TRUSTWORTHY SOFTWARE

PATCHING GUIDANCE

April 2016
Issue 1.2

TS543-1-01

DOCUMENT CONTROL

CHANGE RECORD

VERSION	DATE	SUMMARY
0.A	06/03/2015	First draft of document, for internal TSI comment
0.B	23/06/2015	Revised from internal comments
0.C	26/06/2015	Scheduling Table removed
1.0	30/06/2015	Initial Issue
1.1	02/07/2015	Minor comments applied
1.2	04/07/2015	Additional Guidance in respect of Scheduling and Roll-out

DOCUMENT INFORMATION

DOCUMENT ID	TSI/2014/232
PRODUCTION ID	N/A
TSK-ID	TS543-1-01
PUBLISHER	UK-TSI, IMC, Westwood Heath, CV4 7AL
RIGHTS.COPYRIGHT:	
RIGHTS.CUSTODIAN:	TSI Programme Support Officer (+44 300 030 1928 / enquiries@uk-tsi.org)
RIGHTS.PROTECTIVEMARKING:	NOT PROTECTIVELY MARKED

FOREWORD

TRUSTWORTHY SOFTWARE INITIATIVE

The Trustworthy Software Initiative (TSI) is part of the UK Government's National Cyber Security Programme to improve the UK's ability to combat cyber risks and ensure that the UK leads the way in trustworthy software systems and expertise.

The objective of TSI is to provide the knowledge, skills and capability for supply, demand and education communities such that trustworthy software can be designed, implemented, sustainably maintained and assured in a risk-based, whole-life process.

TSI works with organisations and individuals in the UK, including government, academia, private/public companies, software developers and users, to achieve a recognised level of trust of software by providing targeted education, skills, standards and guidance.

For more information visit the TSI website: www.uk-tsi.org

LICENCE CONDITIONS

DISCLAIMER

CONTENTS

1 INTRODUCTION

1.1. TRUSTWORTHY SOFTWARE

OVERVIEW

With our daily lives and industrial processes becoming increasingly reliant on a wide range of underpinning software, there is a pressing need to address the quality and robustness of that software in order to establish its "trustworthiness" and therefore ensure that it performs as it should, when it should and how it should.

This means addressing the trustworthiness of software throughout its life-cycle, from development through to disposal.

FACETS OF TRUSTWORTHINESS

TSI identifies trustworthiness to predominantly consist of the following five facets:

- Safety - the ability of the system to operate without harmful states
- Reliability - the ability of the system to deliver services as specified
- Availability - the ability of the system to deliver services when requested
- Resilience - the ability of the system to transform, renew and recover in timely response to events
- Security - the ability of the system to remain protected against accidental or deliberate attacks

1.2. PURPOSE AND APPLICABILITY OF DOCUMENT

This document provides guidance as to how Patching practices should be adopted by organisations in order to ensure the trustworthiness of the software they produce, procure or use.

It provides a mapping of the overall concepts, principles and techniques for trustworthy software to this particular lifecycle activity, and signposts related sources of amplifying information that may not otherwise been known to the readership.

It is intended as essential reading for those in the organisation responsible for implementing the decision to adopt trustworthy software, and is applicable to organisations of all sizes.

1.3. OTHER DOCUMENTS

For all associated and/or reference documents, please see Annex A: Bibliography.

1.4. FURTHER INFORMATION

For further information relating to other aspects of Trustworthy Software, including the more comprehensive Trustworthy Software Framework, please visit: www.uk-tsi.org.

A summary version of this information is also available [AD.4].

2 CONCEPTS

2.1. OBJECTIVE

Software should have a level of trustworthiness commensurate to the purpose for which it is used [AD.1].

2.2. SOURCES OF SOFTWARE

Software can arise in a number of different manners, with the major categories being:

- Open Source
- Off The Shelf (OTS)
- Custom production ("Bespoke") for particular purpose
- Modification of Open Source, OTS or Custom

2.3. SUPPLY CHAIN

It is likely that, irrespective of the notional category of software in use, elements will have been reused from other sources, as illustrated in Figure 2-1.

Segment	Embedded Systems	SCADA Systems	Communications Systems	IT Infrastructure	IT Applications
Reuse	Limited	Libraries	Libraries; Mobile Code	Libraries; Mobile Code; Cloud Services	Libraries; Mobile Code; Cloud Services; Mashups

Figure 2-1

An implication of this Supply Chain is that irregularities may arise not only in those elements of software produced for a particular purpose, but also from underlying components that have been reused from anywhere in the extended supply chain.

2.4. TYPES OF IRREGULARITY

No software, other than those of a trivially small nature, can be proven, or even be expected, to be completely free of irregularities, comprising of either a Deviation or a Defect:

A deviation can be classed as:

- an unexpected outcome during run-time that is not caused by a coding error or mistake; and/or
- non-conformity with an explicit/implicit software requirement, due to misinterpretation (or otherwise) of the System Requirement Specification (SRS).

A defect can be classed as:

- a coding error/mistake; and/or
- non-fulfilment of an explicit/implicit software requirement.

It is the latter category, Defects, that are typically of most concern.

2.5. DEFECTS IN THE LIFECYCLE

The typical lifecycle of a piece of software is illustrated by Figure 2, which illustrates the relationship between Trustworthy Software Guidance, and that provided for the specific facet(s) of trustworthiness later in the lifecycle.

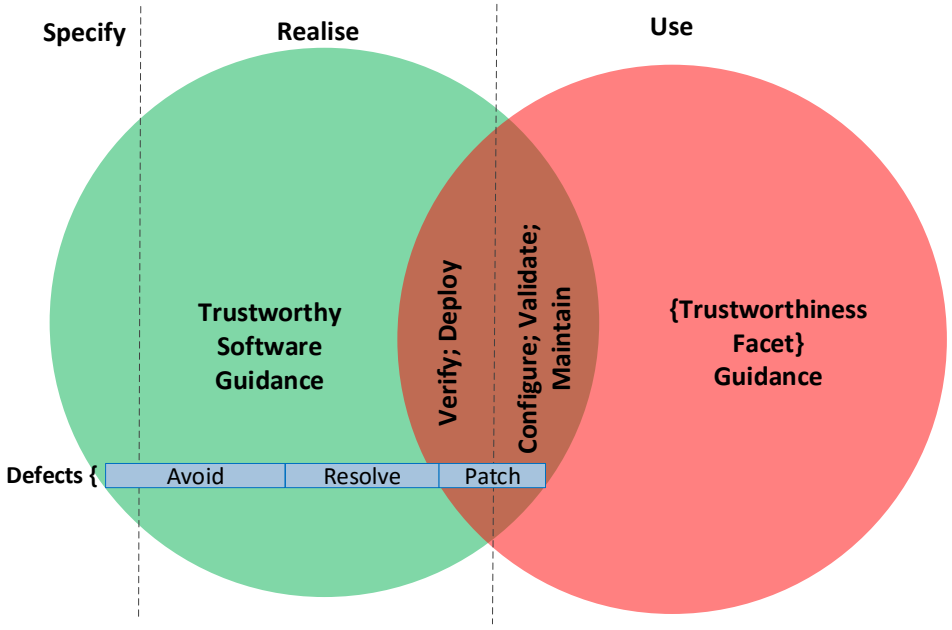

Figure 2-2

Defects in software are generally characterised as being one of three types, which are treated differently during the lifecycle, as illustrated in Figure 2.

- Weaknesses are generic classes of potential deficiency in software, such as buffer overflows, which should be Avoided.
- Vulnerabilities can be:

  i. the existence of a generic Weakness in a particular platform, such as a buffer overflow occurring in a specific operating system or application;

  ii. interactions between multiple software elements that bypass intended controls;

  iii. accidental actions of software developers that result in defects and errors;

  iv. deliberate actions of software developers that bypass intended controls, such as trap doors that permit unauthorised access to the system.

Vulnerabilities should ideally have been Avoided by identification of a potential Weakness, but where they occur and are detected during the realisation phase they should be Resolved.

- Susceptibilities are the confirmed presence of one or more Vulnerability within an implemented system, such as the presence of an operating system with a buffer overflow defect. Susceptibilities in systems stem from:

    i. initial implementation;

    ii. changes to software, such as from adding new facilities;

    ii. use of utility programs, which may be capable of circumventing security measures in the controlling or application software.

  Susceptibilities should ideally have been Avoided by identification of a potential Weakness, or Resolved if detected as a Vulnerability before deployment, but where encountered during Use give rise to the retrospective application of fixes, referred to hereafter as "Patching".

2.6. TYPES OF USE

Software may be encountered in a large and diverse number of contexts, which can be summarised into three primary digital technology realms:

- "IT": Information Technologies, which are the most obvious direct descendants of software's origins within the computing realm
- "OT": Operational Technologies, which is the use of software in a plethora of environments from railway signalling to automotive automation to industrial control systems (ICS)
- "CT": Consumer Technologies, which range from those closely related to the classical computing realm (e.g. handheld "tablets" and smartphones) to the presence of embedded software "white goods" (ranging from internet connected fridges to the programmable controllers but non-connected controllers used for washing machines)

These realms are collectively referred to as "IOCT".

2.7. PATCHING SCENARIOS

A consideration of properties of these IOCT contexts lead to a categorisation of Patching Scenarios:

- Unpatchable (UP): software that is either completely and/or effectively unable to be patched on a routine basis, as either it is stored in non-rewritable media, or there is no provision whilst in situ to apply patches (e.g. the technology would have to be removed and dismantled before any writable media could be accessed). This scenario tends to be confined to some types of OT and CT.
- Direct Patching (DP): the software can be patched, but only by gaining direct physical access to the technology (e.g. by plugging into an On-Board Diagnostics (OBD) port on a motor vehicle[1]). This scenario can apply to all of the IOCT realm
- Online Patching (OP): the software can be patched remotely by authorised persons, provided that a network connection exists. This scenario can apply to all of the IOCT realm
- Automatic patching (AP): the software supports patching without any human intervention, provided that a network connection exists. This scenario can apply to all of the IOCT realm

[1] The motor vehicle scenario has become blurred in recent times, with certain suppliers now providing remote access to OBD facilities by wireless technologies, potentially allowing OP or AP scenarios

2.8. TRUSTWORTHY SOFTWARE FRAMEWORK (TSF)

The Trustworthy Software Framework (TSF) provides a domain- and implementation-agnostic collection of consensus good practice that is intended to be applicable to all risk environments across the IOCT realm, and is codified in two layers:

- The Baseline view, applicable to the lower levels[2] of trustworthiness, is laid down in the Trustworthy Software Essentials (TSE) [AD.2], which contains the most widely applicable subset of the TSF control set
- The Comprehensive view, applicable to the higher levels of trustworthiness, is laid down in a formal British Standards Institution (BSI) specification (PAS754:2014) [AD.1], which contains the full version of the TSF control set

The TSF controls relevant to Patching are summarised in Table 2-3.

Level 1-C : Control Area (Concepts)	Control Serial	Level 2-C: Control Group (Principles)	Level 3-C: Control Summary (Techniques)	L3-B (TSE)	Level3-C: Control Detail (Techniques)
Governance (Gv)	GV.01.50	Understand General Environment	Understand Supply Chain		Understand Whole Supply Chain
	GV.03.40	Implement formal management regime	Trustworthy Software Release Authority (TSRA)	✓	Implement the role of Trustworthy Software Release Authority (TSRA)
Risk (Ri)	RI.01.30	Understand General Risks	Adversity Analysis		Perform Adversity (Hazard + Threat) Analysis
	RI.01.40		Vulnerability Analysis		Perform Vulnerability Analysis of both bespoke and off the shelf elements
	RI.02.20	Understand Trustworthiness Risks	Understand Weaknesses		Maintaining understanding of current and emergent weaknesses
	RI.02.30		Attack patterns and malware		Maintaining understanding of current and emergent attack patterns and malware
	RI.02.40		Understand Vulnerabilities		Maintaining understanding of current and emergent vulnerabilities

[2] The Trustworthiness Levels are defined in both PAS754:2015 [AD.1] and the "Guide to Risk and TL Assessment" [AD.3]

Level 1-C : Control Area (Concepts)	Control Serial	Level 2-C: Control Group (Principles)	Level 3-C: Control Summary (Techniques)	L3-B (TSE)	Level3-C: Control Detail (Techniques)
Controls - Procedural (Cn-Pr)	PR.02.10	Perform Supplier Management	Supply Chain identification		Identify Supply Chain
	PR.04.20	Maintain Configuration Management	Realisation Configuration Management (CM)	✓	Implement and maintain Configuration Management (CM) of Realisation, including artefact integrity and version control
	PR.04.40		Product Release	✓	Formal process for Product Release
	PR.04.60		Trustworthy Software Constraint and Dependency Model (TSCDM)	✓	Implement and maintain a Trustworthy Software Constraint and Dependency Model (TSCDM)
	PR.04.70		Installation Configuration Management (CM)	✓	Implement and maintain Configuration Management (CM) for installed software
	PR.05.20	Confirmation of Assurance	Assurance Review		Establish a process for Assurance and Acceptance Review
	PR.06.10	Perform Trusted Software Asset Management	Trusted delivery		Implement trusted means of software delivery
	PR.06.20		Output Formal Acceptance		Implement an Acceptance process for products and services
	PR.06.40		Software assets review	✓	Software assets should be reviewed regularly
	PR.07.10	Maintain Defect Management	Realisation Defect Management	✓	Ensure all Defects identified during Realisation are recorded, reported and assessed, with rectification at earliest opportunity using process for monitoring through a Realisation Trustworthy Software Defect and Deviation List (R-TSDDL)

Level 1-C : Control Area (Concepts)	Control Serial	Level 2-C: Control Group (Principles)	Level 3-C: Control Summary (Techniques)	L3-B (TSE)	Level3-C: Control Detail (Techniques)
	PR.07.20		In-service Defect Management	✓	Ensure all In Service Defects and Deviations are recorded in an In-service Trustworthy Software Defect and Deviation List (I-TSDDL), reported and assessed, with rectification at earliest opportunity using process for monitoring of deferrals
Controls - Technical (Cn-Te)	TE.03.30	Follow Structured Design	Proven components	✓	When re-using components and libraries, these should have a Trustworthy Software provenance wherever possible
	TE.03.40		Open Source handling		Procedures for handling of Open Source components and libraries
	TE.05.45	Seek Trustworthy Realisation	Mitigate identified failure modes	✓	Ensure mitigations are used for all identified failure modes
	TE.05.65		Control malicious code	✓	Implement measures to control malicious code
	TE.09.50	Perform Internal Pre-release Review	Trustworthy Software Release Notice (TSRN)	✓	Issue formal Trustworthy Software Release Notice (TSRN)
	TE.11.10	Enable Dependable Deployment	Source code persistence	✓	Ensure source code can be obtained throughout lifecycle, such as by Escrow service
	TE.11.60		Continual Remediation	✓	Updating and Patching of implemented software, with routine, critical and emergency options, taking due cognisance of R-TSDDL / I-TSDDL
	TE.11.70		Continual Vigilance		Monitor for anomalies and trends throughout lifecycle, including intrusions

Level 1-C : Control Area (Concepts)	Control Serial	Level 2-C: Control Group (Principles)	Level 3-C: Control Summary (Techniques)	L3-B (TSE)	Level3-C: Control Detail (Techniques)
Compliance (Cm)	CM.02.50	Perform Acceptance Verification (VER)	Maintain Metrics		Maintain Metrics, including progress against I-TSDDL deferrals

Table 2-3

2.9. AUDIENCES FOR PATCHING GUIDANCE

When used as a stand alone document for organizations with no current approach to software trustworthiness, this guidance will facilitate the good patching practices for software in its many guises from embedded equipment through consumer devices to industrial control systems.

For organizations that already address software trustworthiness though the lens of one or more of the five main facets of trustworthiness that typically operate in isolation (safety, reliability, availability, resilience and security), this guidance provides a companion and complement to other relevant publications [e.g. RD.1 – RD.5], and reviewing the guidance in this document alongside practices derived from individual facets could allow the identification of gaps and enhancements.

The guidance is applicable across all 6 groups of practitioners whom have a stake in Patching:

- Research – those who discover vulnerabilities
- Produce – those who realise software
- Assure – those who verify software
- Configure – those who set up software during deployment
- Operate – those who maintain and/or use software in service
- Respond – those who deal with Events relating to software

3 DEFECT LIFECYCLE

3.1. GENERIC LIFECYCLE MODELS

There are *de facto* standardised models relating to System Lifecycle Processes [RD.6] and Software Lifecycle Processes [RD.7], and TSI typically advocates the use of the simplified 3 Stage "Specify-Realise-Use" model, as shown in Figure 2.2.

3.2. DEFECT LIFECYCLE MODEL

In the case of Patching, however, a distinct 4 Phase lifecycle model (Disclosure-Monitor-Deploy-Review) is more relevant, as illustrated in Figure 3-1.

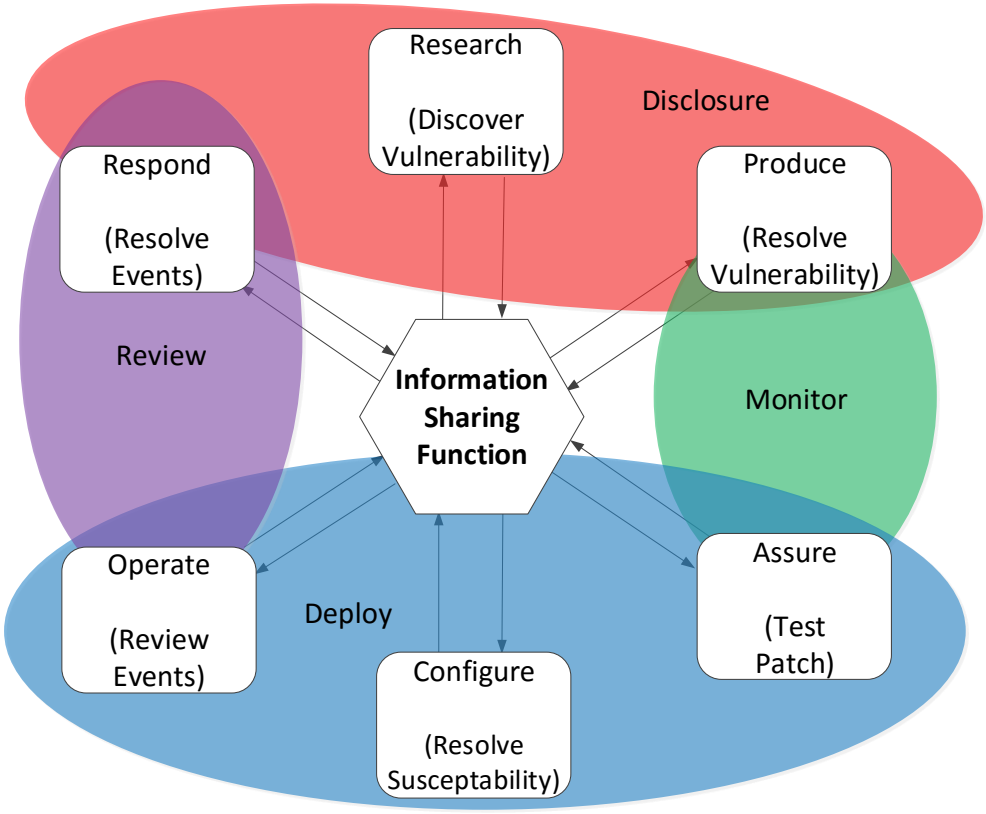

Figure 3-1

These 4 Phases are described in more detail in the following Sections.

4 DISCLOSURE

4.1. ELEMENTS OF DISCLOSURE

The Disclosure Phase can be considered as consisting of 3 Sub-phases, as illustrated at Figure 4-1.

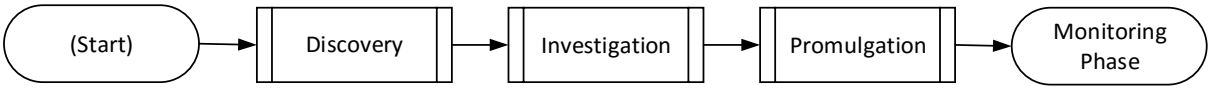

Figure 4-1

4.2. DISCOVERY

Those who discover Weaknesses and/or Vulnerabilities are generically referred to as "Researchers", but this does not necessarily imply persons with a Research role. Weaknesses and/or Vulnerabilities may be discovered by:

- Those who Produce software
- Those who Configure software
- Those who Operate software
- Those who Respond to software Events
- Academics
- Independent specialists with benign intent, including "White Hat" hackers
- Independent specialists with malign intent, including "Black Hat" hackers
- Malign organisations, including Organised Crime (OC), Empowered Small Agents (ESA) and Nation State actors

Once discovered, there are 3 primary ways in which Researchers outside of Producer organisations may act:

- Responsible Disclosure, whereby the information is passed to the Producer organisation, and potentially select other benign interested parties, on a strict need to know basis to allow the Producer organisation to manage the Investigation and Promulgation Phases.
- Non-disclosure, whereby the discovery is not released publicly. In some case this may be the practice from Producer organisations who are unable and/or unwilling to perform remediation, but this approach is also naturally popular with those of malign intent who may wish to use Weaknesses and/or Vulnerabilities for their own ends without providing any prior warning
- Full Disclosure is where there full details, including potentially means of exploitation, are put into the public domain. In cases where Weaknesses and/or Vulnerabilities are potentially exploitable by those of malign intent, this approach raises the risk that such exploitation may occur before any remediation has been implemented

Although the consensus amongst most Producers and other benign interested parties is that Responsible Disclosure is preferable, and many organisations have published guidance in this respect, as of yet attempts

to codify such a Responsible Disclosure approach through one of the main *de facto* or *de jure* Standards Development Organisation (SDO) have come to nothing, the most obvious example being an Internet Engineering Task Force (IETF) Draft which time expired.

There are, however, *de facto* standardised approaches to formatting information for exchange, using the Common Weakness Enumeration (CWE) [RD.8] and Common Vulnerabilities and Exposures (CVE) [RD.9].

4.3. INVESTIGATION

Once discovered, the producer organisation investigates the Weakness and/or Vulnerability, and attempts to remediate the problem(s).

The remediation may take several forms:

- As a dedicated "patch", being a partial, incremental replacement of existing software solely to deal with one set of issues
- As an unplanned, additional minor update[3] which fully replaces existing software, and may be solely related to a single remediation and/or include multiple remediations
- As part of a planned minor[4] or major[5] update which fully replaces existing software, which will include multiple remediations and other changes

There are *de facto* standardised approaches to this Phase of Disclosure [RD.10, RD-11]

4.4. PROMULGATION

Once a remediation has been found and tested, the producer organisation will inform those affected.

There are *de facto* standardised approaches to this Phase of Disclosure [RD.10]

4.5. RESPONSIBILITIES

The work relating to the final Promulgation of information by a Producer organisation to interested parties should be sanctioned explicitly by their Trustworthy Software Release Authority (TSRA).

4.6. RECORDING REQUIREMENTS

The following TSF recommended documentation will need to be amended accordingly:

- Realisation Trustworthy Software Defect and Deviation List (R-TSDDL)
- Trustworthy Software Constraint and Dependency Model (TSCDM)
- Trustworthy Software Release Notice (TSRN)

4.7. TSF APPLICABILITY

The TSF controls affected by the Disclosure phase are shown in Table 4-2.

[3] Typically reflected as the 3rd element of the Version Number, for instance the "Z" of Version X.Y.Z
[4] Typically reflected as the 2nd element of the Version Number, for instance the "Y" of Version X.Y
[5] Typically reflected as the 1st element of the Version Number, for instance the "X" of Version X.Y

Control Serial	L3-C: Control Summary (Techniques)	Safety (Sf)	Reliability (Rl)	Availability (Av)	Resilience (Rs)	Security (Sc)
GV.03.40	Trustworthy Software Release Authority (TSRA)	✓	✓	✓	✓	✓
RI.01.30	Adversity Analysis	✓	✓	✓	✓	✓
RI.01.40	Vulnerability Analysis	✓	✓	✓	✓	✓
RI.02.20	Understand Weaknesses	✓	✓	✓	✓	✓
RI.02.30	Understand Attack patterns and malware		✓	✓	✓	✓
RI.02.40	Understand Vulnerabilities	✓	✓	✓	✓	✓
PR.04.20	Realisation Configuration Management (CM)	✓	✓	✓	✓	✓
PR.04.40	Product Release	✓	✓	✓	✓	✓
PR.04.60	Trustworthy Software Constraint and Dependency Model (TSCDM)	✓	✓	✓	✓	✓
PR.06.10	Trusted delivery	✓	✓	✓	✓	✓
PR.07.10	Realisation Defect Management	✓	✓	✓	✓	✓
TE.03.30	Proven components	✓	✓	✓	✓	✓
TE.03.40	Open Source handling	✓	✓	✓	✓	✓
TE.05.45	Mitigate identified failure modes	✓	✓	✓	✓	✓
TE.05.65	Control malicious code	✓	✓	✓	✓	✓
TE.09.50	Trustworthy Software Release Notice (TSRN)	✓	✓	✓	✓	✓
TE.11.10	Source code persistence	✓	✓	✓	✓	✓
TE.11.60	Continual Remediation	✓	✓	✓	✓	✓
TE.11.70	Continual Vigilance	✓	✓	✓	✓	✓

Table 4-2

5 MONITOR

5.1. ELEMENTS OF MONITORING

The Monitor Phase can be considered as consisting of 3 Sub-phases, as illustrated at Figure 5-1.

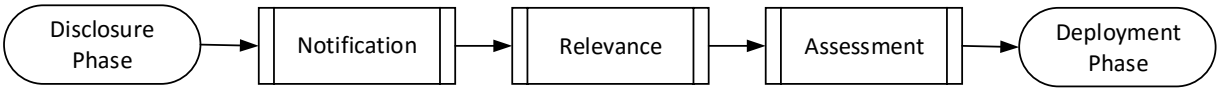

Figure 5-1

5.2. NOTIFICATION

In order to understand the emergent Weaknesses and/or Vulnerabilities that may affect an organisation, the Monitoring process needs to anchor to relevant sources of Notification, be they directly from Producer organisations – and their Product Security Incident Response Team (PSIRT) in particular where they exist, and/or through appropriate specialist organisations such as Computer Security Incident Response Teams (CSIRT)[6], and/or from public domain feeds.

There are *de facto* standardised approaches to formatting information for exchange, using the Common Weakness Enumeration (CWE) [RD.8] and Common Vulnerabilities and Exposures (CVE) [RD.9], but these are not universally followed.

A summary of well known PSIRTs, CSIRTs and Newsfeeds is provided at Annex B.

5.3. RELEVANCE

The relevance of Notifications will initially depend on the nature the organisation:

- Those producing software will be interested in both Weaknesses that could apply to their software, and Vulnerabilities that could apply to externally sourced software components.
- End user organisations will only be interested in Vulnerabilities that could apply to externally sourced software components

A maintained Software Asset Register, including detailed version information, is fundamental to being able to Triage incoming Notifications for relevance to the organisation. There are *de facto* standardised approaches to describing platforms [RD.12]

5.4. ASSESSMENT

Provided that a Notification is Relevant, it is then necessary to carry out a risk-based assessment of how Important and Urgent any action may be.

A generic model of a risk-based approach is provided in Figure 5-2.

[6] The term Computer Emergency Response Team (CERT) is often used for this function, but is deprecated due to trademark issues

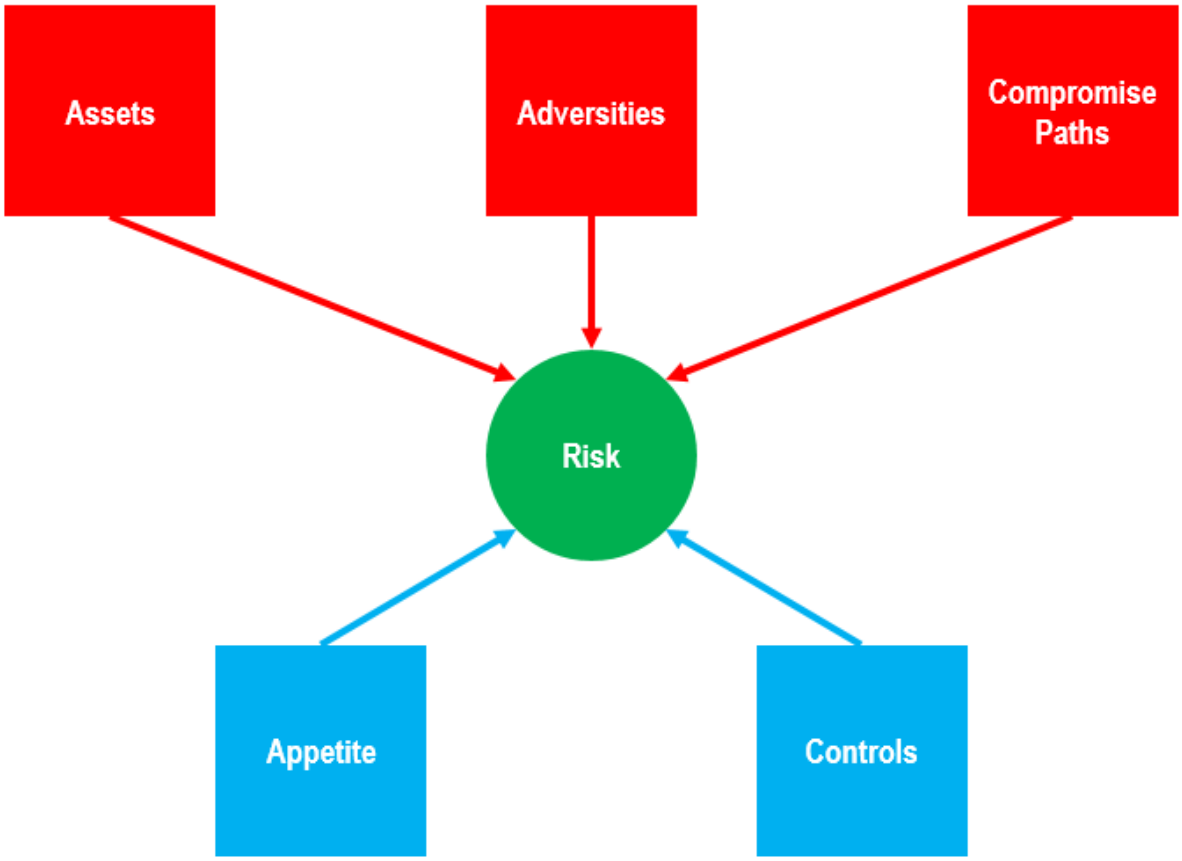

Figure 5-2

A decision as to the Importance and Urgency of any particular patch is a function of two main sets of considerations:

- The generic issues presented by the potential problem, which are often characterised as being Routine, Critical or Emergency
- The particular issues relating the potential problem to the organisational usage

There are *de facto* standardised approaches to this Assessment Phase of Monitoring, with probably the best known being a methodology that takes a Security lens to the challenge, the Common Vulnerability Scoring System (CVSS) [RD.13]. It is predominantly orientated toward IT systems (and certain types of CT systems), and will may therefore not be directly applicable to other types of CT systems, or to OT systems.

In addition, underlying assumptions within the CVSS approach do not necessarily easily map to scenarios where:

- A defect will manifest itself in a time-dependent and/or random manner, without any need for network level or direct interaction with the affected system
- The issue is a deviation rather than defect
- The impacts are not of a Security nature

5.5. RECORDING REQUIREMENT

The following TSF recommended documentation will need to be amended accordingly:

- In-service Trustworthy Software Defect and Deviation List (I-TSDDL)

5.6. TSF APPLICABILITY

The TSF controls affected by the Disclosure phase are shown in Table 5-3.

Control Serial	L3-C: Control Summary (Techniques)	Safety (Sf)	Reliability (Rl)	Availability (Av)	Resilience (Rs)	Security (Sc)
GV.01.50	Understand Supply Chain	✓	✓	✓	✓	✓
RI.02.40	Understand Vulnerabilities	✓	✓	✓	✓	✓
PR.06.40	Software assets review	✓	✓	✓	✓	✓
PR.07.20	In-service Defect Management	✓	✓	✓	✓	✓
TE.09.50	Trustworthy Software Release Notice (TSRN)	✓	✓	✓	✓	✓

Table 5-3

6 DEPLOYMENT

6.1. ELEMENTS OF DEPLOYMENT

The Deployment Phase can be considered as consisting of 3 Sub-phases, as illustrated at Figure 6-1.

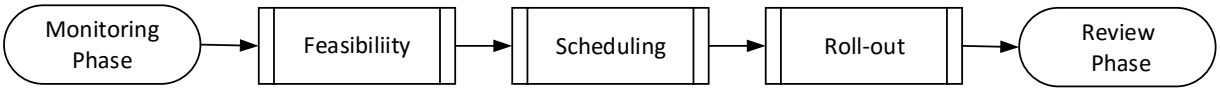

Figure 6-1

A variety of other publications [e.g. RD.1 – RD.5] provide specific and/or domain specific guidance on this Phase, but reviewing the guidance in this document alongside other such practices could allow the identification of gaps and enhancements.

6.2. FEASIBILITY

Once a patch has been Assessed for Importance and Urgency, the Patching Scenarios for the affected software will drive Feasibility of Deployment, as UP software cannot be realistically remediated.

If instances of affected UP software are identified, then a review will be required as to alternative means of mitigation, which in the most extreme cases may require the software to be removed from use.

6.3. SCHEDULING

In the other Patching Scenarios (DP/OP/AP), the next consideration is Scheduling of Deployment.

The ability to Schedule Deployment will need to consider trade-offs and potentially Confounding Factors, including:

- Mission criticality of the service being provided, which can impact in two distinct manners:
    - A confounding factor of restriction on slots provided for outages[7], if the application of a patch cannot be performed without causing an interruption to service
    - The need to trade-off concerns over potential deleterious collateral effects (either direct, or on other elements of the system or systems to which it is connected) of the application of patches[8]: for more mission-critical scenarios this may require offline "regression testing" against a representative configuration to ensure there are no such deleterious collateral effects
- Nature of network connectivity (for OP/AP), including bandwidth and time limitations, or physical accessibility (for DP)
- Ability of the organisation to control its own updates, including any dependence on 3rd parties[9]

[7] This is particularly common with OT systems

[8] Anecdotal evidence suggests that an increasingly small percentage of patches in mainstream IT/CT environments are liable to cause such problems, but neither perfection nor absence of unexpected collateral effects from rare configurations can be assumed

[9] A particular issue in the IT/CT realms is the presence of intermediaries, such as Mobile Network Operators (MNO) or Managed Service Providers (MSP), who may block the ability of end users to directly obtain and/or apply patches that may be available from the producer organisation(s)

- The organisational authorisation processes for deployment of patches

These considerations are obviously not unique to a patch deployment, as they will also apply to other forms of update, so an aligned approach to achieving "current maintained version" in both instances is to be commended.

6.4. ROLL OUT

In the absence of Confounding Factors, it can be assumed that those with only an Implicit Need for Trustworthiness (i.e. TL 1/2) can probably be more relaxed in seeking patch roll out opportunities than those with an Explicit Need for Trustworthiness (i.e.TL3/4), as the likely impact should the defect and/or deviation manifest itself is likely to be less.

Perversely, however, market and technology asymmetries mean that many IT/CT systems in the TL 1/2 risk cases are more likely to be configured to require less organisational involvement in patching (i.e. AP for much OTS software).

Nonetheless the recommended approach remains that opportunities should be sought to achieve current maintained version (for both patches, and other updates) at the earliest appropriate juncture, as there are a litany of examples across many years, in particular in respect of security exploits, which demonstrates that the longer the delay in achieving current maintained version, the more likely a problem from the defect and/or deviation is to manifest itself.

The exact timing of software deployments will vary with organisation and time, but an indicative approach to the various ways of achieving current maintained version is provided in Table 6-2.

<table>
<tr><th rowspan="2">Deployment Model</th><th colspan="3">Update Characteristic</th></tr>
<tr><th>Routine</th><th>Critical</th><th>Emergency</th></tr>
<tr><td>Patch
Additional unplanned update[11]</td><td>Balance against Trade-offs and Confounding Factors, but aim to achieve within short period[10]</td><td>Urgent deployment will typically dominate Trade-offs and Confounding Factors</td><td>Immediate deployment should dominate Trade-offs and Confounding Factors</td></tr>
<tr><td>Planned minor update[12]</td><td>Allow initial period[13] for problems to identified and resolved, then plan deployment a.s.a.p.</td><td colspan="2" rowspan="2">Not applicable</td></tr>
<tr><td>Planned major update[14]</td><td>Allow reasonable period[15] for problems to identified and resolved, then plan deployment a.s.a.p.</td></tr>
</table>

Table 6-2

[10] Typically no greater than 1 month for OP/AP deployments, and 3 months for DP deployments
[11] Typically reflected as the 3rd element of the Version Number, for instance the "Z" of Version X.Y.Z
[12] Typically reflected as the 2nd element of the Version Number, for instance the "Y" of Version X.Y
[13] Typically no less than 1 month for "IT" deployments
[14] Typically reflected as the 1st element of the Version Number, for instance the "X" of Version X.Y
[15] Typically no less than 1 quarter for "IT" deployments

6.5. RECORDING REQUIREMENT

The following TSF recommended documentation will need to be amended accordingly:

- In-service Trustworthy Software Defect and Deviation List (I-TSDDL)

6.6. TSF APPLICABILITY

The TSF controls affected by the Deployment phase are shown in Table 6-3.

Control Serial	L3-C: Control Summary (Techniques)	Safety (Sf)	Reliability (Rl)	Availability (Av)	Resilience (Rs)	Security (Sc)
PR.04.70	Installation Configuration Management (CM)	✓	✓	✓	✓	✓
PR.06.10	Trusted delivery	✓	✓	✓	✓	✓
PR.06.20	Output Formal Acceptance	✓	✓	✓	✓	✓
PR.07.20	In-service Defect Management	✓	✓	✓	✓	✓
TE.11.60	Continual Remediation	✓	✓	✓	✓	✓

Table 6-3

7 REVIEW

7.1. ELEMENTS OF REVIEW

The Review Phase can be considered as consisting of 3 Sub-phases, as illustrated at Figure 7-1.

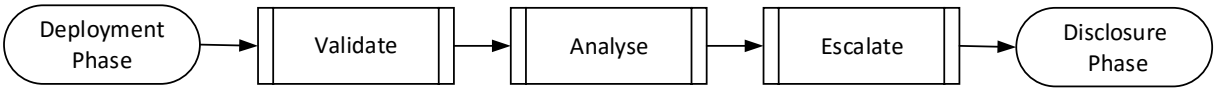

Figure 7-1

7.2. VALIDATE

Once a patch has been Rolled Out, it is prudent to validate the situation:

- By spot checking a number of installations to ensure the correct version(s) have been applied
- By using such spot checks to ensure that 100% of the targeted sample have had patches applied

7.3. ANALYSE

Once in service, system events should be monitored for any potentially deleterious effects arising from the application of patches:

- Any collateral effects that may have arisen from the specific implementation configuration, which either may not have been detected during regression testing, or where no regression testing was performed
- Any new weakness and/or vulnerability that may be revealed subsequent to the installation of the patch

7.4. ESCALATE

If the Analysis sub-phase identifies any problem:

- Any collateral effects will need to be appropriately escalated and treated internally or with the relevant 3rd party MSP
- Any new weakness and/or vulnerability will need to appropriately escalated to the producer organisation

7.5. RECORDING REQUIREMENT

The following TSF recommended documentation will need to be amended accordingly:

- In-service Trustworthy Software Defect and Deviation List (I-TSDDL)

7.6. TSF APPLICABILITY

The TSF controls affected by the Review phase are shown in Table 7-2.

Control Serial	L3-C: Control Summary (Techniques)	Safety (Sf)	Reliability (Rl)	Availability (Av)	Resilience (Rs)	Security (Sc)
PR.04.70	Installation Configuration Management (CM)	✓	✓	✓	✓	✓
PR.05.20	Assurance Review	✓	✓	✓	✓	✓
PR.07.20	In-service Defect Management	✓	✓	✓	✓	✓
TE.11.70	Continual Vigilance	✓	✓	✓	✓	✓
CM.02.50	Maintain Metrics	✓	✓	✓	✓	✓

Table 7-2

8 EVOLUTION

8.1. STATUS/PLAN

This document will be updated on a periodic basis to reflect the constantly evolving nature of the cyber ecosystem and therefore, the need for trustworthiness.

In particular, work is noted to be ongoing on the following areas:

- Disclosure approaches, with a European Network and Information Security Agency (ENISA) project being commenced in 2015
- The use of Risk Based Scoring Systems, to deal with concerns beyond the security-facet centric nature of most current approaches, with TSI being in dialogue with various stakeholders on this topic

8.2. MAINTENANCE/CONTRIBUTIONS

TSI welcomes input from Stakeholders on all aspects of its activity, including any additions or amendments to the Trustworthy Software Library.

If you have any suggestions please feel free to engage with TSI, either through your existing contact, or by emailing enquiries@uk-tsi.org.

ANNEX A: BIBLIOGRAPHY

A.1 ASSOCIATED DOCUMENTS

TSI documents which provide additional information that may be relevant to some or all of the readership.

[AD.1] PAS 754:2014 “Software Trustworthiness. Governance and management. Specification”

[AD.2] TS502-0 “Baseline Framework Specification” ("TS Essentials")

[AD.3] TS507-1 “Guide to Risk and TL Assessment”

[AD.4] TS543-6-01 “Patching Summary”

A.2 REFERENCE DOCUMENTS

Documents from other organisations that may be relevant to some or all of the readership.

[RD.1] “Good Practice Guide – Patch Management”, UK Centre for Protection of National Infrastructure (CPNI), 24 October 2006

[RD.2] “Recommended Practice for Patch Management of Control Systems”, US Department of Homeland Security (DHS) National Cyber Security Division (NCSD), December 2008

[RD.3] SP800-40 “Guide to Enterprise Patch Management Technologies”, US National Institute of Standards and Technology (NIST), Revision 3, July 2013

[RD.4] “Cyber Essentials Scheme”, HM Government (HMG), April 2014

[RD.5] “TR62443-2-3 - Security for Industrial Automation and Control Systems - Patch management in the IACS Environment”, International Electrotechnical Commission (IEC), *due for publication* July 2015

[RD.6] “15288:2015 -- Systems and software engineering -- System life cycle processes”, International Standards Organisation (ISO) / International Electrotechnical Commission (IEC), May 2015

[RD.7] “12207:2008 -- Systems and software engineering -- Software life cycle processes”, ISO/IEC, August 2008

[RD.8] “ X.1524 – Cybersecurity information exchange – Vulnerability/state exchange – Common weakness enumeration”, International Telecommunications Union (ITU-T), 1st Edition, March 2012

[RD.9] “ X.1520 – Cybersecurity information exchange – Vulnerability/state exchange – Common vulnerabilities and exposures”, ITU-T, 2nd Edition, January 2014

[RD.10] “29147:2014 -- Information technology -- Security techniques -- Vulnerability disclosure”, ISO/IEC, February 2014

[RD.11] “ 30111:2013 -- Information technology -- Security techniques -- Vulnerability handling processes”, ISO/IEC, October 2013

[RD.12] “ X.1528 – Cybersecurity information exchange – Vulnerability/state exchange – Common platform enumeration”, ITU-T, 1st Edition, September 2012

[RD.13] “ X.1521 – Cybersecurity information exchange – Vulnerability/state exchange – Common vulnerability scoring system”, ITU-T, 1st Edition, April 2011

ANNEX B: INFORMATION SOURCES

In order to understand the emergent Weaknesses and/or Vulnerabilities that may affect an organisation, the Monitoring process needs to anchor to relevant sources of Notification.

References we make to any specific product, process or service by trade name, trademark manufacturer, or otherwise, or references to websites or material are not endorsements or recommendations.

B.1 PRODUCER ORGANISATIONS

Many Producer organisations publish Notifications, often from what is known as Product Security Incident Response Team (PSIRT) function. A non-exhaustive list of Producers performing this function that are relevant to Mass Market IT and CT usage is provided below:

- Apple (https://www.apple.com/uk/support/security/)
- Cisco (http://tools.cisco.com/security/center/publicationListing.x)
- Dell (http://www.secureworks.com/resources/blog/category/counter-threat-unit-research/)
- IBM (http://www.ibm.com/developerworks/java/jdk/alerts/)
- Microsoft (https://technet.microsoft.com/en-us/library/security/dn610807.aspx)
- Oracle (http://www.oracle.com/technetwork/topics/security/alerts-086861.html)
- VMWare (https://www.vmware.com/uk/security/advisories)

B.2 CSIRTS

A number of national-level Computer Security Incident Response Teams (CSIRT)[16] undertake a Disclosure Management function in conjunction with, or sometimes in place of, producer organisations, as shown in the non-exhaustive list below:

- CERT-UK (http://www.cert.gov.uk)
- CERT/CC (http://www.cert.org)
- ICS-CERT (http://ics-cert.us-cert.gov)
- JP-CERT (http://www.jpcert.or.jp/)
- US-CERT (http://www.us-cert.gov)

B.1 NEWSFEEDS

The following list represent some of the best known public domain news feeds, but is by no means an exhaustive list:

- Bugtraq (http://seclists.org/bugtraq/)
- Secunia (https://secunia.com/community/advisories/)

[16] The term Computer Emergency Response Team (CERT) is often used for this function, but is deprecate due to trademark issues

ANNEX C: ABBREVIATIONS

AD	Associated Document
AP	Automatically Patchable
CERT	Computer Emergency Response Team
CIA	Confidentiality, Integrity, Availability
CM	Configuration Management
CPE	Common Platform Enumeration
CSIRT	Computer Security Incident Response Team
CT	Consumer Technology
CVE	Common Vulnerabilities and Exposures
CVSS	Common Vulnerability Scoring System
CWE	Common Weakness Enumeration
DP	Directly Patchable
ENISA	European Network and Information Security Agency
ESA	Empowered Small Agent
I-TSDDL	In-Service Trustworthy Software Defect and Deviation List
ICS	Industrial Control System
IOCT	Information, Operation and Consumer Technologies
IT	Information Technology
M/E	Mass Market with Explicit Need
M/I	Mass Market with Implicit Need
MSP	Managed Service Provider
N/E	Niche with Explicit Need
OC	Organised Crime
OGL	Open Government Licence
OP	Online Patchable
OTS	Off the Shelf
PAS	Publicly Available Specification
PSIRT	Product Security Incident Response Team
R-TSDDL	Realisation Trustworthy Software Defect and Deviation List
RD	Reference Document
SCADA	Supervisory, Control And Data Acquisition
SDO	Standards Development Organisation

TL	Trustworthiness Level
TSCDM	Trustworthy Software Constraint and Dependency Model
TSDDL	Trustworthy Software Defect and Deviation List
TSE	Trustworthy Software Essentials
TSF	Trustworthy Software Framework
TSI	Trustworthy Software Initiative
TSL	Trustworthy Software Library
TSMS	Trustworthy Software Management System
TSRA	Trustworthy Software Release Authority
TSRN	Trustworthy Software Release Note
UP	Unpatchable

ANNEX D: GLOSSARY

For purposes of this document, the following terms and definitions apply:

DEVIATION	A deviation can be classed as: ▪ an unexpected outcome during run-time that is not caused by a coding error or mistake; and/or ▪ non-conformity with an explicit/implicit software requirement, due to misinterpretation (or otherwise) of the SRS.
DEFECT	A defect can be classed as: ▪ a coding error/mistake; and/or ▪ non-fulfilment of an explicit/implicit software requirement.
SAFETY (FACET)	The ability of the system to operate without harmful states.
RELIABILITY (FACET)	The ability of the system to deliver services as specified.
AVAILABILITY (FACET)	The ability of the system to deliver services when requested.
RESILIENCE (FACET)	The ability of the system to transform, renew and recover in timely response to events.
SECURITY (FACET)	The ability of the system to remain protected against accidental or deliberate attacks.