現場まかせにしないセキュリティ設計、評価、実装のあり方について ーセキュリティは現場で起こっている

松下電工株式会社　福田　尚弘

複雑化、モジュール化したシステムからなるコンピュータ社会にあって、本来セキュリティ設計、評価、実装は重要な位置を占めている。システム開発、運用の分業化、モジュール化によってその傾向は強まっている。しかしながらISMSやISO15408、CMM、ISO12207などの仕組みや枠組みはドキュメントとして揃っているにもかかわらず、その手法に関してはいまだ職人芸に支えられているままである。仕組みや枠組みも重要であるが、本来、セキュリティの脆弱性はちょっとしたミスや理解不足から起こる。これは人間が行う以上どうしても発生する問題である。仕組みや枠組みは責任の所在を明らかにできるが、ミスや理解不足は現場で起こる問題であってゼロにすることは大変困難である。ここで問題なのが、最終的にセキュリティ設計、評価、実装の手法が依然として現場まかせとなっていることである。
ここでは、様々な業種におけるセキュリティ対策の実態を述べ、それを受けて現場まかせにしないためにはどのような工夫が考えられるかを述べ、最後に理想的なセキュリティ設計や実装方法のあり方を模索する。

1. セキュリティ対策の実体は現場のエンジニアに依存する

24（TWENTY FOUR）という米国で大ヒットしているテレビドラマをご存知だろうか？米国連邦機関の組織であるテロ対策ユニット（CTU）の捜査官ジャック・バウアーがテロリストと戦うサスペンス・アクションの内容である。ジャックはCTUから任務を受けてテロと戦うわけだが、神出鬼没で相手の裏をかくテロとの戦いでは往々にしてCTUのルール、CTUの指示を無視して行動することを余儀なくさせられるのである。また、CTUの指示を無視したり欺いたりと行動するため不遇であることが多い。

まさに情報セキュリティの分野とこの世界はオーバラップして見える。テロはセキュリティ上の脅威であり、セキュリティ対策を行う現場の担当者はジャックである。そしてこの戦いに終わりはない（人気シリーズなので次々と次回作がリリースされるのではあるが）。また、CTUは会社組織に見え、ISMSやISO15408、CMMなどのルールでジャックを縛っている構図が見て取れる。

何が言いたいかといえば、ここでの「主人公はジャック」であるということである。全てはジャック、つまり開発者の腕にかかっているのである。

2. 業種ごとに異なるセキュリティ意識

政府・自治体、教育などの機関、金融、エネルギー、情報通信、医療などのインフラ系、製造、運輸、サービス、不動産等の企業、または個人など、異なる業種または業態で異なるセキュリティが要求されると考える。つまり、業種毎にセキュリティ上で重要なポイントが異なる。たとえば、高レベルの顧客情報の保護（金融）、企業機密管理（エネルギー）、情報セキュリティ運用管理（金融）、外部との接続制限（エネルギー）、入退室管理（金融、エネルギー）、プライバシー（運輸）、システム運用管理（情報通信）、セキュリティ対応の手順化（金融）、セキュリティ教育（金融、運輸、情報通信）などである（警察庁：平成17年度「不正アクセス行為対策等の実態調査」からhttp://www.npa.go.jp/cyber/research/）。

それ以外にも、情報機器への信頼性やシステム管理における要求レベルも様々であり、それぞれが相互に接続される場合も考えられる。セキュリティ意識が異なれば、その対策も異なってくる。従って、業種ごとに異なるセキュリティ対応が要求されると考えられる。また、図1の警察庁の「不正アクセス行為

対策等の実態調査」にある「セキュリティポリシーの規定事項」(平成15年～17年までのデータ比較)を見ると、3年間の比較でも、基本方針、組織、運用管理においてはほぼ対応されているにもかかわらず、システム開発やメンテナンスの規定対応はほぼ低い数値のままである。つまり、セキュリティの方針、ルール、運用体制は確立したが、実質的なシステム開発やメンテナンスはまだ十分でないとも見て取れる。一方で、脆弱性調査は利用されてきている。脆弱性検査は全体で18.8%、意識の高い業種では約44%～28%行ったとされている(警察庁:平成17年度「不正アクセス行為対策等の実態調査」からhttp://www.npa.go.jp/cyber/research/)。

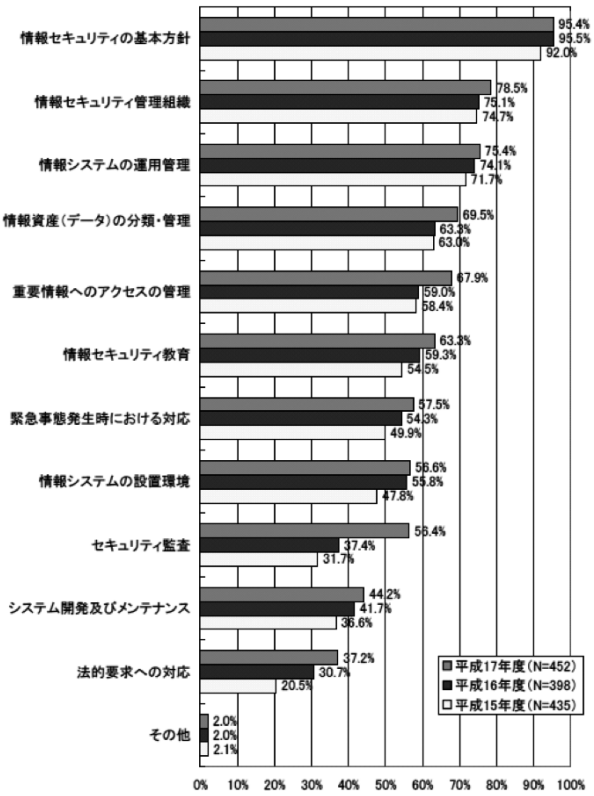
また、セキュリティ対策の効果については各業種から半数以上の回答で、「セキュリティの対投資効果」が見えない、「コストがかかりすぎる」という内容があった(警察庁:平成17年度「不正アクセス行為対策等の実態調査」からhttp://www.npa.go.jp/cyber/research/)。

3. システム開発者にとっての困りごと

ジャックのようにタフで優秀な捜査官であれば安心であるが、事はそうはいかないであろう。異なる業種に属する様々なレベルのエンジニアが様々な要求によって様々なセキュリティモジュールを組み合わせることでセキュリティ対策を行っているためだ。また、セキュリティの方針、ルール、運用体制が確立してきたことにより、どのように対応するかということが課題となってきている。さらに、「セキュリティ対策の効果が見えない」という問題がさらに事を難しくする。そこで、開発側にとっての課題をまとめると以下のようになると考える。

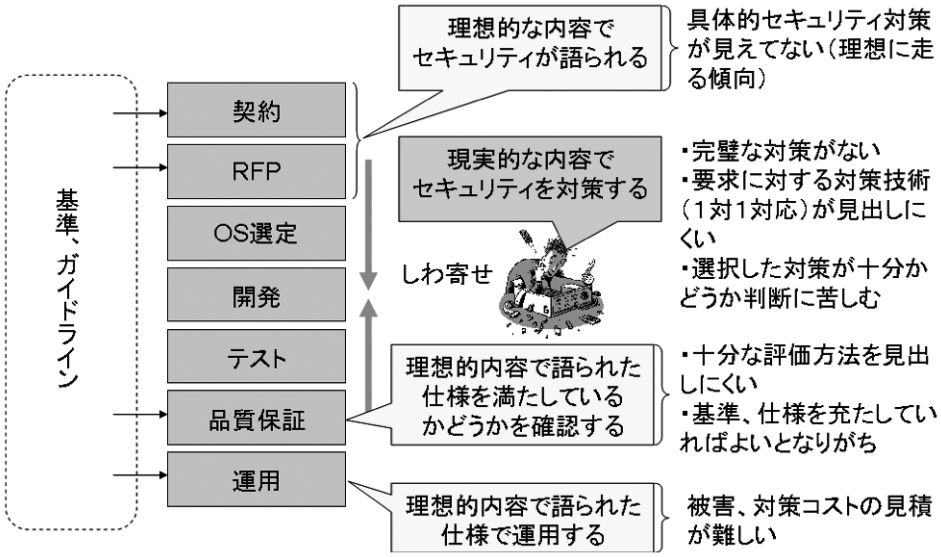
(1) セキュリティ対策をどのようにどこまでやったらよいか具体的にわからない。
(2) セキュリティの確保(担保)のためには様々な基準(ISMS、ISO15408等)はあるが、開発者にとっての具体的な手段を示していない
(3) 様々なセキュア化のための対策(脆弱性検査方法など)が存在するが、どれが何を担保してくれるのか明確ではない。
(4) システムの用途や条件によって(たとえば業種、業態によって)対策が異なる。

図1.「セキュリティポリシーの規定事項」
(平成15年～17年までのデータ比較)



出典:平成17年度「不正アクセス行為対策等の実態調査」のP.63、警察庁、HP:http://www.npa.go.jp/cyber/research/)

図2．セキュリティ対策のしわ寄せ構造



4

4. 開発プロセスの観点から見る

開発側にとっての課題を開発のプロセスから見ると、セキュリティ対策の実態は「しわ寄せ構造」として存在するとも言える。図2は、システムの開発プロセスをセキュリティ対策の観点から描いた図である（多少オーバーに書いた図なのでその点ご容赦願いたい。）

開発プロセスにおけるセキュリティ対策において、ISMSやISO15408、12207など基準やガイドラインは存在し、どのような項目を行えばよいかはある程度わかるが、具体的なセキュリティ対策は開発者に依存しているのが現状である。開発の契約またはRFPを作成する時点では、依頼者のセキュリティ要求が現実に可能な対策を超越する場合がある。特にセキュリティ技術は脅威に対する対策ということで成立している。従って、その時点で対策が成立したとしても、つまり品質側の立場からは、たとえその時点で脆弱性調査がパスされても保証はまったくないのである。完璧な対策は一切ないのである。また、運用側に至っては、被害と対策コストの見積もりが難しい点もある。本来、セキュリティの要求に対し対策技術が1対1に対応するのが理想であるが、実際の実装との整合を考えると必ずしもそのようにはならない。つまり、対策が十分かという問いに対しはっきりYESといえるレベルにないということになる。ここがセキュリティ対策の悩ましいところではないだろうか。

5. セキュリティ対策技術と実装の難しさについて

セキュリティの対策技術の例としてSSL/TLSを考える。ほとんどの人はSSL/TLSが安全かどうかを考えることはないであろう。個人情報の安全な通信にSSL/TLSはほとんど用いられている。IPsecについても同様であるが、ある種の「SSLまたはIPsec＝安全」という刷り込みがあるように思える。セキュリティプロトコルの専門化であれば、全てのバージョン、モード、パッチがあたっていないSSLやIPsecが安全であるということは言わないであろう。ここで問題となるのは、そのような刷り込みがセキュリティ対策で問題となる場合もある。また、SSLといっても正確にはHTTP、電子メール、VPNのSSLなどアプリケーションが異なれば若干異なるプロトコルになっ

ている。また、SSLを実装する場合での注意もある。HTTPはWebページへのアクセスを終了するとコネクションを切断する。一方、SSLは一度接続した相手のセッション鍵をしばらく保持する仕組みである。この問題はHTTPがKeep－Aliveすることで解決するが、システム設計者からするとサーバホストの構成によっては特性を理解して設計しないといけない。最近の開発スタイルで一からコードを書くものはないであろう。従って、深くは理解し得ないモジュール、OS、ライブラリおよびハードウェアを組み合わせ開発しているためどこかに落とし穴が存在する可能性が潜んでいると言えよう。業種によってセキュリティ要求やシステムが異なることを述べたが、標準技術で固めたシステムを開発する以上、適材適所というわけにいかず、どこかに歪が発生する。

6. OSIの階層別の観点から見る

単体のセキュリティ対策製品を考えるときOSIによるモデルで全てのレイアーで安全であることがその製品が安全であると言えよう。まずプラットフォームのOSが安全か、ミドルウェア、アプリケーション、ドライバに至るまでセキュリティ対策は必要であろう。その上でシステムとして構築した場合に全体で安全でなくてはならない。

7. 現場まかせにしない工夫

セキュリティ設計を行うには、シンプルかつモジュール化されていること、設計に透明性があること、最小限の情報を保持し最小権限で動作すること、エラー処理、ログ処理、性能低下を防ぐこと、フェールセーフ、耐障害など様々なことを検討しなければならない。それを業種別、業態別、実装形態別など様々なシステムの用途や条件に合わせて設計しなければならない。図2の「セキュリティ対策のしわ寄せ構造」で描いたが、開発プロセスの中で幾つかのプロセスはほぼ十分か、内容が充実しつつある。上流のセキュリティポリシーの方針、下流の脆弱性調査手法、監査方法などである。これら固まりつつある項目と、設計段階で必要となる対策技術とを結びつける対応マップが存在すると開発者には有用ではないかと考える。エンジニアは常に同じシステムを作っているわけではない。従って、違うセキュリティ要求に対し対応マップを参照することで容易に対策を導き出せるとありがたい。

8. まとめ

冒頭から多少偏った視点での寄稿であった点をご容赦いただきたい。「開発者の裁量/スキルによる対策にはしたくない」、「最終的に問題が起こったときその対策を選択した根拠を示したい」という思いがあってのことである。現状のセキュリティ対策状況を見ると、セキュリティ指針や仕組み、監査などが先行し、開発者がそれに難儀するまたは自分の裁量で対応するという状況ではないだろうか。真の意味でセキュアなシステムを作ることは難しい。このことを理解し、開発プロセスの上流から下流までにしわ寄せが発生しない開発体制の構築が望まれる。尚、本件は、セキュアプログラミングWGで検討中である。

参　考:

平成17年度「不正アクセス行為対策等の実態調査」、警察庁、HP:http://www.npa.go.jp/cyber/research/