Template for comments and secretariat observations

Date: 2012-09-14	Document: **WG 23 N0421**

1	2	(3)	4	5	(6)	(7)
MB[1]	**Clause No./ Subclause No./ Annex** (e.g. 3.1)	**Paragraph/ Figure/Table/ Note** (e.g. Table 1)	**Type of comment**[2]	**Comment (justification for change) by the MB**	**Proposed change by the MB**	**Secretariat observations** on each comment submitted
CA-1			Ge	While we appreciate the work that SC 22/WG 23 has gone to develop language-specific Annexes for the document, we believe that the document is incomplete without annexes for the major languages, such as C++, COBOL, Fortran, Java and PHP	Add Annexes for the suggested languages. Retard the publication schedule of the TR, or publish a 3rd edition as soon as these annexes become available. Explicitly, add the annex for PHP which has been submitted and reviewed by WG 23.	Accept. PHP is already added. Third edition is planned once we get appropriate progress made. The Committee cannot guarantee which annexes may be added.
CA-2	6.39		Te	Section 6.39 (REU) is inconsistent, and the annexes C.39, D.39, etc are inconsistent with 6.39. The writeup now focuses almost exclusively on how designers and coders use termination and design termination strategy, not how a language and runtime influence termination strategy.	Abnormal termination is a significant vulnerability, but some significant redesign will be needed for this section in the programming-language aspects. We recommend for now that this vulnerability be placed in section 7 and the equivalent language-specific annexe portions be removed, with a plan to rework the language/ runtime environment for the next revision.	Reject the comment. Agree that identified issues in C.39 and G.39 are valid and will be carried forward into next revision.
CA-3	D.3.2		Te	The section has ignored the statements in the main section (6.3.5) that provide excellent guidance on mitigating problems, including the use of analysis tools.	Place the following text before the bullets of D.3.2 as a bullet: - follow the advice provided by 6.3.5	Accept
CA-4	D.15.2		Te	The section has ignored the statements in the main section (6.15.5) that provide excellent guidance on mitigating problems, including the use of analysis tools.	Place the following text before the bullets of D.3.2 as a bullet: - follow the advice provided by 6.15.5	Accept - in D.15.2
CA-5	D.12.2		Te	The section has ignored the statements in the main section (6.12.5) that provide excellent guidance on mitigating problems, especially the use of analysis tools.	Place the following text before the bullets of D.3.2 as a bullet: – follow the advice provided by 6.12.5 bullet 2 and bullet 3. – or replicate the bullet from D.8.2	Place the following text before the bullets of D.12.2 as a bullet: - follow the advice provided by 6.12.5
CA-	D.11.2		Te	This section ignores the bounds-checking library that was	Add the following bullet to D.11.2:	Accept

1 **MB** = Member body (enter the ISO 3166 two-letter country code, e.g. CN for China; comments from the ISO/CS editing unit are identified by **)

2 **Type of comment:** **ge** = general **te** = technical **ed** = editorial

NOTE Columns 1, 2, 4, 5 are compulsory.

Template for comments and secretariat observations

Date: 2012-09-14	Document: **WG 23 N0421**

1	2	(3)	4	5	(6)	(7)
MB[1]	**Clause No./ Subclause No./ Annex** (e.g. 3.1)	**Paragraph/ Figure/Table/ Note** (e.g. Table 1)	**Type of comment**[2]	**Comment (justification for change) by the MB**	**Proposed change by the MB**	**Secretariat observations** on each comment submitted
6				created for C to address memory copy errors	Use the library functions memcpy_s, memmove_s, memset_s, strrncpy_s, ... to ensure that buffer bounds are not exceeded during buffer write operations. Alternatively, replicate guidance from D.8.2 but add memcpy and memmove functions as well.	
CA-7	D.9.2		Te	This section misses the bounds-checking library for strncpy and strnmove	Add strncpy_s and strnmove_s to the list of recommended functions. In fact, make these 2 preferential to strncpy and strnmove. Alternatively, replicate guidance from D.8.2	Accept, addition of strncpy_s and strnmove_s. Ignore the "alternatively"
CA-8	D.6.2		Te	The write-up is missing consideration of the use of static analysis tools	Add a bullet that says: Follow the recommendation of 6.6.5 to use static analysis tools.	Accept as "Follow the guidance of 6.6.5" - no explicit mention of SCA tools.
CA-9	D.6.2		Te	The write-up is missing consideration of other enum issues relating to iteration over enums that repeat or that have gaps.	Add a bullet that says: -Avoid using loops that iterate over an enum that has representation specified for the enums, unless it can be guaranteed that there are no gaps or repetition of representation values within the enum definition.	Accept.
CA-10	6.18		Te	None of the languages that currently have an annex admit to having a sign extension error problem. This means that either the vulnerability does not exist as written, or thee Annex authors do not understand the problem.	Remove the vulnerability [XZI] or modify it to be meaningful. If this vulnerability is removed, address the issue by adding to 6.7.3 [FLC], new paragraph between 1 and 2: as follows:	Accept in principle. There was no consensus to remove [XZI] at this point in time – UK objects on the basis that some languages do have this issue.

1 **MB** = Member body (enter the ISO 3166 two-letter country code, e.g. CN for China; comments from the ISO/CS editing unit are identified by **)

2 **Type of comment:** **ge** = general **te** = technical **ed** = editorial

NOTE Columns 1, 2, 4, 5 are compulsory.

1	2	(3)	4	5	(6)	(7)
MB[1]	**Clause No./ Subclause No./ Annex** (e.g. 3.1)	**Paragraph/ Figure/Table/ Note** (e.g. Table 1)	**Type of comment**[2]	**Comment (justification for change) by the MB**	**Proposed change by the MB**	**Secretariat observations** on each comment submitted
					When the conversion results in no change in representation but a change in value for the new type, this may result in a value that is not expressible in the new type, or that has a dramatically different order or meaning. One such situation is the change of sign between the origin and destination (negative -> positive or positive -> negative), which changes the relative order of members of the two types and could result in memory access failures if the values are used in address calculations. Make corresponding corrections to the annexes.	The addition of the proposed paragraph for 6.7.3 [FLC] is accepted. Note: there is consensus that the annexes need to be changed to reflect this addition. The Committee will examine this issue in the next revision cycle.
JP-1	Entire	Entire	Te	It was proposed to raise the coverage of CWE on the N4704, i.e. PDTR 24772 at the WG23 meeting. Although the comment was accepted in principle, concrete resolutions were not present, no modification was made. JP comments propose to raise the coverage of CWE top 25, and analysed how they are addressed. There are so many CWE's that it may not be easy to address to all of them. However, if we concentrate on the most quoted CWE's i.e. CWE top 25, then we can make meaningful improvements with the limited efforts. The attachment 1 is the result of the analysis whether 24772.2 address to the CWE top 25 or not. Out of 25, 14 are covered and 11 are not covered. Out of 11 uncovered, four corresponding descriptions were found in the TR24772.2. So this means TR24772.2 has coverage of 18 CWE, and there are only 7 CWE which are not covered. So, descriptions are drafted for CWE's that are not covered by the TR 24772.2. If we can add four references, and 7 new descriptions to the TR 24772.2, then we can improve that TR 24772.2 covers all of CWE top 25.	The steps are described one by one in the following rows.	The Committee agrees with the sentiment of this statement. No action to take.
JP-	6.10.2	Cross	Te	To raise the CWE top 25 coverage, add the reference.	Add the following line after the CWE: 129, to denote the reference to the CWE 676.	Accept

1 **MB** = Member body (enter the ISO 3166 two-letter country code, e.g. CN for China; comments from the ISO/CS editing unit are identified by **)

2 **Type of comment: ge** = general **te** = technical **ed** = editorial

NOTE Columns 1, 2, 4, 5 are compulsory.

1	2	(3)	4	5	(6)	(7)
MB[1]	**Clause No./ Subclause No./ Annex** (e.g. 3.1)	**Paragraph/ Figure/Table/ Note** (e.g. Table 1)	**Type of comment**[2]	**Comment (justification for change) by the MB**	**Proposed change by the MB**	**Secretariat observations** on each comment submitted
2		reference			676. Use of Potentially Dangerous Function	
JP-3	7.21.2	Cross reference	Te	To raise the CWE top 25 coverage, add the reference.	Add the following line after the CWE: 807, to denote the reference to the CWE 862. 862. Missing Authorization	Accept
JP-4	8.8.2	Cross reference	Te	To raise the CWE top 25 coverage, add the reference.	Add the following line after the CWE: 642, to denote the reference to the CWE 311. 311. Missing Encryption of Sensitive Data	Accept
JP-5	7.24	New	Te	Create the 7.24 as shown on the right column.	7.Z Download of Code Without Integrity Check [???] 7.Z.1 Description of application vulnerability The product downloads source code or an executable from a remote location and executes the code without sufficiently verifying the origin and integrity of the code. 7.Z.2 Cross reference CWE: 494. Download of Code Without Integrity Check 7.Z.3 Mechanism of failure An attacker can execute malicious code by compromising the host server, performing DNS spoofing, or modifying the code in transit. 7.Z.4 Avoiding the vulnerability or mitigating its effects Perform proper forward and reverse DNS lookups to detect DNS spoofing. Encrypt the code with a reliable encryption scheme before transmitting. This is only a partial solution since it will not prevent your code from being modified on the hosting site or in transit. Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness easier to avoid. Specifically, it may be helpful to use tools or frameworks to perform integrity checking on the transmitted code. If you are providing the code that is to be downloaded,	Agreed in principle. The consensus of the committee is that these vulnerabilities are not mature enough to add to the current working draft of the TR. A drafting committee will be formed to work on these vulnerabilities, making sure that they are included in the 3rd edition of 24772.

[1] **MB** = Member body (enter the ISO 3166 two-letter country code, e.g. CN for China; comments from the ISO/CS editing unit are identified by **)

[2] **Type of comment:** **ge** = general **te** = technical **ed** = editorial

NOTE Columns 1, 2, 4, 5 are compulsory.

1	2	(3)	4	5	(6)	(7)
MB[1]	**Clause No./ Subclause No./ Annex** (e.g. 3.1)	**Paragraph/ Figure/Table/ Note** (e.g. Table 1)	**Type of comment**[2]	**Comment (justification for change) by the MB**	**Proposed change by the MB**	**Secretariat observations** on each comment submitted
					such as for automatic updates of your software, then use cryptographic signatures for your code and modify your download clients to verify the signatures.	
JP-6	7.25	New	Te	Create the 7.25 as shown on the right column	7.Y Incorrect Authorization [???] 7.Y.1 Description of application vulnerability The software performs an authorization check when an actor attempts to access a resource or perform an action, but it does not correctly perform the check. This allows attackers to bypass intended access restrictions. 7.Y.2 Cross reference CWE: 863. Incorrect Authorization 7.Y.3 Mechanism of failure Assuming a user with a given identity, authorization is the process of determining whether that user can access a given resource, based on the user's privileges and any permissions or other access-control specifications that apply to the resource. When access control checks is incorrectly applied, users are able to access data or perform actions that they should not be allowed to perform. This can lead to a wide range of problems, including information exposures, denial of service, and arbitrary code execution. 7.Y.4 Avoiding the vulnerability or mitigating its effects Ensure that you perform access control checks related to your business logic. These checks may be different than the access control checks that you apply to more generic resources such as files, connections, processes, memory, and database records. For example, a database may restrict access for medical records to a specific database user, but each record might only be intended to be accessible to the patient and the patient's doctor.	Agreed in principle. The consensus of the committee is that these vulnerabilities are not mature enough to add to the current working draft of the TR. A drafting committee will be formed to work on these vulnerabilities, making sure that they are included in the 3rd edition of 24772.

1 **MB** = Member body (enter the ISO 3166 two-letter country code, e.g. CN for China; comments from the ISO/CS editing unit are identified by **)

2 **Type of comment:** **ge** = general **te** = technical **ed** = editorial

NOTE Columns 1, 2, 4, 5 are compulsory.

1	2	(3)	4	5	(6)	(7)
MB[1]	**Clause No./ Subclause No./ Annex** (e.g. 3.1)	**Paragraph/ Figure/Table/ Note** (e.g. Table 1)	**Type of comment**[2]	**Comment (justification for change) by the MB**	**Proposed change by the MB**	**Secretariat observations** on each comment submitted
JP-7	7.26	New	Te	Create the 7.26 as shown on the right column	7.X Inclusion of Functionality from Untrusted Control Sphere [???] 7.X.1 Description of application vulnerability The software imports, requires, or includes executable functionality (such as a library) from a source that is outside of the intended control sphere. 7.X.2 Cross reference CWE: 829. Inclusion of Functionality from Untrusted Control Sphere 7.X.3 Mechanism of failure When including third-party functionality, such as a web widget, library, or other source of functionality, the software must effectively trust that functionality. Without sufficient protection mechanisms, the functionality could be malicious in nature (either by coming from an untrusted source, being spoofed, or being modified in transit from a trusted source). The functionality might also contain its own weaknesses, or grant access to additional functionality and state information that should be kept private to the base system, such as system state information, sensitive application data, or the DOM of a web application. This might lead to many different consequences depending on the included functionality, but some examples include injection of malware, information exposure by granting excessive privileges or permissions to the untrusted functionality, DOM-based XSS vulnerabilities, stealing user's cookies, or open redirect to malware. 7.X.4 Avoiding the vulnerability or mitigating its effects Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this	Agreed in principle. The consensus of the committee is that these vulnerabilities are not mature enough to add to the current working draft of the TR. A drafting committee will be formed to work on these vulnerabilities, making sure that they are included in the 3rd edition of 24772.

[1] **MB** = Member body (enter the ISO 3166 two-letter country code, e.g. CN for China; comments from the ISO/CS editing unit are identified by **)

[2] **Type of comment:** **ge** = general **te** = technical **ed** = editorial

NOTE Columns 1, 2, 4, 5 are compulsory.

1	2	(3)	4	5	(6)	(7)
MB[1]	**Clause No./ Subclause No./ Annex** (e.g. 3.1)	**Paragraph/ Figure/Table/ Note** (e.g. Table 1)	**Type of com-**[2] **ment**	**Comment (justification for change) by the MB**	**Proposed change by the MB**	**Secretariat observations** on each comment submitted
					weakness easier to avoid. When the set of acceptable objects, such as filenames or URLs, is limited or known, create a mapping from a set of fixed input values (such as numeric IDs) to the actual filenames or URLs, and reject all other inputs. For example, ID 1 could map to "inbox.txt" and ID 2 could map to "profile.txt". Features such as the ESAPI AccessReferenceMap provide this capability. For any security checks that are performed on the client side, ensure that these checks are duplicated on the server side, in order to avoid CWE-602. Attackers can bypass the client-side checks by modifying values after the checks have been performed, or by changing the client to remove the client-side checks entirely. Then, these modified values would be submitted to the server.	
JP-8	7.27	New	Te	Create the 7.27 as shown on the right column	7.W Improper Restriction of Excessive Authentication Attempts [???] 7.W.1 Description of application vulnerability The software does not implement sufficient measures to prevent multiple failed authentication attempts within in a short time frame, making it more susceptible to brute force attacks. 7.W.2 Cross reference CWE: 307. Improper Restriction of Excessive Authentication Attempts 7.W.3 Mechanism of failure The attacker targeted a member of Twitter's support team and was able to successfully guess the member's password using a brute force attack by guessing a large number of common words. Once the attacker gained access as the member of the support staff, he used the administrator panel to gain access to 33 accounts that	Agreed in principle. The consensus of the committee is that these vulnerabilities are not mature enough to add to the current working draft of the TR. A drafting committee will be formed to work on these vulnerabilities, making sure that they are included in the 3rd edition of 24772.

1 **MB** = Member body (enter the ISO 3166 two-letter country code, e.g. CN for China; comments from the ISO/CS editing unit are identified by **)

2 **Type of comment:** **ge** = general **te** = technical **ed** = editorial

NOTE Columns 1, 2, 4, 5 are compulsory.

1	2	(3)	4	5	(6)	(7)
MB[1]	**Clause No./ Subclause No./ Annex** (e.g. 3.1)	**Paragraph/ Figure/Table/ Note** (e.g. Table 1)	**Type of comment**[2]	**Comment (justification for change) by the MB**	**Proposed change by the MB**	**Secretariat observations** on each comment submitted
					belonged to celebrities and politicians. Ultimately, fake Twitter messages were sent that appeared to come from the compromised accounts. 7.W.4 Avoiding the vulnerability or mitigating its effects Common protection mechanisms include: Disconnecting the user after a small number of failed attempts Implementing a timeout Locking out a targeted account Requiring a computational task on the user's part. Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness easier to avoid. Consider using libraries with authentication capabilities such as OpenSSL or the ESAPIAuthenticator.	
JP-9	7.28	New	Te	Create the 7.28 as shown on the right column	7.V URL Redirection to Untrusted Site ('Open Redirect') [???] 7.V.1 Description of application vulnerability A web application accepts a user-controlled input that specifies a link to an external site, and uses that link in a Redirect. This simplifies phishing attacks. 7.V.2 Cross reference CWE: 601. URL Redirection to Untrusted Site ('Open Redirect') 7.V.3 Mechanism of failure An http parameter may contain a URL value and could cause the web application to redirect the request to the specified URL. By modifying the URL value to a malicious site, an attacker may successfully launch a phishing scam and steal user credentials. Because the server name in the modified link is identical to the	Agreed in principle. The consensus of the committee is that these vulnerabilities are not mature enough to add to the current working draft of the TR. A drafting committee will be formed to work on these vulnerabilities, making sure that they are included in the 3rd edition of 24772.

[1] **MB** = Member body (enter the ISO 3166 two-letter country code, e.g. CN for China; comments from the ISO/CS editing unit are identified by **)

[2] **Type of comment:** **ge** = general **te** = technical **ed** = editorial

NOTE Columns 1, 2, 4, 5 are compulsory.

1	2	(3)	4	5	(6)	(7)
MB[1]	**Clause No./ Subclause No./ Annex** (e.g. 3.1)	**Paragraph/ Figure/Table/ Note** (e.g. Table 1)	**Type of comment**[2]	**Comment (justification for change) by the MB**	**Proposed change by the MB**	**Secretariat observations** on each comment submitted
					original site, phishing attempts have a more trustworthy appearance. 7.V.4 Avoiding the vulnerability or mitigating its effects Input Validation Assume all input is malicious. Use an "accept known good" input validation strategy, i.e., use a whitelist of acceptable inputs that strictly conform to specifications. Reject any input that does not strictly conform to specifications, or transform it into something that does. Do not rely exclusively on looking for malicious or malformed inputs (i.e., do not rely on a blacklist). However, blacklists can be useful for detecting potential attacks or determining which inputs are so malformed that they should be rejected outright. When performing input validation, consider all potentially relevant properties, including length, type of input, the full range of acceptable values, missing or extra inputs, syntax, consistency across related fields, and conformance to business rules. As an example of business rule logic, "boat" may be syntactically valid because it only contains alphanumeric characters, but it is not valid if you are expecting colors such as "red" or "blue." Use a whitelist of approved URLs or domains to be used for redirection.	
JP-10	7.29	New	Te	Create the 7.29 as shown on the right column	7.U Uncontrolled Format String[???] 7.U.1 Description of application vulnerability The software uses externally-controlled format strings in printf-style functions, which can lead to buffer overflows or data representation problems. 7.U.2 Cross reference CWE: 134. Uncontrolled Format String 7.U.3 Mechanism of failure The programmer rarely intends for a format string to be	Agreed in principle. The consensus of the committee is that these vulnerabilities are not mature enough to add to the current working draft of the TR. A drafting committee will be formed to work on these vulnerabilities, making sure that they are included in the 3rd edition of 24772.

1 **MB** = Member body (enter the ISO 3166 two-letter country code, e.g. CN for China; comments from the ISO/CS editing unit are identified by **)

2 **Type of comment:** **ge** = general **te** = technical **ed** = editorial

NOTE Columns 1, 2, 4, 5 are compulsory.

1	2	(3)	4	5	(6)	(7)
MB[1]	**Clause No./ Subclause No./ Annex** (e.g. 3.1)	**Paragraph/ Figure/Table/ Note** (e.g. Table 1)	**Type of com-[2] ment**	**Comment (justification for change) by the MB**	**Proposed change by the MB**	**Secretariat observations** on each comment submitted
					user-controlled at all. This weakness is frequently introduced in code that constructs log messages, where a constant format string is omitted. In cases such as localization and internationalization, the language-specific message repositories could be an avenue for exploitation, but the format string issue would be resultant, since attacker control of those repositories would also allow modification of message length, format, and content. 7.U.4 Avoiding the vulnerability or mitigating its effects Ensure that all format string functions are passed as static string which cannot be controlled by the user and that the proper number of arguments are always sent to that function as well. If at all possible, use functions that do not support the %n operator in format strings.	
JP-11	7.30	New	Te	Create the 7.30 as shown on the right column	7.T Use of a One-Way Hash without a Salt [???] 7.T.1 Description of application vulnerability The software uses a one-way cryptographic hash against an input that should not be reversible, such as a password, but the software does not also use a salt as part of the input. 7.T.2 Cross reference CWE: 759. Use of a One-Way Hash without a Salt 7.T.3 Mechanism of failure This makes it easier for attackers to pre-compute the hash value using dictionary attack techniques such as rainbow tables. 7.T.4 Avoiding the vulnerability or mitigating its effects Generate a random salt each time you process a new password. Add the salt to the plaintext password before hashing it. When you store the hash, also store the salt. Do not use the same salt for every password that you	Agreed in principle. The consensus of the committee is that these vulnerabilities are not mature enough to add to the current working draft of the TR. A drafting committee will be formed to work on these vulnerabilities, making sure that they are included in the 3rd edition of 24772.

[1] **MB** = Member body (enter the ISO 3166 two-letter country code, e.g. CN for China; comments from the ISO/CS editing unit are identified by **)

[2] **Type of comment:** **ge** = general **te** = technical **ed** = editorial

NOTE Columns 1, 2, 4, 5 are compulsory.

1	2	(3)	4	5	(6)	(7)
MB[1]	**Clause No./ Subclause No./ Annex** (e.g. 3.1)	**Paragraph/ Figure/Table/ Note** (e.g. Table 1)	**Type of comment**[2]	**Comment (justification for change) by the MB**	**Proposed change by the MB**	**Secretariat observations** on each comment submitted
					process. Use one-way hashing techniques that allow you to configure a large number of rounds, such as bcrypt. This may increase the expense when processing incoming authentication requests, but if the hashed passwords are ever stolen, it significantly increases the effort for conducting a brute force attack, including rainbow tables. With the ability to configure the number of rounds, you can increase the number of rounds whenever CPU speeds or attack techniques become more efficient. When you use industry-approved techniques, you need to use them correctly. Don't cut corners by skipping resource-intensive steps (CWE-325). These steps are often essential for preventing common attacks.	

1 **MB** = Member body (enter the ISO 3166 two-letter country code, e.g. CN for China; comments from the ISO/CS editing unit are identified by ******)

2 **Type of comment:** **ge** = general **te** = technical **ed** = editorial

NOTE Columns 1, 2, 4, 5 are compulsory.