CHRIS/13/8F

The Canadian Experience Implementing The PRIMAR Security System

Greg Levonian
Michael J. Casey

The Canadian Experience Implementing The PRIMAR Security System

The Canadian Experience Implementing The PRIMAR Security System

1 PSS (PRIMAR Security Scheme) Overview

PSS (the PRIMAR Security System) is a security protocol, which is designed to facilitate the secure distribution of S-57 data. It has subsystems to enable data protection (including access control) as well as a mechanism for data integrity and authentication. Noteworthy is that each component is independent of the other – the data protection component could be implemented without the data integrity assurance and authentication component or the data integrity and authentication component could be implemented without the data protection component.

It is not the purpose of this paper to define or explain PSS but rather to give our impressions and explain our experiences implementing its key security modules. A brief outline of the two components is presented below. Further information can be found in the document "*Security Interface Version 1.2*" available from PRIMAR/ECC.

1.1 Encryption

PSS uses encryption technology to make unauthorized data access difficult. Clear S-57 base and update files are encrypted with CELL_KEYS, which are never revealed to end-users. The encrypted files are then widely disseminated and are assumed to be readily available to mariners. The Blowfish algorithm is used for this data file encryption.

Users identify themselves with USER_PERMITs and present requests to decrypt cells. At the RENC, using information within the USER_PERMITs, CELL_PERMITs are constructed. These CELL_PERMITs contain encrypted versions of the CELL_KEYs and will allow users to decrypt their respective cells. The CELL_PERMITs given to one user will not work for any others. CELL_PERMITs also have expiry dates which ECDIS units are expected to respect.

It is worth noting that the data access control mechanism requires a trust relationship with ECDIS manufacturer. An unscrupulous ECDIS system or a somewhat sophisticated hacker, who had access to the ECDIS unit, could easily, permanently decrypt cells, if their CELL_PERMITs were available.

A more complete description of the encryption component of the PRIMAR Security

Protocol can be found in the document "*Security Interface Version 1.2*" available from PRIMAR/ECC.

1.2 Data Integrity Assurance and Authentication

PSS provides for data integrity assurance and authentication using the SHA1/DSA algorithms.

The system has a top-level SA (Scheme Administrator) private key/public key pair (which currently held by PRIMAR). The SA public key is made widely available (in X509v3 format, as well as PRIMAR's own self-signed certificate format). The SA private key is used to sign data producer's public keys, resulting in HO/RENC certificates.

Each S-57 data file, be it a base or update file, has a signature file, comprising of one or more entries. (This allows for multiple organizations to sign the same file.) Each entry is a signature followed by the necessary certificate.

The procedure to verify a signature/certificate entry is as follows: First the SA signature of the HO/RENC's certificate is verified. If the signature verifies, the public key is extracted and used to verify the data file.

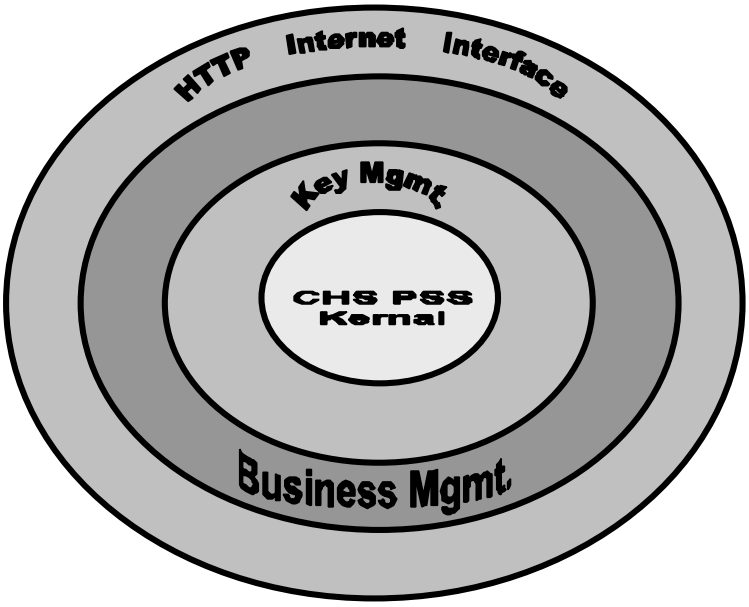
A more complete description of the encryption component of the PRIMAR Security Protocol can be found in the PRIMAR/ECC document "*Security Interface Version 1.2*"

2 Project Objectives

2.1 Development of a PSS Back-end Kernel

If PSS were to be accepted as an IHO standard to be used by multiple HOs and RENCs, along with good documentation, reference code for a kernel, that would handle all PSS security functions, would be a necessity.

The following figure shows the kernel definition.



Having each HO or RENC re-invent the wheel and re-code software to implement PSS would be waste of effort and would introduce many opportunities for error. Since PRIMAR does not make their source or object code public, we proceeded with an independent implementation.

2.2 Expertise Acquisition

Like many hydrographic offices CHS expertise within the security field was very limited. Computer security, encryption, and authentication are particularly difficult, technical, fields. The situation is further complicated by the media hype and politics that security technologies have engendered. At the same time, the major ideas of encryption and authentication are deceptively simple, even dangerously so, and without hard facts, or expert advice, dangerous assumptions are easy for lay people to make.

2.3 Familiarization with PSS

In the last several years, PSS has been increasingly mentioned as a solution to the perceived security issues surrounding S-57 data distribution. It was felt that in order to make informed decisions about this proposed PRIMAR standard, we would need to gain a much higher degree of familiarity with it than we had.

2.4 Document Level of Effort Required to Deploy PSS

To date, no HO or RENC, with the exception of PRIMAR (who designed the system and who's experiences because of this would not be typical), has an implemented the PSS back-end. Though our PSS assessment project did not include a complete turnkey installation, by going halfway down the road, and by producing a kernel, we felt that we would be in a position to somewhat assess what the final level of effort might be like.

So an additional goal was to find out, regardless of how good a scheme PSS is, how easy or hard is it to implement.

3 Tools used, Methods Employed and Experiences Gained

3.1 Development Language

The choice of development language is influenced not only by its technical merit and its suitability to the project, but also by various non-technical issues, such as the support available for it or the availability of programmers proficient in the language.

For our PSS project Java (version 1.3) was chosen as the implementation language. Not only is Java a modern, clean, object-oriented language, with advanced features, but also, with the increasing popularity of Java, information about Java and programmers versed in it are relatively easy to find. Its syntax is reasonably straightforward and readable and code written in it is very self-documenting. The "write once, run anywhere" Java portability was a very important consideration as well, since, as the goal of the project was to produce reference code, we did not want to be platform specific.

The availability of excellent cryptographic libraries for Java was also a major consideration.

3.2 Cryptographic Tools

Java provides both encryption and digital signature classes using a refined "provider" interface called JCE (Java Cryptographic Extensions). This means that, though the interface to the classes is defined by SUN, third parties can plug in their own algorithms

to do the actual key generation, random number generation, digital signing, encryption, decryption and so on.

SUN, of course, provides a JCE provider, which make JCE a self-sufficient extension, and it was the SUN JCE Provider that was used for our PSS project. We chose version 1.2, since, at the time 1.2.1 was beta, and we saw no reason to use a pre-release version. Installation is also somewhar more difficult with version 1.2.1 than 1.2, since with version 1.2.1 the JCE interface must be installed independently of the JCE provider classes. Something else worth noting is that JCE will be bundled with Java 1.4 (already available in beta), making installation a non-issue.

3.3 Development Environment

The choice of development environment has a large impact on, not just the speed with which software is delivered, but perhaps more importantly, on the quality of the code produced. Though Java has matured significantly since its inception, one area where big improvements are possible is with its IDEs (Integrated Development Environment). Compared with other systems Java's IDEs are poorly designed, and often buggy.

This begin said, the best of them, in our opinion, is certainly IBM's VAJ (Visual Age for Java). Although VAJ has a steep learning curve, it assists the programmer in many ways, and allows for true object oriented software development by completely shielding the programmer from the underlying file system.

The downside of this is that installing extensions, such as the JCE, into IBM VAJ, can be tricky, and SUN documentation will not apply. Purchasing IBM's telephone support for VAJ contract is almost a necessity.

4 PSS Assessment

An assessment of PSS was one of the primary goals of our study. During our PSS implementation, a number of our opinions changed, as we understood motivations for doing things a particular way, or we realized improvements that could be made.

Below, are some of our impressions of PSS.

4.1 Functionality and Implementation

First and foremost, does PSS do, what it is supposed to? The answer to this question is a simple yes. It works as advertised, and a proper implementation of PSS will certainly

deliver a reasonable system. However, it has a number of areas in which it should be improved.

At first glance the encryption key lengths seem short and hence the encryption weak. But this is not a real problem since stronger encryption makes no sense. Any copy protection system has more open vulnerabilities than even the weakest non-trivial encryption scheme.

A nuisance issue is that checksums are calculated inconsistently, on data, and in a manner, that leaves one questioning the motivation of the design.

A more serious issue is that portions of some keys are appended to themselves, and the claim is made that this somehow increases security. But making key length longer by reusing the same bits does not add any additional security. Key bits are only useful insofar as they are a source of randomness.

Finally, the PSS proprietary certificate format does not contain any information about organization name or other identification. This is an issue when an S-57 file has more than one signature accompanying it. Though the first signature must to be the HO's that produced the data, the origins of other signatures are unspecified. The user has no way to determine from who the other signatures are; at least without consulting some kind of external database not specified within PSS.

4.2 In house vs. Off-The-Shelf Systems

PSS is not an off-the-shelf product. However, this does mean that it is completely a proprietary design. It does use a standard, published:

- Encryption algorithm (Blowfish)
- Standard data padding scheme
- Message digest or hash function (SHA1)
- Digital signature algorithm (DSA)

However, in other important areas, where it could have used standard and open formats, it chooses to use in-house proprietary solutions. Below is list of non-standard, proprietary solutions within PSS, where an open, standards-based solutions exit.

- Encrypted file format
- Digital signature format
- Certificate format

Not only do these proprietary formats reduce interoperability but they also make implementation a more error-prone task, since custom code must be written, where standard libraries could have been used. It also makes testing more difficult since

standard, previously-existing software components cannot be used in any testing procedures, and all test procedures must also be written in-house.

The HO/RENC certificate format defined by PSS is non-standard for no apparent reason. The X509v3 format is a particularly widely accepted standard, and there is no reason why PSS could not use it. This situation is further confused by the fact that the SA, (PRIMAR) certificate is provided in X509v3 format.

We feel these are serious yet fixable issues. The standard could be migrated to more open data formats.

4.3 Interoperability

All the issues discussed previously are not just implementation issues - they also make PSS essentially uninteroperable with, not just other existing security software, but just as importantly future systems. Though as mentioned standard algorithms are used, all data formats are proprietary, and this means that no information can be communicated into or out of PSS from other security systems.

This would be acceptable if S-57 data protection was a closed problem. However, we feel that data protection and authentication are very broad horizontal problems. Data encrypted and signed by one program should not assume that the same program will be decrypting it and verifying its signature. Similarly public keys and certificates need to be accepted from many sources, signed, verified, stored, and so on. Data distributors need to be able to create their own trust models, using their own defined certificate signature hierarchies. This is why it is imperative that certificate formats are standard.

This is also why, for example, PKI (Public Key Infrastructure) is an emerging standard. PKI is not driven by any one application but rather with the idea, that data encryption and authentication is a broad horizontal issue, a service that many user applications will need to access. For countries such as Canada, which are building PKI based infrastructure for secure communication with its citizens the complete uninteroperability of PSS with PKI is a grave concern.

The trend in the IT industry is for encryption and authentication to be built into communication infrastructure, as well as major operating systems. By introducing proprietary data formats PSS will not be able to take advantage of emerging common infrastructures.

4.4 Quality of Documentation

PSS documentation needs to be improved.

Our understanding at the start of our PSS project was that PRIMAR had provided us with documentation and test data, which would enable us, at least in principle, to implement the PSS back-end. This opinion was supported by sections in the documentation, which describe PSS processes that would occur only at a RENC or HO.

However, during implementation we found the documentation ambiguous and, in areas, incomplete. This necessitated communications with PRIMAR/ECC to fill in gaps in the documentation and to clarify ambiguities. We felt that, since the documentation had never been used before by someone who did not already have intimate knowledge of PSS, this was a normal situation. Bugs in documentation, just like in software, are often found in use. But PRIMAR/ECC has now stated the documentation was never written to code PSS back-end - it was designed only for ECDIS manufacturers to add PSS services to their front-end, ECDIS software. Any PSS back-end implementation would have to occur within a PRIMAR/ECC partnership framework.

4.5 Ease of Implementation Resource requirements

Implementation level of effort is difficult to assess since it has as much to do with the implementer as it does with work required. An expert can find a task easy that a novice would find very difficult. Implementation effort is also is relative and has a great deal to do with expectations. We judge something difficult, if we expect it to be easier than it is.

With these caveats in mind, objectively, PSS is not a particularly complex system. Nothing about it is especially difficult to understand, and with the proper expertise and tools, difficult to implement. The observation has been made earlier that PSS is a narrow, vertical system, which does not interoperate with standards-based systems, that it has inflexible trust models and so on. However, it is worth noting that this is a functionality/simplicity trade-off; these attributes make PSS much simpler than it would be otherwise.

However, it would be very easy, to underestimate the resources required to implement PSS.

One reason for this is that the difficulties with a PSS implementation lie not within PSS itself, but rather with encryption and data authentication in general. The field is highly technical and it simply requires expertise that most HOs do not have. Contracting out PSS implementation is only a partial solution – this option does not completely obviate the problem of acquiring expertise, since in house knowledge will still be required, for working with the implementation and for support. Security is not an "implement and

forget" operation.

The required expertise for a PSS implementation goes beyond security systems. Any software system is easy to underestimate and debugging and quality assurance is more difficult that it at first appears.

Also, technical resources are not the only ones that need to be considered. Security systems are not just "tacked on" to existing processes, they impact them in a very fundamental way often making old ones obsolete while creating new ones. Key databases have to be reconciled with customer databases and distribution formats have to be reconsidered. Telephone support has to accommodate new support issues. Disaster recovery plans have to include security system failures. The list goes on and on. This all means that any PSS implementation has to consider broad organization-wide issues, as well as their resource requirements.

The lack of PRIMAR/ECC endorsed reference software also makes implementation more difficult than it need be. Such source code would not only make the task of implementation much easier in itself, but it would also help document the system.

Finally, though the quality of the available documentation has been previously discussed in it's own section above, it is worth noting here, that non-existence of a clear back-end implementers guide, certainly impacts on the ease of implementation significantly.

Phases of PSS Project needed to be conducted concurrently (for example, Java expertise was acquired while the IDE was explored) so the following are estimates. But for informative purposes, below is a breakdown of how long various component of our PSS Project took.

Java learning and familiarization	2 Months
IDE learning and familiarization	1.5 Months
Java JCE learning and familiarization	1.5 Months
Encryption/Decryption support routine implementation	1 Months
Digital signature support routine implementation	1.5 Months
CHS PSS Kernel Library design and implementation	3.5 Months
Total	11 Months

A senior systems analyst with approximately 10 years experience performed these tasks.

The only capital expenses were the IBM VAJ software and its phone support contract, which amounted to approximately 300.00 USD and 500.00 USD respectively.

It is worth noting here that the project included neither integration with our business system nor did it include any communication protocol (e.g. HTTP) implementation.

4.6 Ease of Migration vs. Lock-In

Lastly a very important but often overlooked criteria in defining standards or choosing software is how easy it is to migrate to a different standard or software, should this be desirable in the future. In it's current state, PSS does not fare well particularly well along this dimension.

Because its data formats are proprietary it, or parts of it, could not be easily replaced without the co-ordination of all PSS users. Since PSS users would be a very diverse group - HOs, RENCs, ECDIS manufacturers, and mariners, etc., it seems that developing consensus for a change would likely be difficult; even if a clearly better way of doing things were to come along, PSS as is would have to be supported for several years.

It is worth noting that PSS may have already reached this "critical mass". Although no HOs or RENCs aside from PRIMAR have implemented it, a number of ECDIS manufacturers have. One of the most compelling arguments for PSS standardization is the level of ECDIS support for PSS.

Does the level of ECDIS manufacturer support justify IHO PSS standardization despite the concerns raised in this paper?

5 Recommendations

The job of a critic is an enviable one. It's easy to point out flaws, harder to suggest improvements, and most difficult to properly design and implement a system. PRIMAR/ECC has tackled a very difficult problem and has presented a working solution. Nonetheless, the following issues need consideration.

5.1 Is an S-57 Security System Standard Necessary?

The CHS position that S-57 security systems are unnecessary, even detrimental, is well know and will not be elaborated here. However, for completeness it does need to be stated. Before we consider what kind of security standard we want, we should ask ourselves, if we need one at all. The Canadian position is that S-57 security standardization currently is contrary to the S-57 community’s interests.

5.2 Legacy Idiosyncrasies

The PRIMAR System contains a number of features, which are at best legacy

idiosyncrasies. Though it is understandable why PRIMAR, considering its ECDIS manufacturer user base, would be reluctant to make changes to PSS, the IHO interests in a clean design might be otherwise.

The formats of the user and cell permits contain CRC32 checksums, which are calculated in an inconsistent manner. Though the cell permit checksum is calculated on the entire cell permit, the user permit checksum is calculated only on part of the user permit data, ignoring the rest of it. The user permit checksum is calculated on underlying binary data, before it is converted to hexadecimal, while the cell permit checksum is calculated on the ASCII hexadecimal text of the data it represents. Finally, though the cell permit checksum is encrypted (though there is no benefit in doing so), the user permit checksum is in clear form. This inconsistency adds up to unnecessary complexity.

More importantly, the checksums aren't needed at all. The idea behind them, ostensibly, is to make errors in transmission detectable – However the designers of network stacks, communication protocols, and storage media, have the responsibility of assuring error-free transmission and storage, and the application layer simply is not an appropriate place for error-checking of this type.

Another perhaps even more idiosyncratic feature is the calculation of HW_ID6 that is used to encrypt cell keys. The idea is that, though the HW_ID is only 4 bytes long, a 6-byte key is desired, so, the first two bytes of the HW_ID are appended to the end of the HW_ID to create the HW_ID6. However, key length is only increases security to the extent it is a source of randomness. The entropy within the HW_ID is the same as in the HW_ID6, so the HW_ID6 is no more secure than the HW_ID. This should be changed.

5.3 Encrypted Data File Format

Though PSS uses a standard encryption algorithms and a standard padding scheme, it's encrypted file format is not compatible with standard based encrypted file formats. This is a serious problem, and a standard file format should be adopted.

5.4 Signature File Formats

Though PSS uses a standard message digest and digital signature algorithms, its signature file format is not compatible with standard based signature file formats. This is a serious problem, and a standard signature file format should be adopted.

Also file formats exist that combine the encrypted file with it signature and this should be considered for PSS.

5.5 Certificate Format and Infrastructure

X.509v3 digital certificates should be used not just for the SA, PRIMAR, but rather, they should replace the “Self Signed Key” format, and the HO/RENC certificate format.

There would be significant advantages to this. Not only would it would get rid of one more proprietary data file format, but also more importantly, the X509v3 certificate standard is very well supported. Basic X509v3 certificate support is even in Windows 98, Windows NT, and Windows 2000. X509v3 certificates are used in PKI, and the governments of many countries have already made commitments to use PKI based security systems for all general-purpose secure online transactions.

Other X509v3 advantages would result from the capability of X509v3 certificates to record multiple signatures. This would mean that more complex trust models could be set up, trust models that more realistically reflect the actual delegation of authority to an HO or RENC. For example, regardless of who else signs the CHS certificate, it should certainly be signed by the higher Canadian governmental body, from which we receive our mandate and our legitimacy. A RENC’s certificate should be signed by all HOs whose data the RENC distributes. The constraint that PSS certificates allow for only one signature is very limiting.

With the use of X509v3 certificates the role of the SA can be diminished considerably, while the overall security of the system increases. This would be because, though currently the SA has to sign every HO/RENC certificate, and is responsible for the assuring their legitimacy, with X509v3 certificate use, this entire problem can be turned back onto the HO and RENCs themselves. Commercial CAs as well as governments are willing to sign X509v3 certificates, and very thorough procedures already exist for certificate (and private key) issuance.

With PKI based X509v3 certificates users would also be able to take advantage of a great deal of infrastructure to obtain certificates, and verify them. X509v3 certificates are already available from many websites as well as from X500 LDAP directories.

X509v3 certificates would also eliminate the current problem that, when a data file contains more that one signature, aside from the first signature, which has to be the HO’s, the user has no way of determining whose the other signatures are. This problem arises because PSS certificate format only contains the certificate owner's Public Key and the SA signature of it. X509v3 certificates on the other hand contain a rich data set including owner name, organization, date of issue, and so on. With this information the user would know much more about from where legitimacy comes from.

A problem that is not addressed by PSS is that of certificate expiry and evocation, and again the X509.v3 standard has a built-in solution.

Finally, the fact that with PSS the SA must sign every certificate – and nobody else can –

means there is a single point of failure in the entire authentication mechanism. If the SA's private key were ever compromised, spoofing attacks would be possible. Not only would X509v3 root certificates be less likely to be compromised in this way, since they would be maintained by professional CA organizations, but the X509v3 ability to contain multiple signatures would also significantly reduce the possibility of a successful spoofing attack.

5.6 Documentation

Naturally, PSS requires good documentation, not for just ECDIS manufactures, but also for HOs and RENCs. In our opinion the current documentation is not adequate and anyone who is implementing PSS will require consultation and help from PRIMAR. PSS documentation should be improved.

5.7 Reference Software

Though PSS needs better documentation, good documentation cannot replace reference software.

The CHS has done some work in this area, and we are making the results of our work freely available but true reference software should be endorsed by PRIMAR, or if the IHO accepts PSS as a standard, by the IHO itself.