

ISO/IEC 24772-1:2024-10 (E)

Programming languages - Avoiding vulnerabilities in programming languages - Part 1: Language-independent catalogue of vulnerabilities

Contents		Page
Foreword.....		xv
Introduction.....		xvii
1	Scope.....	1
2	Normative references.....	1
3	Terms and definitions.....	1
	3.1 Communication.....	1
	3.2 Execution model.....	1
	3.3 Properties.....	2
	3.4 Safety and security.....	3
	3.5 Vulnerabilities.....	3
	3.6 Specific vulnerabilities.....	3
4	Using this document.....	4
	4.1 Purpose of this document.....	4
	4.2 Applying this document.....	5
	4.3 Structure of this document.....	6
5	General vulnerability issues and primary avoidance mechanisms.....	7
	5.1 General vulnerability issues.....	7
	5.1.1 Predictable execution.....	7
	5.1.2 Sources of unpredictability in language specification.....	8
	5.1.3 Sources of unpredictability in language usage.....	9
	5.2 Primary avoidance mechanisms.....	9
6	Programming language vulnerabilities.....	11
	6.1 General.....	11
	6.2 Type system [IHN].....	12
	6.2.1 Description of application vulnerability.....	12
	6.2.2 Related coding guidelines.....	12
	6.2.3 Mechanism of failure.....	12
	6.2.4 Applicable language characteristics.....	13
	6.2.5 Avoiding the vulnerability or mitigating its effects.....	13
	6.2.6 Implications for language design and evolution.....	14
	6.3 Bit representations [STR].....	14
	6.3.1 Description of application vulnerability.....	14
	6.3.2 Related coding guidelines.....	14
	6.3.3 Mechanism of failure.....	15
	6.3.4 Applicable language characteristics.....	15
	6.3.5 Avoiding the vulnerability or mitigating its effects.....	15
	6.3.6 Implications for language design and evolution.....	16
	6.4 Floating-point arithmetic [PLF].....	16
	6.4.1 Description of application vulnerability.....	16
	6.4.2 Related coding guidelines.....	16
	6.4.3 Mechanism of failure.....	16
	6.4.4 Applicable language characteristics.....	17
	6.4.5 Avoiding the vulnerability or mitigating its effects.....	17
	6.4.6 Implications for language design and evolution.....	18
	6.5 Enumerator issues [CCB].....	18
	6.5.1 Description of application vulnerability.....	18
	6.5.2 Related coding guidelines.....	19

6.5.3	Mechanism of failure.....	19
6.5.4	Applicable language Characteristics.....	19
6.5.5	Avoiding the vulnerability or mitigating its effects.....	20
6.5.6	Implications for language design and evolution.....	20
6.6	Conversion errors [FLC].....	20
6.6.1	Description of application vulnerability.....	20
6.6.2	Related coding guidelines.....	20
6.6.3	Mechanism of failure.....	21
6.6.4	Applicable language characteristics.....	21
6.6.5	Avoiding the vulnerability or mitigating its effects.....	21
6.6.6	Implications for language design and evolution.....	22
6.7	String termination [CJM].....	22
6.7.1	Description of application vulnerability.....	22
6.7.2	Related coding guidelines.....	22
6.7.3	Mechanism of failure.....	22
6.7.4	Applicable language characteristics.....	22
6.7.5	Avoiding the vulnerability or mitigating its effects.....	23
6.7.6	Implications for language design and evolution.....	23
6.8	Buffer boundary violation (buffer overflow) [HCB].....	23
6.8.1	Description of application vulnerability.....	23
6.8.2	Related coding guidelines.....	23
6.8.3	Mechanism of failure.....	24
6.8.4	Applicable language characteristics.....	24
6.8.5	Avoiding the vulnerability or mitigating its effects.....	24
6.8.6	Implications for language design and evolution.....	25
6.9	Unchecked array indexing [XYZ].....	25
6.9.1	Description of application vulnerability.....	25
6.9.2	Related coding guidelines.....	25
6.9.3	Mechanism of failure.....	25
6.9.4	Applicable language characteristics.....	26
6.9.5	Avoiding the vulnerability or mitigating its effects.....	26
6.9.6	Implications for language designers.....	26
6.10	Unchecked array copying [XYW].....	27
6.10.1	Description of application vulnerability.....	27
6.10.2	Related coding guidelines.....	27
6.10.3	Mechanism of failure.....	27
6.10.4	Applicable language characteristics.....	27
6.10.5	Avoiding the vulnerability or mitigating its effects.....	28
6.10.6	Implications for language design and evolution.....	28
6.11	Pointer type conversions [HFC].....	28
6.11.1	Description of application vulnerability.....	28
6.11.2	Related coding guidelines.....	28
6.11.3	Mechanism of failure.....	29
6.11.4	Applicable language characteristics.....	29
6.11.5	Avoiding the vulnerability or mitigating its effects.....	29
6.11.6	Implications for language design and evolution.....	29
6.12	Pointer arithmetic [RVG].....	29
6.12.1	Description of application vulnerability.....	29
6.12.2	Related coding guidelines.....	29
6.12.3	Mechanism of failure.....	29
6.12.4	Applicable language characteristics.....	30
6.12.5	Avoiding the vulnerability or mitigating its effects.....	30
6.12.6	Implications for language design and evolution.....	30
6.13	Null pointer dereference [XYH].....	30
6.13.1	Description of application vulnerability.....	30
6.13.2	Related coding guidelines.....	30
6.13.3	Mechanism of failure.....	30
6.13.4	Applicable language characteristics.....	30
6.13.5	Avoiding the vulnerability or mitigating its effects.....	30
6.13.6	Implications for language design and evolution.....	31
6.14	Dangling reference to heap [XYK].....	31
6.14.1	Description of application vulnerability.....	31
6.14.2	Related coding guidelines.....	31
6.14.3	Mechanism of failure.....	31
6.14.4	Applicable language characteristics.....	32

6.14.5	Avoiding the vulnerability or mitigating its effects	32
6.14.6	Implications for language design and evolution	32
6.15	Arithmetic wrap-around error [FIF]	33
6.15.1	Description of application vulnerability	33
6.15.2	Related coding guidelines	33
6.15.3	Mechanism of failure	33
6.15.4	Applicable language characteristics	34
6.15.5	Avoiding the vulnerability or mitigating its effects	34
6.15.6	Implications for language design and evolution	34
6.16	Using shift operations for multiplication and division [PIK]	34
6.16.1	Description of application vulnerability	34
6.16.2	Related coding guidelines	34
6.16.3	Mechanism of failure	34
6.16.4	Applicable language characteristics	34
6.16.5	Avoiding the vulnerability or mitigating its effects	35
6.16.6	Implications for language design and evolution	35
6.17	Choice of clear names [NAI]	35
6.17.1	Description of application vulnerability	35
6.17.2	Related coding guidelines	36
6.17.3	Mechanism of Failure	36
6.17.4	Applicable language characteristics	36
6.17.5	Avoiding the vulnerability or mitigating its effects	36
6.17.6	Implications for language design and evolution	37
6.18	Dead store [WXQ]	37
6.18.1	Description of application vulnerability	37
6.18.2	Related coding guidelines	37
6.18.3	Mechanism of failure	37
6.18.4	Applicable language characteristics	37
6.18.5	Avoiding the vulnerability or mitigating its effects	38
6.18.6	Implications for language design and evolution	38
6.19	Unused variable [YZS]	38
6.19.1	Description of application vulnerability	38
6.19.2	Related coding guidelines	38
6.19.3	Mechanism of failure	38
6.19.4	Applicable language characteristics	38
6.19.5	Avoiding the vulnerability or mitigating its effects	38
6.19.6	Implications for language design and evolution	39
6.20	Identifier name reuse [YOW]	39
6.20.1	Description of application vulnerability	39
6.20.2	Related coding guidelines	39
6.20.3	Mechanism of failure	39
6.20.4	Applicable language characteristics	40
6.20.5	Avoiding the vulnerability or mitigating its effects	40
6.20.6	Implications for language design and evolution	40
6.21	Namespace issues [BJL]	41
6.21.1	Description of application vulnerability	41
6.21.2	Related coding guidelines	41
6.21.3	Mechanism of Failure	41
6.21.4	Applicable language characteristics	41
6.21.5	Avoiding the Vulnerability or Mitigating its Effects	42
6.21.6	Implications for language design and evolution	42
6.22	Missing initialization of variables [LAV]	42
6.22.1	Description of application vulnerability	42
6.22.2	Related coding guidelines	42
6.22.3	Mechanism of failure	43
6.22.4	Applicable language characteristics	43
6.22.5	Avoiding the vulnerability or mitigating its effects	43
6.22.6	Implications for language design and evolution	44
6.23	Operator precedence and associativity [JCW]	44

6.23.1	Description of application vulnerability	44
6.23.2	Related coding guidelines	44
6.23.3	Mechanism of failure	45
6.23.4	Applicable language characteristics	45
6.23.5	Avoiding the vulnerability or mitigating its effects	45
6.23.6	Implications for language design and evolution	45
6.24	Side-effects and order of evaluation of operands [SAM]	45
6.24.1	Description of application vulnerability	45
6.24.2	Related coding guidelines	46
6.24.3	Mechanism of failure	46
6.24.4	Applicable language characteristics	47
6.24.5	Avoiding the vulnerability or mitigating its effects	47
6.24.6	Implications for language design and evolution	47
6.25	Likely incorrect expression [KOA]	47
6.25.1	Description of application vulnerability	47
6.25.2	Related coding guidelines	47
6.25.3	Mechanism of failure	48
6.25.4	Applicable language characteristics	48
6.25.5	Avoiding the vulnerability or mitigating its effects	48
6.25.6	Implications for language design and evolution	48
6.26	Dead and deactivated code [XYQ]	49
6.26.1	Description of application vulnerability	49
6.26.2	Related coding guidelines	49
6.26.3	Mechanism of failure	49
6.26.4	Applicable language characteristics	50
6.26.5	Avoiding the vulnerability or mitigating its effects	50
6.26.6	Implications for language design and evolution	50
6.27	Switch statements and lack of static analysis [CLL]	51
6.27.1	Description of application vulnerability	51
6.27.2	Related coding guidelines	51
6.27.3	Mechanism of failure	51
6.27.4	Applicable language characteristics	51
6.27.5	Avoiding the vulnerability or mitigating its effects	51
6.27.6	Implications for language design and evolution	52
6.28	Non-demarcation of control flow [EOJ]	52
6.28.1	Description of application vulnerability	52
6.28.2	Related coding guidelines	52
6.28.3	Mechanism of failure	52
6.28.4	Applicable language characteristics	52
6.28.5	Avoiding the vulnerability or mitigating its effects	52
6.28.6	Implications for language design and evolution	53
6.29	Loop control variable abuse [TEX]	53
6.29.1	Description of application vulnerability	53
6.29.2	Related coding guidelines	53
6.29.3	Mechanism of failure	54
6.29.4	Applicable language characteristics	54
6.29.5	Avoiding the vulnerability or mitigating its effects	54
6.29.6	Implications for language design and evolution	54
6.30	Off-by-one error [XZH]	54
6.30.1	Description of application vulnerability	54
6.30.2	Related coding guidelines	55
6.30.3	Mechanism of failure	55
6.30.4	Applicable language characteristics	55
6.30.5	Avoiding the vulnerability or mitigating its effects	55
6.30.6	Implications for language design and evolution	55
6.31	Unstructured programming [EWD]	56
6.31.1	Description of application vulnerability	56
6.31.2	Related coding guidelines	56
6.31.3	Mechanism of failure	56

6.31.4	Applicable language characteristics.....	56
6.31.5	Avoiding the vulnerability or mitigating its effects.....	56
6.31.6	Implications for language design and evolution.....	57
6.32	Passing parameters and return values [CSJ].....	57
6.32.1	Description of application vulnerability.....	57
6.32.2	Related coding guidelines.....	57
6.32.3	Mechanism of failure.....	57
6.32.4	Applicable language characteristics.....	58
6.32.5	Avoiding the vulnerability or mitigating its effects.....	58
6.32.6	Implications for language design and evolution.....	59
6.33	Dangling references to stack frames [DCM].....	59
6.33.1	Description of application vulnerability.....	59
6.33.2	Related coding guidelines.....	59
6.33.3	Mechanism of failure.....	59
6.33.4	Applicable language characteristics.....	60
6.33.5	Avoiding the vulnerability or mitigating its effects.....	60
6.33.6	Implications for language design and evolution.....	60
6.34	Subprogram signature mismatch [OTR].....	61
6.34.1	Description of application vulnerability.....	61
6.34.2	Related coding guidelines.....	61
6.34.3	Mechanism of failure.....	61
6.34.4	Applicable language characteristics.....	61
6.34.5	Avoiding the vulnerability or mitigating its effects.....	62
6.34.6	Implications for language design and evolution.....	62
6.35	Recursion [GDL].....	62
6.35.1	Description of application vulnerability.....	62
6.35.2	Related coding guidelines.....	62
6.35.3	Mechanism of failure.....	62
6.35.4	Applicable language characteristics.....	63
6.35.5	Avoiding the vulnerability or mitigating its effects.....	63
6.35.6	Implications for language design and evolution.....	63
6.36	Ignored error status and unhandled exceptions [OYB].....	63
6.36.1	Description of application vulnerability.....	63
6.36.2	Related coding guidelines.....	63
6.36.3	Mechanism of failure.....	63
6.36.4	Applicable language characteristics.....	64
6.36.5	Avoiding the vulnerability or mitigating its effects.....	64
6.36.6	Implications for language design and evolution.....	65
6.37	Type-breaking reinterpretation of data [AMV].....	65
6.37.1	Description of application vulnerability.....	65
6.37.2	Related coding guidelines.....	65
6.37.3	Mechanism of failure.....	66
6.37.4	Applicable language characteristics.....	66
6.37.5	Avoiding the vulnerability or mitigating its effects.....	66
6.37.6	Implications for language design and evolution.....	67
6.38	Deep vs. shallow copying [YAN].....	67
6.38.1	Description of application vulnerability.....	67
6.38.2	Related coding guidelines.....	67
6.38.3	Mechanism of failure.....	67
6.38.4	Applicable language characteristics.....	67
6.38.5	Avoiding the vulnerability or mitigating its effects.....	68
6.38.6	Implications for language design and evolution.....	68
6.39	Memory leaks and heap fragmentation [XYL].....	68
6.39.1	Description of application vulnerability.....	68
6.39.2	Related coding guidelines.....	68
6.39.3	Mechanism of failure.....	68
6.39.4	Applicable language characteristics.....	69
6.39.5	Avoiding the vulnerability or mitigating its effects.....	69
6.39.6	Implications for language design and evolution.....	69

6.40	Templates and generics [SYM]	70
6.40.1	Description of application vulnerability	70
6.40.2	Related coding guidelines	70
6.40.3	Mechanism of failure	70
6.40.4	Applicable language characteristics	71
6.40.5	Avoiding the vulnerability or mitigating its effects	71
6.40.6	Implications for language design and evolution	71
6.41	Inheritance [RIP]	71
6.41.1	Description of application vulnerability	71
6.41.2	Related coding guidelines	71
6.41.3	Mechanism of failure	72
6.41.4	Applicable language characteristics	72
6.41.5	Avoiding the vulnerability or mitigating its effects	72
6.41.6	Implications for language design and evolution	73
6.42	Violations of the Liskov substitution principle or the contract model [BLP]	73
6.42.1	Description of application vulnerability	73
6.42.2	Related coding guidelines	73
6.42.3	Mechanism of failure	74
6.42.4	Applicable language characteristics	74
6.42.5	Avoiding the vulnerability or mitigating its effects	74
6.42.6	Implications for language design and evolution	74
6.43	Redispatching [PPH]	74
6.43.1	Description of application vulnerability	74
6.43.2	Related coding guidelines	75
6.43.3	Mechanism of failure	75
6.43.4	Applicable language characteristics	75
6.43.5	Avoiding the vulnerability or mitigating its effects	75
6.43.6	Implications for language design and evolution	75
6.44	Polymorphic variables [BKK]	75
6.44.1	Description of application vulnerability	75
6.44.2	Related coding guidelines	76
6.44.3	Mechanism of failure	76
6.44.4	Applicable language characteristics	76
6.44.5	Avoiding the vulnerability or mitigating its effects	77
6.44.6	Implications for language design and evolution	77
6.45	Extra intrinsics [LRM]	77
6.45.1	Description of application vulnerability	77
6.45.2	Related coding guidelines	77
6.45.3	Mechanism of failure	77
6.45.4	Applicable language characteristics	77
6.45.5	Avoiding the vulnerability or mitigating its effects	78
6.45.6	Implications for language design and evolution	78
6.46	Argument passing to library functions [TRJ]	78
6.46.1	Description of application vulnerability	78
6.46.2	Related coding guidelines	78
6.46.3	Mechanism of failure	78
6.46.4	Applicable language characteristics	78
6.46.5	Avoiding the vulnerability or mitigating its effects	79
6.46.6	Implications for language design and evolution	79
6.47	Inter-language calling [DJS]	79
6.47.1	Description of application vulnerability	79
6.47.2	Related coding guidelines	79
6.47.3	Mechanism of failure	79
6.47.4	Applicable language characteristics	80
6.47.5	Avoiding the vulnerability or mitigating its effects	80
6.47.6	Implications for language design and evolution	81
6.48	Dynamically-linked code and self-modifying code [NYY]	81
6.48.1	Description of application vulnerability	81
6.48.2	Related coding guidelines	81

6.48.3	Mechanism of failure	81
6.48.4	Applicable language characteristics	81
6.48.5	Avoiding the vulnerability or mitigating its effects	82
6.48.6	Implications for language design and evolution	82
6.49	Library signature [NSQ]	82
6.49.1	Description of application vulnerability	82
6.49.2	Related coding guidelines	82
6.49.3	Mechanism of failure	82
6.49.4	Applicable language characteristics	83
6.49.5	Avoiding the vulnerability or mitigating its effects	83
6.49.6	Implications for language design and evolution	83
6.50	Unanticipated exceptions from library routines [HJW]	83
6.50.1	Description of application vulnerability	83
6.50.2	Cross reference	83
6.50.3	Related coding guidelines	83
6.50.4	Applicable language characteristics	83
6.50.5	Avoiding the vulnerability or mitigating its effects	84
6.50.6	Implications for language design and evolution	84
6.51	Pre-processor directives [NMP]	84
6.51.1	Description of application vulnerability	84
6.51.2	Related coding guidelines	84
6.51.3	Mechanism of failure	84
6.51.4	Applicable language characteristics	85
6.51.5	Avoiding the vulnerability or mitigating its effects	85
6.51.6	Implications for language design and evolution	85
6.52	Suppression of language-defined run-time checking [MXB]	85
6.52.1	Description of application vulnerability	85
6.52.2	Related coding guidelines	86
6.52.3	Mechanism of Failure	86
6.52.4	Applicable language characteristics	86
6.52.5	Avoiding the vulnerability	86
6.52.6	Implications for language design and evolution	86
6.53	Provision of inherently unsafe operations [SKL]	86
6.53.1	Description of application vulnerability	86
6.53.2	Related coding guidelines	87
6.53.3	Mechanism of Failure	87
6.53.4	Applicable language characteristics	87
6.53.5	Avoiding the vulnerability or mitigating its effect	87
6.53.6	Implications for language design and evolution	87
6.54	Obscure language features [BRS]	87
6.54.1	Description of application vulnerability	87
6.54.2	Related coding guidelines	88
6.54.3	Mechanism of failure	88
6.54.4	Applicable language characteristics	88
6.54.5	Avoiding the vulnerability or mitigating its effects	88
6.54.6	Implications for language design and evolution	89
6.55	Unspecified behaviour [BQF]	89
6.55.1	Description of application vulnerability	89
6.55.2	Related coding guidelines	89
6.55.3	Mechanism of failure	89
6.55.4	Applicable language characteristics	90
6.55.5	Avoiding the vulnerability or mitigating its effects	90
6.55.6	Implications for language design and evolution	90
6.56	Undefined behaviour [EWF]	90
6.56.1	Description of application vulnerability	90
6.56.2	Related coding guidelines	90
6.56.3	Mechanism of failure	91
6.56.4	Applicable language characteristics	91
6.56.5	Avoiding the vulnerability or mitigating its effects	91

6.56.6	Implications for language design and evolution	91
6.57	Implementation-defined behaviour [FAB]	91
6.57.1	Description of application vulnerability	91
6.57.2	Related coding guidelines	92
6.57.3	Mechanism of failure	92
6.57.4	Applicable language characteristics	92
6.57.5	Avoiding the vulnerability or mitigating its effects	92
6.57.6	Implications for language design and evolution	93
6.58	Deprecated language features [MEM]	93
6.58.1	Description of application vulnerability	93
6.58.2	Related coding guidelines	93
6.58.3	Mechanism of failure	93
6.58.4	Applicable language characteristics	94
6.58.5	Avoiding the vulnerability or mitigating its effects	94
6.58.6	Implications for language design and evolution	94
6.59	Concurrency – Activation [CGA]	94
6.59.1	Description of application vulnerability	94
6.59.2	Related coding guidelines	94
6.59.3	Mechanism of Failure	95
6.59.4	Applicable language characteristics	95
6.59.5	Avoiding the vulnerability or mitigating its effects	95
6.59.6	Implications for language design and evolution	96
6.60	Concurrency – Directed termination [CGT]	96
6.60.1	Description of application vulnerability	96
6.60.2	Related coding guidelines	96
6.60.3	Mechanism of failure	96
6.60.4	Applicable language characteristics	97
6.60.5	Avoiding the vulnerability or mitigating its effect	97
6.60.6	Implications for language design and evolution	97
6.61	Concurrent data access [CGX]	97
6.61.1	Description of application vulnerability	97
6.61.2	Related coding guidelines	97
6.61.3	Mechanism of failure	98
6.61.4	Applicable language characteristics	98
6.61.5	Avoiding the vulnerability or mitigating its effect	98
6.61.6	Implications for language design and evolution	98
6.62	Concurrency – Premature termination [CGS]	99
6.62.1	Description of application vulnerability	99
6.62.2	Related coding guidelines	99
6.62.3	Mechanism of failure	99
6.62.4	Applicable language characteristics	100
6.62.5	Avoiding the vulnerability or mitigating its effect	100
6.62.6	Implications for language design and evolution	100
6.63	Lock protocol errors [CGM]	100
6.63.1	Description of application vulnerability	100
6.63.2	Related coding guidelines	101
6.63.3	Mechanism of failure	101
6.63.4	Applicable language characteristics	102
6.63.5	Avoiding the vulnerability or mitigating its effect	102
6.63.6	Implications for language design and evolution	102
6.64	Reliance on external format strings [SHL]	103
6.64.1	Description of application vulnerability	103
6.64.2	Related coding guidelines	103
6.64.3	Mechanism of failure	103
6.64.4	Applicable language characteristics	103
6.64.5	Avoiding the vulnerability or mitigating its effects	104
6.64.6	Implications for language design and evolution	104
6.65	Modifying constants [UJO]	104
6.65.1	Description of application vulnerability	104

6.65.2	Related coding guidelines	104
6.65.3	Mechanism of failure	104
6.65.4	Applicable language characteristics	105
6.65.5	Avoiding the vulnerability or mitigating its effects	105
6.65.6	Implications for language design and evolution	105
7	Application vulnerabilities	105
7.1	General	105
7.2	Unrestricted file upload [CBF]	105
7.2.1	Description of application vulnerability	105
7.2.2	Related coding guidelines	105
7.2.3	Mechanism of failure	106
7.2.4	Avoiding the vulnerability or mitigating its effects	106
7.3	Download of code without integrity check [DLB]	106
7.3.1	Description of application vulnerability	106
7.3.2	Related coding guidelines	107
7.3.3	Mechanism of failure	107
7.3.4	Avoiding the vulnerability or mitigating its effects	107
7.4	Executing or loading untrusted code [XYS]	107
7.4.1	Description of application vulnerability	107
7.4.2	Related coding guidelines	107
7.4.3	Mechanism of failure	107
7.4.4	Avoiding the vulnerability or mitigating its effects	108
7.5	Inclusion of functionality from untrusted control sphere [DHU]	108
7.5.1	Description of application vulnerability	108
7.5.2	Related coding guidelines	108
7.5.3	Mechanism of failure	108
7.5.4	Avoiding the vulnerability or mitigating its effects	108
7.6	Use of unchecked data from an uncontrolled or tainted source [EFS]	109
7.6.1	Description of application vulnerability	109
7.6.2	Related coding guidelines	109
7.6.3	Mechanism of failure	109
7.6.4	Avoiding the vulnerability or mitigating its effects	109
7.7	Cross-site scripting [XYT]	110
7.7.1	Description of application vulnerability	110
7.7.2	Related coding guidelines	110
7.7.3	Mechanism of failure	110
7.7.4	Avoiding the vulnerability or mitigating its effects	111
7.8	URL redirection to untrusted site ("open redirect") [PYQ]	112
7.8.1	Description of application vulnerability	112
7.8.2	Related coding guidelines	112
7.8.3	Mechanism of failure	112
7.8.4	Avoiding the vulnerability or mitigating its effects	112
7.9	Injection [RST]	113
7.9.1	Description of application vulnerability	113
7.9.2	Related coding guidelines	113
7.9.3	Mechanism of failure	114
7.9.4	Avoiding the vulnerability or mitigating its effects	115
7.10	Unquoted search path or element [XZQ]	115
7.10.1	Description of application vulnerability	115
7.10.2	Related coding guidelines	115
7.10.3	Mechanism of failure	116
7.10.4	Avoiding the vulnerability or mitigating its effects	116
7.11	Path traversal [EWR]	116
7.11.1	Description of application vulnerability	116
7.11.2	Related coding guidelines	116
7.11.3	Mechanism of failure	117
7.11.4	Avoiding the vulnerability or mitigating its effects	118
7.12	Resource names [HTS]	118

7.12.1	Description of application vulnerability	118
7.12.2	Related coding guidelines	119
7.12.3	Mechanism of Failure	119
7.12.4	Avoiding the vulnerability or mitigating its effects	119
7.13	Resource exhaustion [XZP]	119
7.13.1	Description of application vulnerability	119
7.13.2	Related coding guidelines	120
7.13.3	Mechanism of failure	120
7.13.4	Avoiding the vulnerability or mitigating its effects	120
7.14	Authentication logic error [XZO]	121
7.14.1	Description of application vulnerability	121
7.14.2	Related coding guidelines	121
7.14.3	Mechanism of failure	121
7.14.4	Avoiding the vulnerability or mitigating its effects	122
7.15	Improper restriction of excessive authentication attempts [WPL]	122
7.15.1	Description of application vulnerability	122
7.15.2	Related coding guidelines	122
7.15.3	Mechanism of failure	122
7.15.4	Avoiding the vulnerability or mitigating its effects	123
7.16	Hard-coded credentials [XYP]	123
7.16.1	Description of application vulnerability	123
7.16.2	Related coding guidelines	123
7.16.3	Mechanism of failure	123
7.16.4	Avoiding the vulnerability or mitigating its effects	124
7.17	Insufficiently protected credentials [XYM]	124
7.17.1	Description of application vulnerability	124
7.17.2	Related coding guidelines	124
7.17.3	Mechanism of failure	124
7.17.4	Avoiding the vulnerability or mitigating its effects	124
7.18	Missing or inconsistent access control [XZN]	125
7.18.1	Description of application vulnerability	125
7.18.2	Related coding guidelines	125
7.18.3	Mechanism of failure	125
7.18.4	Avoiding the vulnerability or mitigating its effects	125
7.19	Incorrect authorization [BJE]	125
7.19.1	Description of application vulnerability	125
7.19.2	Related coding guidelines	125
7.19.3	Mechanism of failure	125
7.19.4	Avoiding the vulnerability or mitigating its effects	126
7.20	Adherence to least privilege [XYN]	126
7.20.1	Description of application vulnerability	126
7.20.2	Related coding guidelines	126
7.20.3	Mechanism of failure	126
7.20.4	Avoiding the vulnerability or mitigating its effects	126
7.21	Privilege sandbox issues [XYO]	127
7.21.1	Description of application vulnerability	127
7.21.2	Related coding guidelines	127
7.21.3	Mechanism of failure	127
7.21.4	Avoiding the vulnerability or mitigating its effects	128
7.22	Missing required cryptographic step [XZS]	128
7.22.1	Description of application vulnerability	128
7.22.2	Related coding guidelines	128
7.22.3	Mechanism of failure	128
7.22.4	Avoiding the vulnerability or mitigating its effects	128
7.23	Improperly verified signature [XZR]	129
7.23.1	Description of application vulnerability	129
7.23.2	Related coding guidelines	129
7.23.3	Mechanism of failure	129
7.23.4	Avoiding the vulnerability or mitigating its effects	129

7.24	Use of a one-way hash without a salt [MVX]	129
7.24.1	Description of application vulnerability	129
7.24.2	Related coding guidelines	129
7.24.3	Mechanism of failure	129
7.24.4	Avoiding the vulnerability or mitigating its effects	129
7.25	Inadequately secure communication of shared resources [CGY]	130
7.25.1	Description of application vulnerability	130
7.25.2	Related coding guidelines	130
7.25.3	Mechanism of failure	130
7.25.4	Avoiding the vulnerability or mitigating its effect	131
7.26	Memory locking [XZX]	131
7.26.1	Description of application vulnerability	131
7.26.2	Related coding guidelines	131
7.26.3	Mechanism of failure	131
7.26.4	Avoiding the vulnerability or mitigating its effects	132
7.27	Sensitive information not cleared before use [XZK]	132
7.27.1	Description of application vulnerability	132
7.27.2	Related coding guidelines	132
7.27.3	Mechanism of failure	132
7.27.4	Avoiding the vulnerability or mitigating its effects	132
7.28	Time consumption measurement [CCM]	132
7.28.1	Description of application vulnerability	132
7.28.2	Related coding guidelines	133
7.28.3	Mechanism of failure	133
7.28.4	Avoiding the vulnerability or mitigating its effect	133
7.29	Discrepancy information leak [XZL]	133
7.29.1	Description of application vulnerability	133
7.29.2	Related coding guidelines	134
7.29.3	Mechanism of failure	134
7.29.4	Avoiding the vulnerability or mitigating its effects	134
7.30	Unspecified functionality [BVQ]	134
7.30.1	Description of application vulnerability	134
7.30.2	Related coding guidelines	134
7.30.3	Mechanism of failure	135
7.30.4	Avoiding the vulnerability or mitigating its effects	135
7.31	Fault tolerance and failure strategies [REU]	135
7.31.1	Description of application vulnerability	135
7.31.2	Related coding guidelines	136
7.31.3	Mechanism of failure	136
7.31.4	Avoiding the vulnerability or mitigating its effects	137
7.32	Distinguished values in data types [KLK]	137
7.32.1	Description of application vulnerability	137
7.32.2	Related coding guidelines	137
7.32.3	Mechanism of failure	138
7.32.4	Avoiding the vulnerability or mitigating its effects	138
7.33	Clock issues [CCI]	139
7.33.1	Description of application vulnerability	139
7.33.2	Related coding guidelines	139
7.33.3	Mechanism of failure	139
7.33.4	Avoiding the vulnerability or mitigating its effect	141
7.34	Time drift and jitter [CDJ]	141
7.34.1	Description of application vulnerability	141
7.34.2	Related coding guidelines	142
7.34.3	Mechanism of failure	142
7.34.4	Avoiding the vulnerability or mitigating its effect	142
Annex A (informative) Vulnerability taxonomy and list		143
Annex B (informative) Selected principles for language designers		150
Bibliography		152