficient Verification of Data		347. Improperly Verified Signature
398. Code Quality	399. Resource Management Errors	400. Resource Exhaustion (file descriptor, disk space, sockets,...)	
	417. Channel and Path Errors	288. Authentication Bypass by Alternate Path/Channel	
		426. Untrusted Search Path	428. Unquoted Search Path or Element
	429. Handler Errors	433. Unparsed Raw Web Content Delivery	
	471. Modification of Assumed-Immutable Data	472. Web Parameter Tampering	

Table 5Condensed Table with OWASP entries

Table 6 expands all elements in Table 5 to their end leaf node to provide the highest possible granularity. Note that Parent and G-Parent are ancestors of the leaf nodes, but generations may have been skipped to provide the best clarity of the weakness described

by each leaf node.

G-Parent	*Parent*	*Leaf*
22. Path Traversal	23. Relative Path Traversal	24. Path Issue - dot dot slash - '../filedir'
		25. Path Issue - leading dot dot slash - '/../filedir'
		26. Path Issue - leading directory dot dot slash - '/directory/../filename'
		27. Path Issue - directory doubled dot dot slash - 'directory/../../filename'
		28. Path Issue - dot dot backslash - '..\filename'
		29. Path Issue - leading dot dot backslash - '\..\filename'
		30. Path Issue - leading directory dot dot backslash - '\directory\..\filename'
		31. Path Issue - directory doubled dot dot backslash - 'directory\..\..\filename'
		32. Path Issue - triple dot - '...'
		33. Path Issue - multiple dot - '....'
		34. Path Issue - doubled dot dot slash - '....//'
		35. Path Issue - doubled triple dot slash - '.../...//'
	36. Absolute Path Traversal	37. Path Issue - slash absolute path - /absolute/pathname/here
		38. Path Issue - backslash absolute path - \absolute\pathname\here
		39. Path Issue - drive letter or Windows volume - 'C:dirname'
		40. Path Issue - Windows UNC share - '\\UNC\share\name\'
63. Link Following	60. UNIX Path Link Problems	61. UNIX symbolic link (symlink) following
		62. UNIX Hard Link
	63. Windows Path Link Problems	64. Windows Shortcut Following (.LNK)
		65. Windows Hard Link
20. Input Validation		114. Process Control

G-Parent	*Parent*	*Leaf*
79. Cross Site Scripting (XSS)		80. Basic XSS
		81. XSS in error pages
		82. Script in IMG tags
		83. XSS using Script in Attributes
		84. XSS using Script Via Encoded URI Schemes
		85. Doubled character XSS manipulators, e.g. '<<script'
		86. Invalid Characters in Identifiers
		87. Alternate XSS syntax
74. Injection	89. SQL Injection	564. SQL Injection: Hibernate
		98. PHP File Inclusion
134. Format String Vulnerability	122. Heap Overflow	121. Stack Overflow
		124. Boundary Beginning Violation (“buffer underwrite”)
		128. Wrap-around Error
		192. Integer Coercion Error
		197. Numeric Truncation Error
		231. Extra Value Error
		476. Null Dereference
		365. Race Condition in Switch
		368. Context Switching Race Condition
		415. Double Free
		416. Use after Free
		479. Unsafe Function Call from a Signal Handler
119. Buffer errors	120. Unbounded transfer ('Classic overflow')	122. Heap Overflow
	190. Integer Overflow (wrap or wraparound)	128. Wrap-around Error

G-Parent	*Parent*	*Leaf*
		76. Equivalent Special Element Injection
		78. OS Command Injection
		90. LDAP Injection
		91. XML Injection (aka Blind Xpath injection)
		92. Custom Special Character Injection
		144. Line Delimiter
		145. Section Delimiter
		95. Direct Dynamic Code Evaluation ('Eval Injection')
		97. Server-Side Includes (SSI) Injection
		98. PHP File Inclusion
		99. Resource Injunction
		365. Race Condition in Switch
		368. Context Switching Race Condition
		415. Double Free
		479. Unsafe Function Call from a Signal Handler
		129. Unchecked Array Indexing
		476. Null Dereference
		231. Extra Value Error
		128. Wrap-around Error
		192. Integer Coercion Error
		193. Off-by-one Error
		194. Sign Extension Error
		78. OS Command Injection
159. Common Special Element Manipulations		161. Multiple Leading Special Elements
		163. Multiple Trailing Special Elements
		165. Multiple Internal Special Element
		166. Missing Special Element

G-Parent	*Parent*	*Leaf*
		167. Extra Special Element
		168. Inconsistent Special Elements
200. Information Leak (information disclosure)	212. Cross-Boundary Cleansing Information Leak	226. Sensitive Information Uncleared before Use
	203. Discrepancy Information Leaks	204. Response Discrepancy Information Leak
		208. Timing Discrepancy Information Leak
	205. Behavioral Discrepancy Information Leak (child of 203)	206. Internal Behavioral Inconsistency Information Leak
		207. External Behavioral Inconsistency Information Leak
	209. Error Message Information Leaks	81. XSS in Error Pages
		535. Information Leak Through Shell Error Message
		536. Information Leak Through Servlet Runtime Error Message
		537. Information Leak Through Java Runtime Error Message
		550. Information Leak Through Server Error Message
		600. Missing Catch Block
		215. Information Leak through Debug Information
		219. Sensitive Data Under Web Root
		230. Missing Value Error
		234. Missing Parameter Error
		238. Missing Element Error
	522. Insufficiently Protected Credentials	256. Plaintext Storage
		257. Storing Passwords in a Recoverable Format

G-Parent	*Parent*	*Leaf*
	265. Privilege/sandbox Issues	250. Often Misused: Privilege Management
		266. Incorrect Privilege Assignment
		267. Unsafe Privilege
		268. Privilege Chaining
		269. Privilege Management Error
		270. Privilege Context Switching Error
	271. Privilege Dropping/Lowering Errors	272. Least Privilege Violation
		273. Failure to Check Whether Privileges were Dropped Successfully
		274. Insufficient Privileges
	275. Permission Issues	276. Insecure Default Permissions
	284. Access Control Issues	285. Missing or Inconsistent Access Control
		303. Authentication Logic Error
	592. Authentication Bypass Issues	289. Authentication Bypass by Alternate Name
		290. Authentication Bypass by Spoofing
		294. Authentication Bypass by Replay
		302. Authentication Bypass by Assumed-Immutable Data
		305. Authentication Bypass by Primary Weakness
311. Failure to encrypt data	Using a Broken or Risky Cryptographic Algorithm	301. Reflection Attack in an Authentication Protocol
320. Key Management Errors		257. Storing Passwords in a Recoverable Format
		325. Missing Required Cryptographic Step
		259. Hard-coded Password

G-Parent	*Parent*	*Leaf*
	345. Insufficient Verification of Data	347. Improperly Verified Signature
		400. Resource Exhaustion (file descriptor, disk space, sockets,...)
		288. Authentication Bypass by Alternate Path/Channel
	426. Untrusted Search Path	428. Unquoted Search Path or Element
		433. Unparsed Raw Web Content Delivery
398. Code Quality		
	471. Modification of Assumed-Immutable Data	192. Integer Coercion Error
		197. Numeric Truncation Error
		128. Wrap-around Error
		473. PHP External Variable Modification

Table 6Conversion to Leaf Nodes

A consideration that must be made is that the scope of SC22 OWGV is that of what can specifically be done at the coding level to avoid vulnerabilities. There are 122 entries in Table 6, but several of the entries are specifically design issues. The next step will be to differentiate those items which can be affected through coding standards and those that are design standards which are outside of the scope of SC22 OWGV. This is something that the SC22 OWGV group should debate and determine as there is a blur between design and coding and a consensus opinion of the group would be the most valuable. In many cases, the leaf node weaknesses in Table 6 will need to be addressed both as a coding and as a design issue. For the purposes of SC22 OWGV, anything that can be influenced or affected at the coding level should remain in scope and addressed by SC22 OWGV.

One other consideration that SC22 OWGV may want to make are those weaknesses in CWE that can be easily prevented. Code quality issues such as Dead Code (CWE 561), Memory Leak (CWE 401), Unused Variable (CWE 563), and Improper String Length Checking (CWE 135) are indicative of code that isn't well written or rigorously tested. These are usually not serious vulnerabilities (if they even are the basis for vulnerabilities), and may only be used to do an irritating DoS or as aides to crafting some other attack. These are relatively easy to find and fix as there are many tools available to address these issues.

More debatable weaknesses are code quality issues such as User Interface Inconsistency

(CWE 446), Suspicious Comment (CWE 546) (e.g. “#workaround” or “#need to fix” or “#hack”), or Memory Locking (CWE 591). These can aid or even be the basis of vulnerabilities. However these seem to be outside of the scope of SC22 OWGV.

Including the weaknesses described in the previous two paragraphs in Table 6 yields Table 7 below.

G-Parent	***Parent***	***Leaf***
22. Path Traversal	23. Relative Path Traversal	24. Path Issue - dot dot slash - '../filedir'
		25. Path Issue - leading dot dot slash - '/../filedir'
		26. Path Issue - leading directory dot dot slash - '/directory/../filename'
		27. Path Issue - directory doubled dot dot slash - 'directory/../../filename'
		28. Path Issue - dot dot backslash - '..\filename'
		29. Path Issue - leading dot dot backslash - '\..\filename'
		30. Path Issue - leading directory dot dot backslash - '\directory\..\filename'
		31. Path Issue - directory doubled dot dot backslash - 'directory\..\..\filename'
		32. Path Issue - triple dot - '...'
		33. Path Issue - multiple dot - '....'
		34. Path Issue - doubled dot dot slash - '....//'
		35. Path Issue - doubled triple dot slash - '.../...//'
	36. Absolute Path Traversal	37. Path Issue - slash absolute path - /absolute/pathname/here
		38. Path Issue - backslash absolute path - \absolute\pathname\here

G-Parent	*Parent*	*Leaf*
		39. Path Issue - drive letter or Windows volume - 'C:dirname'
		40. Path Issue - Windows UNC share - '\\UNC\share\name\'
63. Link Following	60. UNIX Path Link Problems	61. UNIX symbolic link (symlink) following
		62. UNIX Hard Link
	63. Windows Path Link Problems	64. Windows Shortcut Following (.LNK)
		65. Windows Hard Link
20. Input Validation		114. Process Control
79. Cross Site Scripting (XSS)		80. Basic XSS
		81. XSS in error pages
		82. Script in IMG tags
		83. XSS using Script in Attributes
		84. XSS using Script Via Encoded URI Schemes
		85. Doubled character XSS manipulators, e.g. '<<script'
		86. Invalid Characters in Identifiers
		87. Alternate XSS syntax
74. Injection	89. SQL Injection	564. SQL Injection: Hibernate
		98. PHP File Inclusion
134. Format String Vulnerability	122. Heap Overflow	121. Stack Overflow
		124. Boundary Beginning Violation (“buffer underwrite”)
		128. Wrap-around Error
		192. Integer Coercion Error
		197. Numeric Truncation Error
		231. Extra Value Error
		476. Null Dereference

G-Parent	*Parent*	*Leaf*
		365. Race Condition in Switch
		368. Context Switching Race Condition
		415. Double Free
		416. Use after Free
		479. Unsafe Function Call from a Signal Handler
119. Buffer errors	120. Unbounded transfer ('Classic overflow')	122. Heap Overflow
	190. Integer Overflow (wrap or wraparound)	128. Wrap-around Error
		76. Equivalent Special Element Injection
		78. OS Command Injection
		90. LDAP Injection
		91. XML Injection (aka Blind Xpath injection)
		92. Custom Special Character Injection
		144. Line Delimiter
		145. Section Delimiter
		95. Direct Dynamic Code Evaluation ('Eval Injection')
		97. Server-Side Includes (SSI) Injection
		98. PHP File Inclusion
		99. Resource Injunction
		365. Race Condition in Switch
		368. Context Switching Race Condition
		415. Double Free
		479. Unsafe Function Call from a Signal Handler

G-Parent	*Parent*	*Leaf*
		129. Unchecked Array Indexing
		476. Null Dereference
		231. Extra Value Error
		128. Wrap-around Error
		192. Integer Coercion Error
		193. Off-by-one Error
		194. Sign Extension Error
		78. OS Command Injection
159. Common Special Element Manipulations		161. Multiple Leading Special Elements
		163. Multiple Trailing Special Elements
		165. Multiple Internal Special Element
		166. Missing Special Element
		167. Extra Special Element
		168. Inconsistent Special Elements
200. Information Leak (information disclosure)	212. Cross-Boundary Cleansing Information Leak	226. Sensitive Information Uncleared before Use
	203. Discrepancy Information Leaks	204. Response Discrepancy Information Leak
		208. Timing Discrepancy Information Leak
	205. Behavioral Discrepancy Information Leak (child of 203)	206. Internal Behavioral Inconsistency Information Leak
		207. External Behavioral Inconsistency Information Leak
	209. Error Message Information Leaks	81. XSS in Error Pages
		535. Information Leak Through Shell Error Message

G-Parent	*Parent*	*Leaf*
		536. Information Leak Through Servlet Runtime Error Message
		537. Information Leak Through Java Runtime Error Message
		550. Information Leak Through Server Error Message
		600. Missing Catch Block
		215. Information Leak through Debug Information
		219. Sensitive Data Under Web Root
		230. Missing Value Error
		234. Missing Parameter Error
		238. Missing Element Error
	522. Insufficiently Protected Credentials	256. Plaintext Storage
		257. Storing Passwords in a Recoverable Format
	265. Privilege/sandbox Issues	250. Often Misused: Privilege Management
		266. Incorrect Privilege Assignment
		267. Unsafe Privilege
		268. Privilege Chaining
		269. Privilege Management Error
		270. Privilege Context Switching Error
	271. Privilege Dropping/Lowering Errors	272. Least Privilege Violation
		273. Failure to Check Whether Privileges were Dropped Successfully
		274. Insufficient Privileges
	275. Permission Issues	276. Insecure Default Permissions

G-Parent	*Parent*	*Leaf*
	284. Access Control Issues	285. Missing or Inconsistent Access Control
		303. Authentication Logic Error
	592. Authentication Bypass Issues	289. Authentication Bypass by Alternate Name
		290. Authentication Bypass by Spoofing
		294. Authentication Bypass by Replay
		302. Authentication Bypass by Assumed-Immutable Data
		305. Authentication Bypass by Primary Weakness
311. Failure to encrypt data	Using a Broken or Risky Cryptographic Algorithm	301. Reflection Attack in an Authentication Protocol
320. Key Management Errors		257. Storing Passwords in a Recoverable Format
		325. Missing Required Cryptographic Step
		259. Hard-coded Password
	345. Insufficient Verification of Data	347. Improperly Verified Signature
		400. Resource Exhaustion (file descriptor, disk space, sockets,...)
		288. Authentication Bypass by Alternate Path/Channel
	426. Untrusted Search Path	428. Unquoted Search Path or Element
		433. Unparsed Raw Web Content Delivery
398. Code Quality		
	399. Resource Management Errors	401. Memory Leak
		591. Memory Locking
		446. User Interface Inconsistency

G-Parent	*Parent*	*Leaf*
	471. Modification of Assumed-Immutable Data	192. Integer Coercion Error
		197. Numeric Truncation Error
		128. Wrap-around Error
		473. PHP External Variable Modification
	561. Dead Code	570. Expression is Always False
		571. Expression is Always True
		563. Unused Variable

Table 7Inclusion of Some Code Quality Issues

Table 7 contains 129 items. Once discussion and debate is complete, it is expected that some items will be removed as entirely design issues and a few others added. The final result will be approximately 75-100 vulnerabilities that can to some degree be addressed through language selection and use.

Part 3. Mapping from CWE to CERT Secure Coding Standards

The approach up to Table 7 was a mapping from the current most frequently exploited vulnerabilities to CWE entries. To make these items actionable to a software developer, recommendations must be made in terms that the developers can use. CERT has done a good job in their Secure Coding effort (www.cert.org/secure-coding). Table 8 links the identified CWE leaf nodes to the CERT Secure Coding Entries. The recommendations and rules for the C language are used. Note that though this is done only for the C language, the same recommendations and rule may be applicable to C++, Java and other languages.

Some of the entries in Table 7 are closely related both in their cause and possible mitigations. Therefore the elements have been grouped to reflect that. Groupings are separated by a row containing "**********"

For each grouping, a template based on Document N0072 (http://www.aitcnet.org/isai/_NextMeeting/_Last%20Meeting/22-OWGV-N-0072/n0072.html) was created. Data from the CWE entries and the CERT Secure Coding web site was used to fill in the templates. The templates will need additional work to complete them. Three entries at the end (473. PHP External Variable Modification, 238. Missing Element Error, 600. Missing Catch Block) seemed to be out of scope or no longer valid and so no template was needed nor created for them.

CWE Entry	*CERT Secure Coding Entry for C*
24. Path Issue - dot dot slash - '../filedir'	FIO02-A. Canonicalize filenames originating from untrusted sources FIO05-A. Identify files using multiple file attributes
25. Path Issue - leading dot dot slash - '/../filedir'	FIO02-A. Canonicalize filenames originating from untrusted sources FIO05-A. Identify files using multiple file attributes TMP00-A. Do not create temporary files in shared directories
26. Path Issue - leading directory dot dot slash - '/directory/../filename'	FIO02-A. Canonicalize filenames originating from untrusted sources FIO05-A. Identify files using multiple file attributes TMP00-A. Do not create temporary files in shared directories
27. Path Issue - directory doubled dot dot slash - 'directory/../../filename'	FIO02-A. Canonicalize filenames originating from untrusted sources FIO05-A. Identify files using multiple file attributes TMP00-A. Do not create temporary files in shared directories
28. Path Issue - dot dot backslash - '..\filename'	FIO02-A. Canonicalize filenames originating from untrusted sources FIO05-A. Identify files using multiple file attributes TMP00-A. Do not create temporary files in shared directories
29. Path Issue - leading dot dot backslash - '\..\filename'	FIO02-A. Canonicalize filenames originating from untrusted sources FIO05-A. Identify files using multiple file attributes TMP00-A. Do not create temporary files in shared directories
30. Path Issue - leading directory dot dot backslash - '\directory\..\filename'	FIO02-A. Canonicalize filenames originating from untrusted sources FIO05-A. Identify files using multiple file attributes TMP00-A. Do not create temporary files in shared directories
31. Path Issue - directory doubled dot dot backslash - 'directory\..\..\filename'	FIO02-A. Canonicalize filenames originating from untrusted sources FIO05-A. Identify files using multiple file attributes TMP00-A. Do not create temporary files in shared directories

CWE Entry	*CERT Secure Coding Entry for C*
32. Path Issue - triple dot - '...'	FIO02-A. Canonicalize filenames originating from untrusted sources FIO05-A. Identify files using multiple file attributes TMP00-A. Do not create temporary files in shared directories
33. Path Issue - multiple dot - '....'	FIO02-A. Canonicalize filenames originating from untrusted sources FIO05-A. Identify files using multiple file attributes TMP00-A. Do not create temporary files in shared directories
34. Path Issue - doubled dot dot slash - '....//'	FIO02-A. Canonicalize filenames originating from untrusted sources FIO05-A. Identify files using multiple file attributes TMP00-A. Do not create temporary files in shared directories
35. Path Issue - doubled triple dot slash - '.../...//'	FIO02-A. Canonicalize filenames originating from untrusted sources FIO05-A. Identify files using multiple file attributes TMP00-A. Do not create temporary files in shared directories
**********	
37. Path Issue - slash absolute path - /absolute/pathname/here	FIO05-A. Identify files using multiple file attributes
38. Path Issue - backslash absolute path - \absolute\pathname\here	FIO05-A. Identify files using multiple file attributes
39. Path Issue - drive letter or Windows volume - 'C:dirname'	FIO05-A. Identify files using multiple file attributes
40. Path Issue - Windows UNC share - '\\UNC\share\name\'	FIO05-A. Identify files using multiple file attributes
**********	
61. UNIX symbolic link (symlink) following	FIO05-A. Identify files using multiple file attributes
62. UNIX Hard Link	FIO05-A. Identify files using multiple file attributes TMP00-A. Do not create temporary files in shared directories
**********	
64. Windows Shortcut Following (.LNK)	FIO05-A. Identify files using multiple file attributes TMP00-A. Do not create temporary files in shared directories
65. Windows Hard Link	FIO05-A. Identify files using multiple file attributes TMP00-A. Do not create temporary files in shared directories

CWE Entry	*CERT Secure Coding Entry for C*
**********	
192. Integer Coercion Error	INT02-A. Understand integer conversion rules INT03-A. Use a secure integer library INT11-A. Be careful converting small signed integers to larger unsigned integers INT13-A. Do not assume that a right shift operation is implemented as a logical or an arithmetic shift INT15-A. Take care when converting from pointer to integer or integer to pointer INT31-C. Ensure that integer conversions do not result in lost or misinterpreted data INT32-C. Ensure that integer operations do not result in an overflow INT35-C. Upcast integers before comparing or assigning to a larger integer size INT36-C. Do not shift a negative number of bits or more bits than exist in the operand INT37-C. Arguments to character handling functions must be representable as an unsigned char
**********	
197. Numeric Truncation Error	INT02-A. Understand integer conversion rules INT31-C. Ensure that integer conversions do not result in lost or misinterpreted data
**********	
230. Missing Value Error	FIO04-A. Detect and handle input output errors
231. Extra Value Error	Need recommendations/rules
**********	
476. Null Pointer Dereference	EXP34-C. Ensure a pointer is valid before dereferencing it (can be/sort of) DCL30-C. Do not refer to an object outside of its lifetime MEM00-A. Allocate and free memory in the same module, at the same level of abstraction
**********	
365. Race Condition in Switch	Need recommendations/rules
**********	
368. Context Switching Race Condition	Need recommendations/rules

CWE Entry	*CERT Secure Coding Entry for C*
**********	
415. Double Free	MEM01-A. Set pointers to dynamically allocated memory to NULL after they are released DCL30-C. Do not refer to an object outside of its lifetime MEM00-A. Allocate and free memory in the same module, at the same level of abstraction EXP34-C. Ensure a pointer is valid before dereferencing it
416. Use after Free	MEM01-A. Set pointers to dynamically allocated memory to NULL after they are released DCL30-C. Do not refer to an object outside of its lifetime MEM00-A. Allocate and free memory in the same module, at the same level of abstraction
**********	
401. Memory Leak	MEM00-A. Allocate and free memory in the same module, at the same level of abstraction Need additional recommendations/rules
**********	
256. Plaintext Storage	FIO06-A. Create files with appropriate access permissions
257. Storing Passwords in a Recoverable Format	Need recommendations/rules
**********	
250. Often Misused: Privilege Management	Need recommendations/rules
**********	
266. Incorrect Privilege Assignment	Need recommendations/rules
267. Unsafe Privilege	Need recommendations/rules
268. Privilege Chaining	Need recommendations/rules
269. Privilege Management Error	Need recommendations/rules
270. Privilege Context Switching Error	Need recommendations/rules
272. Least Privilege Violation	Need recommendations/rules
273. Failure to Check Whether Privileges were Dropped Successfully	Need recommendations/rules
274. Insufficient Privileges	Need recommendations/rules

CWE Entry	*CERT Secure Coding Entry for C*
276. Insecure Default Permissions	Need recommendations/rules
**********	
259. Hard-coded Password	Need recommendations/rules
**********	
591. Memory Locking	Need recommendations/rules
**********	
570. Expression is Always False	MSC00-A. Compile cleanly at high warning levels MSC07-A. Detect and remove dead code Need additional recommendations/rules
571. Expression is Always True	MSC00-A. Compile cleanly at high warning levels MSC07-A. Detect and remove dead code Need additional recommendations/rules
**********	
563. Unused Variable **********	MSC00-A. Compile cleanly at high warning levels

Table 8Mapping some entries from CWE (Table 7) to CERT Secure Coding Entry

CWE Entry	*CERT Secure Coding Entry for C*
114. Process Control	Need recommendations/rules
**********	
80. Basic XSS	Need recommendations/rules
81. XSS in error pages	Need recommendations/rules
82. Script in IMG tags	Need recommendations/rules
83. XSS using Script in Attributes	Need recommendations/rules
84. XSS using Script Via Encoded URI Schemes	Need recommendations/rules
85. Doubled character XSS manipulators, e.g. '<<script'	Need recommendations/rules

CWE Entry	*CERT Secure Coding Entry for C*
86. Invalid Characters in Identifiers	Need recommendations/rules
87. Alternate XSS syntax	Need recommendations/rules
**********	
564. SQL Injection: Hibernate	Need recommendations/rules
**********	
98. PHP File Inclusion	Need recommendations/rules
**********	
121. Stack Overflow	ARR30-C. Guarantee that array indices are within the valid range STR31-C. Guarantee that storage for strings has sufficient space for character data and the null terminator STR32-C. Guarantee that all byte strings are null-terminated STR33-C. Size wide character strings correctly STR34-C. Do not copy data from an unbounded source to a fixed-length array STR00-A. Use TR 24731 for remediation of existing string manipulation code STR01-A. Use managed strings for development of new string manipulation code STR02-A. Sanitize data passed to complex subsystems STR03-A. Do not inadvertently truncate a null terminated byte string INT32-C. Ensure that integer operations do not result in an overflow INT04-A. Enforce limits on integer values originating from untrusted sources.
**********	
124. Boundary Beginning Violation (“buffer underwrite”)	ARR30-C. Guarantee that array indices are within the valid range
**********	
128. Wrap-around Error	INT08-A. Verify that all integer values are in range
**********	
129. Unchecked Array Indexing	ARR30-C. Guarantee that array indices are within the valid range
**********	

CWE Entry	*CERT Secure Coding Entry for C*
479. Unsafe Function Call from a Signal Handler	Need recommendations/rules
**********	
122. Heap Overflow	ARR30-C. Guarantee that array indices are within the valid range STR31-C. Guarantee that storage for strings has sufficient space for character data and the null terminator STR32-C. Guarantee that all byte strings are null-terminated STR33-C. Size wide character strings correctly STR34-C. Do not copy data from an unbounded source to a fixed-length array STR00-A. Use TR 24731 for remediation of existing string manipulation code STR01-A. Use managed strings for development of new string manipulation code STR02-A. Sanitize data passed to complex subsystems STR03-A. Do not inadvertently truncate a null terminated byte string INT32-C. Ensure that integer operations do not result in an overflow
**********	
76. Equivalent Special Element Injection	Need recommendations/rules
**********	
78. OS Command Injection	Need recommendations/rules
**********	
90. LDAP Injection	Need recommendations/rules
91. XML Injection (aka Blind Xpath injection)	Need recommendations/rules
92. Custom Special Character Injection	Need recommendations/rules
95. Direct Dynamic Code Evaluation ('Eval Injection')	Need recommendations/rules
98. PHP File Inclusion	Need recommendations/rules
99. Resource Injection	Need recommendations/rules
**********	

CWE Entry	*CERT Secure Coding Entry for C*
144. Line Delimiter	Need recommendations/rules
145. Section Delimiter	Need recommendations/rules
**********	
97. Server-Side Includes (SSI) Injection	Need recommendations/rules
**********	
193. Off-by-one Error	Need recommendations/rules
**********	
194. Sign Extension Error	INT13-A. Do not assume that a right shift operation is implemented as a logical or an arithmetic shift
**********	
161. Multiple Leading Special Elements	Need recommendations/rules
163. Multiple Trailing Special Elements	Need recommendations/rules
165. Multiple Internal Special Elements	Need recommendations/rules
166. Missing Special Element	Need recommendations/rules
167. Extra Special Element	Need recommendations/rules
168. Inconsistent Special Elements	Need recommendations/rules
**********	
226. Sensitive Information Uncleared before Use	MEM03-A. Clear sensitive information stored in dynamic memory prior to deallocation
**********	
204. Response Discrepancy Information Leak	Need recommendations/rules
206. Internal Behavioral Inconsistency Information Leak	Need recommendations/rules
207. External Behavioral Inconsistency Information Leak	Need recommendations/rules
208. Timing Discrepancy Information Leak	Need recommendations/rules
**********	
234. Missing Parameter Error	DCL31-C. Ensure every function has a function prototype
**********	

CWE Entry	*CERT Secure Coding Entry for C*
285. Missing or Inconsistent Access Control	Need recommendations/rules
**********	
288. Authentication Bypass by Alternate Path/Channel	Need recommendations/rules
289. Authentication Bypass by Alternate Name	Need recommendations/rules
290. Authentication Bypass by Spoofing	Need recommendations/rules
294. Authentication Bypass by Replay	Need recommendations/rules
301. Reflection Attack in an Authentication Protocol	Need recommendations/rules
302. Authentication Bypass by Assumed-Immutable Data	Need recommendations/rules
303. Authentication Logic Error	Need recommendations/rules
305. Authentication Bypass by Primary Weakness	Need recommendations/rules
**********	
400. Resource Exhaustion (file descriptor, disk space, sockets,...)	Need recommendations/rules
**********	
428. Unquoted Search Path or Element	Need recommendations/rules
**********	
347. Improperly Verified Signature	Need recommendations/rules
**********	
325. Missing Required Cryptographic Step	Need recommendations/rules
**********	
215. Information Leak through Debug Information	Need recommendations/rules
**********	
535. Information Leak Through Shell Error Message	Need recommendations/rules
536. Information Leak Through Servlet Runtime Error Message	Need recommendations/rules

CWE Entry	*CERT Secure Coding Entry for C*
537. Information Leak Through Java Runtime Error Message	Need recommendations/rules
550. Information Leak Through Server Error Message	Need recommendations/rules
**********	
219. Sensitive Data Under Web Root	May be out of scope/Design issue
433. Unparsed Raw Web Content Delivery	May be out of scope/Design issue
**********	
446. User Interface Inconsistency	May be out of scope/Design issue
********** 473. PHP External Variable Modification 238. Missing Element Error 600. Missing Catch Block	********** Tech specific instance of MAID No description in CWE – too vague to decide Out of scope

There are potentially some CERT Secure Coding Entries that are very important to do and relatively painless to implement, yet do not appear in the above table. Only one has been identified thus far and it appears in the table below. Additional analysis of the CERT site will need to be made to identify additional entries. The one entry identified is already covered by an entry that already appears in the right hand column of the table above. However, it is important to acknowledge their importance and the fact that each is superceded by an entry above.

CERT Secure Coding Entry for C	*Superceded by*
EXP33-C. Do not reference uninitialized variables	MSC00-A. Compile cleanly at high warning levels