

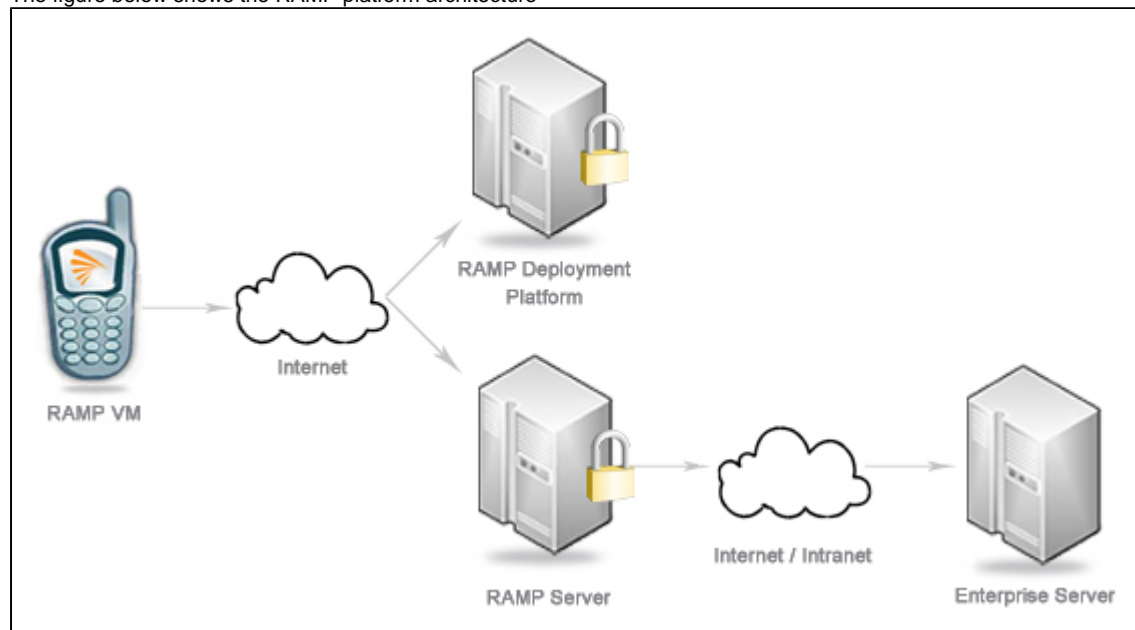


RAMP Security

Overview and Executive Summary

The RAMP secure mobile enterprise application platform (MEAP) is intended to be used for developing a wide spectrum of mobile applications. For applications to be successful, users and distributors must have confidence in the security of the application. This white paper explains why the RAMP platform can be trusted from a security perspective.

The figure below shows the RAMP platform architecture



The following must be secured:

- The RAMP VM must be installed on the mobile device from the RAMP deployment manager.
- Features must be retrieved by the RAMP VM from the RAMP Server.
- The RAMP VM must communicate with the enterprise server via the RAMP server.

The important point to note is that most communication takes place over an untrusted network: the internet.

Digital security, like a chain fence, is only as strong as its weakest link. The RAMP platform ensures that each link from the RAMP VM to the RAMP server and to the enterprise server can be trusted. This trust comes from:

- The use of digital certificates and digital signatures. The RAMP VM can be signed using a code signing key.
- The use of industry standard security protocols; in particular the Web Services Security and XML Encryption standards.
- The use of strong cryptography.

Strong Cryptography

The RAMP platform provides U.S. NIST validated implementations of the following:

- Digital signature verification using 1024-bit to 4096-bit RSA (NIST certificate #427).
- Hashing using the Secure Hash Algorithm (SHA-1) (NIST certificate #876).
- TDES encryption and decryption in ECB and CBC modes (NIST certificate #723).
- AES encryption and decryption in ECB and CBC modes (NIST certificate #886).
- Secure random number generation (NIST certificate #508).

RAMP VM Installation

Ideally a user should be able to verify that the RAMP VM that they install comes from a trusted source. The RAMP VM itself can be code signed.

Feature Installation

A RAMP VM can only connect to predefined RAMP servers. This ensures that an application created using RAMP can only send a password

to a predetermined set of RAMP servers; it is not possible for a RAMP VM to connect to unauthorized servers. Nevertheless since the content is retrieved over the internet one must ensure that the feature delivered by the RAMP server is the same feature that is delivered to the phone. Some hostile node between the RAMP server and the RAMP VM should not be capable of changing the feature in any way.

The RAMP Server signs every feature requested by a RAMP VM. If the feature is modified between the RAMP server and the RAMP VM, the signature will be wrong. The RAMP VM verifies the signature of any feature before installing and running it. Should the signature be wrong, the RAMP VM displays a fatal error message to the user and logs the error at the RAMP server.

Application Security

A RAMP VM must have the ability to communicate with an enterprise server. This communication goes via the RAMP server that provides data compression, image scaling and other functionality important for the limited capabilities of mobile phones.

The presence of the RAMP server can pose a challenge to security; data must be encrypted for confidentiality from the RAMP VM but if the RAMP server needs to decrypt the data, the provider of the enterprise service has no guarantee that confidential information might not be compromised at the RAMP server. This problem can be completely eliminated by having the RAMP server hosted and managed by the enterprise service provider. The RAMP server can be hosted in the same data center as the enterprise services themselves, eliminating 3rd party security risks. The prevalence of this problem in WAP led to it being colloquially known as "The gap in WAP". In fact the problem is not unique to WAP but exists in the Opera Mini browser and SIM Toolkit-based applications as well.

The problem with the gap in WAP is that all security takes place at the **transport layer**. There are many widely deployed transport layer protocols such as SSL, WTLS and 03.48 security (for SMS). The solution to the problem is not to secure data at the transport layer but rather at the **application layer**. There are advantages to performing security at the application layer:

- Selective encryption can be used; if information does not need to be kept secret it does not need to be encrypted. This can speed up applications (SSL in particular can be very slow on mobile devices because of a rather complicated handshake protocol).
- Clients can encrypt different messages with different message keys; for example some message parts could be encrypted with a key known by the RAMP server while other parts could be encrypted with a key known only to the enterprise server.

For many years there was not a widely accepted standard for application-level security meaning that service providers were faced with the problem of using imperfect transport layer security or putting together an *ad hoc* security protocol at the application layer. Fortunately, the situation has improved in the last few years with the advent of web services and, in particular, with the 2006 publication of the WS-Security protocols. The RAMP VM supports a small but powerful subset of WS-Security (and related XML encryption and signature standards) that allow clients to:

- Establish session keys
- Encrypt and decrypt data for Message Confidentiality
- Verify message integrity.

The WS-Security functionality is exposed via RAMPScript functions as detailed below.

Session Key Establishment

WS-Security draws a distinction between **key transport** and **key wrapping**.

Key Transport

Key transport uses a public key (e.g. an RSA key) to encrypt a symmetric key (e.g. a TDES or AES key). The RAMP VM can be packaged with an arbitrary number of public keys. It is always packaged with at least one since it needs the ability to verify the signature of a feature downloaded to it. However, one or more public keys of the RAMP server and the enterprise server can also be packaged with the RAMP VM. The RAMP VM uses RSA OAEP for key transport.

The `wssTransportKey` function is used to encrypt a symmetric key. Additionally, RAMPScript provides the `wssRandomKey` function for generating good quality (high entropy) random TDES session keys.

Key Wrapping

RAMPScript supports a function, `wssTransportKey` that implements the TDES key wrapping mechanism defined in the XML encryption specification as used in WS-Security. The mechanism includes an integrity check so that altering the value of an encrypted key means that it cannot be decrypted. The `wssUnwrapKey` function can be used to decrypt a key.

Message Confidentiality

XML Encryption defines how to use TDES in CBC mode for message encryption. In particular it uses a random initialization vector and a random padding scheme. Applications can access the encryption and decryption algorithms via the `wssEncrypt` and `wssDecrypt` functions. The RAMP VM will report an error if a message cannot be decrypted if the padding is wrong (which would indicate that a message has been modified after encryption).

Message Integrity

WS-Security uses digital signatures for message integrity. Clients can use the *wssVerifySignature* method to verify the signature of a message; i.e. to check that a message has not been modified between the server and the client and that the message must have originated with the server.

The signature algorithm supported is RSA PKCS#1, v1.5 which is the most widely used signature algorithm.