Security Problem Definition

1. Compliant Targets of Evaluation

Software applications to be considered for evaluation under this PP can be categorized under the following broad categories:

1. Enterprise Server Applications
2. Enterprise Desktop Applications
3. Enterprise-grade Mobile Applications

In addition to the above categories there are large number of applications (Desktop and Mobile) that fall under "Consumer-grade" category. While such applications could be evaluated under the Software Application cPP, it is not the intention of this iTC to create requirement specific to this category. The iTC doesn't believe the consumer grade app ecosystem would support the cost and timelines associated with a typical Common Criteria evaluation.

One more way (and perhaps a more useful way in the context of creating SFRs) to categorize apps is based on type of installation/deployment. These categories are as follows:

1. Traditional software running on an execution environment, e.g: Enterprise agent applications/sensors
2. Software appliance type of applications. E.g.:
    a. Enterprise management application
    b. Software defined network appliances
3. Distributed applications (e.g. enterprise resource planning systems)
4. Web apps
5. Virtualized applications (e.g. running on a VM or in container)

Out of the above, this first iteration of the cPP will aim to cover categories 1, 2a, and 3. Software define network appliances are being covered by the Network iTC. Web apps are very different in terms of how they are built and operate. Their threat model is also quite different, and is not addressed in this cPP at this point. At the time of writing the SPD, it was also decided to not cover virtualized applications due to the slightly different threat model and the fact that cPPs for a trusted operational environment (namely hypervisor) doesn't exist.

2. Threats and Assumptions

2.1 Threats

T.LOCAL_ATTACK: An attacker can act through unprivileged access on the same computing platform on which the application executes. For example, attackers may provide maliciously formatted input to the application in the form of files or other local communications thus providing unauthorized access to plaintext sensitive data.

T.UNAUTHORIZED_ADMINISTRATOR_ACCESS: An attacker may attempt to gain administrator access to the application by nefarious means such as masquerading as an administrator to the application, replaying an administrative session (in its entirety, or selected portions), or performing man-in-the-middle attacks, which would provide access to the administrative session. Successfully gaining administrator access allows malicious actions that compromise the security of the application to gain access to data.

T.WEAK_CRYPTOGRAPHY: Attackers may exploit weak cryptographic algorithms or perform a cryptographic exhaust against the key space. Poorly chosen encryption algorithms, modes, and key sizes will allow attackers to compromise the algorithms, or brute force exhaust the key space and give them unauthorized access allowing them to read, manipulate and/or control the traffic with minimal effort.

T.UNTRUSTED_COMMUNICATION_CHANNELS: Attackers may take advantage of poorly designed or non-secure protocols or poor key management to successfully perform man-in-the middle attacks, replay attacks, etc. Successful attacks will result in loss of confidentiality and integrity of the critical network traffic, and potentially could lead to a compromise of the application itself. Attackers may attempt to target applications that do not use standardized secure tunneling protocols to protect the critical network traffic. This threat is of particular concern when an application uses protocols that have not been subject to extensive peer review.

T.UPDATE_COMPROMISE: Threat agents may attempt to provide a compromised update of the application which undermines the security functionality of the application. Non-validated updates or updates validated using non-secure or weak cryptography leave the updated application vulnerable to surreptitious alteration.

T.PLATFORM_UPDATE: Updating the platform that the application operates on could break application's functionality. As such an end user might choose not to update the platform, thereby preventing the patching of known issues on the platform. An attacker could exploit such unpatched vulnerabilities in the platform to then mount an attack on the application.

3. Assumptions

A.PLATFORM: The TOE relies upon a trustworthy computing platform for its execution. This includes the underlying platform and whatever runtime environment it provides to the TOE.

A.PROPER_USER: The user of the application is trusted to use the software in compliance with the applied enterprise security policy.
A.PROPER_ADMIN: The administrator of the application is trusted to administer the software within compliance of the applied enterprise security policy.

A.SECURE_LOCATION: For enterprise servers that run enterprise applications, it is assumed that these servers are housed in a physically secure location

3.2 Organizational Security Policy

There are no OSPs for applications.

4. Security Objectives

4.1 Security Objectives for the TOE

O.INTEGRITY: The TOE will provide a means to verify the integrity and authenticity of downloaded updates.

O.MANAGEMENT: The TOE will provide authorized administrators (and no other users) with the capability to configure and apply security policies.

O.PROTECTED_STORAGE: To address the issue of loss of confidentiality of sensitive user data in the event of loss of physical control of the storage medium, data-at-rest protection will be used. This involves encrypting sensitive data and keys stored by the TOE in order to prevent unauthorized access to this data.

O.PROTECTED_COMMS: To address both passive (eavesdropping) and active (packet modification) network attack threats, conformant TOEs will leverage a trusted channel for sensitive data. Sensitive data includes cryptographic keys, passwords, and any other data specific to the application that should not be exposed outside of the application.

O.PRIVACY: PII is sensitive data that needs to be well controlled; as such an application should only access and disseminate this data with explicit authorization from the user.

O.BEST_PRACTICES: As a best practice, the TOE will use only documented services and APIs of the runtime environment.

4.2 Security Objectives for the Operational Environment

OE.PLATFORM: The TOE relies upon the underlying platform for its security and as a result this platform must be trustworthy. It is the organization's responsibility to ensure that the platform meets the trustworthiness requirements of the organization's security policies.

OE.PROPER_USER: The user of the application uses the software within compliance of the applied enterprise security policy.

OE.PROPER_ADMIN: The administrator of the application software is trusted to administer the software within compliance of the applied enterprise security policy.