SQUAREではじめるセキュリティ要求工学※

Nancy R. Mead[*1]　吉岡 信和[*2]

[*1] Software Engineering Institute, Carnegie Mellon University　[*2] 国立情報学研究所

会社などの組織の中で，セキュリティ要求を効率よく獲得し，規定するためには，一般の要求工学の場合と同様，そのための手順（プロセス）を定め，それに従って進めることが有効である．特にセキュリティの場合，システムを利用する直接的なユーザのみならず，システムの運用者や，システムを導入する企業のビジネス戦略や扱う顧客情報の取り扱いを決定する経営陣など，非常に多くの利害関係者（ステークホルダ）が存在し，それぞれの権限，文化を考慮して進める必要があり，特にプロセスが重要になる．SQUARE（Security Quality Requirements Engineering：セキュリティ品質要求工学）は，セキュリティに関するシステムの品質を高めるために定められたプロセスモデルである．本稿では，そのプロセスの詳細とSQUAREに関する活動に関して解説する．

SQUAREとは？

SQUAREは，カーネギーメロン大学（CMU）で開発され，ITのためのセキュリティ要求を獲得し，それを分類，優先度付けするための手順を規定している[1)]（詳細は，http://www.cert.org/sse/square.html を参照のこと）．特に，このプロセスでは，システム開発のライフサイクルの中で，その早い段階でセキュリティの考えを組み入れることに主眼が置かれている．このプロセスは，実システムのセキュリティに関するドキュメント作成や分析，さらにその改良，変更の管理・運用のためにも利用できる．

初期の頃のSQUAREは，2004年と2005年のサマープロジェクトとして，カーネギーメロン大学の修士の学生により，いくつかの顧客の事例[☆1]に適用された．そのためのツールのプロトタイプも開発されている．このドラフト版プロセスは，事例研究の後に改定され，**表-1**のように標準化された．表の中の最初の第1ステップから4ステップまでは，原則的にはセキュリティ要求工学よりも前に行うべきであり，適切にセキュリティ要求を獲得するために必要となる[☆2]．

どのようにSQUAREを使うのか？

SQUAREのプロセスは，幹部マネージャやステークホルダの支援のもとで，プロジェクトの要求を取りまとめるエンジニアとセキュリティの専門家の両方がそろうことにより最も効果的に運用できる．SQUAREにより適切なセキュリティ要求を獲得するためには，リスクが適切に評価された後にセキュリティの要求が獲得され，アーキテクチャやデザインの重要な決定がされる前にその要求が規定されるべきである．すなわち，リスク分析，セキュリティ要求獲得，アーキテクチャ選択という順序が最も効率のよいプロセスとなる．SQUAREでは，重要なビジネス上のリスクは，セキュリティ要求の開発プロセスの一活動として熟考する．以下から，SQUAREの各ステップの詳細を紹介する．

ステップ1．定義をステークホルダで合意する

「定義の合意」は，セキュリティ要求工学に先立ち必要となる．この合意は，どのような開発であっても行ったほうがよいが，ことセキュリティに関しては重要である．なぜならば，セキュリティに関連があるステークホルダは非常に多彩な知識的背景を持ち，その間のコミュニケーションが問題となるからである．さらに，システムが置かれる状況・組織によって必要になるセキュリティは異なる．たとえば，あるプロジェクトを考えたと

☆1 詳しいことは，
http://www.sei.cmu.edu/publications/documents/04.reports/04sr015.html
http://www.sei.cmu.edu/publications/documents/05.reports/05sr005.html
http://www.sei.cmu.edu/publications/documents/04.reports/04tn045.html
などを参照のこと．

☆2 より詳しいことは文献1)を参照のこと．

番号	Lite 番号	ステップ	入力	技術	参加者	出力
1	1	定義について合意をとる	IEEE 標準などの定義候補	構造化されたインタビューやテーマを絞ったフォーカスグループ	ステークホルダ，要求チーム	合意された定義
2	2	資産（asset）とセキュリティゴールを確認する	定義，候補となる目標，ビジネスドライバ，ポリシー，手順，事例	進行役がいるワークセッション（手を動かす会議）文献調査インタビュー	ステークホルダ，要求エンジニア	資産（asset）とゴール
3	—	セキュリティの要求を定義するための中間成果物の開発	可能性のある成果物（シナリオ，ミスユースケース，テンプレート，フォームなど）	ワークセッション	要求エンジニア	必要な中間生成物：シナリオ，ミスユースケース，テンプレート，フォームなど
4	3	リスクの評価（risk assessment）	ミスユースケース，シナリオ，セキュリティゴール	リスク評価手法，脅威分析を含む，組織のリスクへの耐久力に対する予想されるリスクの分析	要求エンジニア，リスクの専門家，ステークホルダ	リスクの評価結果
5	—	要求獲得技術の選択	ゴール，定義，候補となる技術，ステークホルダの熟練，組織のスタイル，文化，セキュリティの必要性のレベル，コスト，利益分析など	ワークセッション	要求エンジニア	選択された獲得技術
6	4	セキュリティの要求の獲得	中間生成物，リスク評価結果，選択された技術	JAD（ジョイントアプリケーション開発）インタビュー，文献調査，モデルベースの分析，チェックリスト，再利用可能な要求のリスト，タイプ，ドキュメントレビュー	要求エンジニアに取りまとめられたステークホルダ	セキュリティの要求に関する最初の断片
7	—	要求をレベル（システムやソフトウェアなど）に従って分類し，それらが要求なのか他の種類の制約なのかを見極める	最初の要求，アーキテクチャ	標準的な分類集合を使ったワークセッション	要求エンジニア，必要に応じて他の専門家	分類分けされた要求
8	5	要求に優先度をつける	分類された要求とリスク評価結果	AHP,Triage,Win-win など優先度をつけるための手法	要求エンジニアに取りまとめられたステークホルダ	優先度分けされた要求
9	—	要求の検査	優先度分けされた要求，候補となる形式的な検査技術	Fagan 手法，ピアレビューなどの検査手法	検査チーム	最初に選択された要求，決定されたときのプロセスや根拠などの文書

表 -1　SQUARE と SQUARE-Lite のプロセス

き，そのチームのメンバは，それまでの経験に基づき，何らかの定義を念頭に置いて開発を進めるであろう．しかし，その定義は，必ずしもメンバ内で合意できているとは限らない．たとえば，ある政府機関のためのセキュリティには，セキュリティの許可レベルに基づいたアクセス制御が必要になるが，他の組織には，物理的なセキュリティやサーバセキュリティが必要になったりする．すなわち，同じセキュリティでも想像していることが異なっているかもしれないのである．

☆3 このステップの資産の認識は，国立情報学研究所で行われたシンポジウムのパネルディスカッションの議論結果を受け，付け加えられた．そのため文献 1）などのステップ 2 には，資産は含まれていない．

ここで，合意を得るために定義を新たに熟考し，生み出す必要は必ずしもない．IEEE や SWEBOK のようなすでに規定されている情報源から選択したり，もしくは適合させたりすることで，かなりの領域の定義をカバーできる．すなわち，関係者が集まり議論テーマを絞ったフォーカスグループミーティングを開くことで，セキュリティ要求の活動に必要な，一貫性のある定義の集合を選び出すことができるであろう．

ステップ 2．資産とセキュリティゴールを認識する

「資産（Asset）とセキュリティゴールの認識☆3」は，IT システムを開発する際には必要となり，組織レベルで行われるべきである．この認識を組織レベルで行うことにより，組織の方針や経営上のセキュリティに関する

一貫性を経営陣も含め確認することが可能となる．異なるステークホルダは，異なるゴールを持っているかもしれない．たとえば，ある人材を供給するステークホルダは，個人情報の機密性を整備することに関心があるであろうし，他方，財務にかかわるステークホルダは，財務情報を許可なしでアクセスしたり変更したりされることがないことを保証することに関心があるにちがいない．このステップでは，経営上の経験者を含む利害関係のある組織の代表者たちを集めることが重要である．なぜならば，さまざまなステークホルダのゴールを認識したら，それらに優先度をつける必要があるからである．つまり，セキュリティゴールは，ビジネス上の方針にも影響し，システム開発者だけではその優先度を最終決定することは無理な場合がある．このコンセンサスが得られない場合，最終的には組織の最高責任者がゴールの優先度を決定する必要がある．

ステップ3．中間成果物を作成する

「中間成果物（artifact）の開発」は，これ以降のステップを実践するために必要である．ここで，セキュリティに関係する中間成果物には，下記が含まれる．

- システムのアーキテクチャ
- ユースケース図，ユースケースシナリオ
- ミスユースケース（ユースケースに対して，攻撃や対策を追加し，それらの間の関係を明示したもの），ミスユースケースシナリオ（詳しくは，本特集の「コモンクライテリアにおけるセキュリティ要求の規定の現状と課題」の関連研究を参照してほしい）
- 攻撃ツリー（attack tree）攻撃の可能性を抽象的なゴールから具体的な手順まで木構造でブレイクダウンしたモデル☆4

組織によっては，プロジェクトの手順に関するコンセプトや，プロジェクトのゴールが文章化されていなかったり，システムの通常の場合の利用手順書，脅威シナリオ，要求定義を助ける他のもろもろの文書すら存在しなかったりするかもしれない．しかしながら，これらの文書なしでは，要求の獲得プロセス全体が砂上の楼閣に終わるか，後のステップで，そのような文書を得るために何度もバックトラックが発生し，時間を無駄に浪費することになりかねない．

ステップ4．リスクを評価する

「リスクの評価」には，リスクの評価法に関する専門家と，ステークホルダや要求工学者のサポートが必要となる．リスクを評価するための多くの手法が存在し，その中から必要なものを選択することができる．そして，リスク評価の専門家は，組織の要求に基づいて，ある特定の手法を勧めるだろう．ステップ3の成果物は，このリスク評価のステップの入力となる．リスクの評価結果は，高い優先度で考慮すべきセキュリティ上のビジネス上の損失を見つけ出す助けとなる．リスクを評価していない組織は，概して，セキュリティ要求を獲得する際に，経営上のリスクに対する行うべき取り組みが行えていない．すなわち，そのような組織は，何がセキュリティとして解決されるべき問題なのかの真の理解がなく，暗号化などのセキュリティのメカニズムの選択に終始しがちである．

ステップ5．要求獲得技術を選択する

「要求獲得技術の選択」は，異なる背景知識を持つステークホルダが複数存在する場合には重要になる．なぜならば，ステークホルダがそれぞれ違う文化的背景を持つ場合に，コミュニケーションそのものが障害になり要求を効率よく獲得できないことがあるからである．これを克服するために要求獲得手法が有効である．比較的形式的な獲得手法としては，ARM（Accelerated Requirements Method：ブレーンストーミングや項目整理など3つのフェーズに分けて議論を行う手法）[2]やJAD（ジョイントアプリケーション開発：ユーザなども含む全ステークホルダを開発に参加させる手法）[5]，構造化されたインタビューなどがある．このほかにも，場合によっては，主要なステークホルダとともに，それぞれが必要となるセキュリティ要求を非形式的な話し合いでお互い理解することで十分な場合もある．

ステップ6．セキュリティ要求を獲得する

「セキュリティ要求の獲得」は，ステップ5で選択した技術を使った実際の要求獲得プロセスである．ほとんどの要求獲得技術は，どのように要求を獲得したらよいかの詳細なガイダンス（手引き）を提供している．これは，ステップ4で作成したミスユースケース，攻撃ツリーやそれに含まれる脅威のシナリオなどの中間成果物に基づいて行う．具体的には，これらの情報をもとに，機密性や整合性などシステムが持つべきセキュリティの要求を，形式的な手順のインタビューやグループディスカッションなどで整理していく．

ステップ7．要求を分類する

「要求の分類」により，要求工学者は，要求やゴール（望ましい状態）をその性質・特徴に合わせて整理する．た

☆4 詳しくは，http://en.wikipedia.org/wiki/Attack_tree を参照してほしい．

とえば，機密性，整合性，運用に関するもの，認証に関するもの，利便性に関するものなどセキュリティやその他の性質や機能で分類する．この分類により，要求間の関係を確認しつつ，その一貫性，整合性がレビューできるだけではなく，抜け･漏れなどの発見がしやすくなる．

この段階でシステムへの制約となる要求は，一般に，Web ベースアプリケーションなど，特定のシステムアーキテクチャがこの要求獲得プロセスに先立ちすでに決まっているときに発生する．たとえば，Web ベースアプリケーションでは，Web クライアントで実行されるプレゼンテーション層，アプリケーション層，データベース層の 3 層アーキテクチャで設計されることがあらかじめ決まっていることが多い．そのようなアーキテクチャ上の要求は，この段階で，その制約に関するリスクをあらかじめ評価しておくことができるという意味では都合がよい．たとえば，3 層アーキテクチャの場合，プレゼンテーション層が配置される Web クライアントの脆弱性は広く認識されているし，ネットワーク上の攻撃は，この 3 層の間の通信部分に発生するリスクが高いということがこの段階で評価可能となる．このステップでの要求の分類は，続く，ステップ 8 の優先付けを行う際の助けにもなる．

ステップ 8．要求に優先度をつける

「要求の優先度付け」は，それまでのステップに依存するだけではなく，そのためには，コストと利益の分析が必要となる．この損益の分析は，セキュリティのどの要求が，そのコストを払ってでも満たすべきかを決定するために必要となる [3]．自然言語で書かれた曖昧な要求に対して，一貫性や信頼性を持たせて優先度をつけるために，意思決定（Decision making）支援手法が有効である．たとえば，その 1 つである AHP（Analytic Hierarchy Process Methodology：階層分析法）を用いる場合，すべての要求のペアをつくり，それを相対的に価値とコストを比較し，要求間の半順序を定義していく．曖昧な項目に関して絶対的な評価基準（たとえば 5 段階評価）を定めようとしても，それを評価する人の主観的な意見がどうしても混入してしまう．この手法では，俗人性を排除するために，より客観的な評価が行える 2 項目間の相対比較しか行わない．すべての組合せに関して相対評価数がつけられた後，価値とコストの 2 次元グラフ上に要求をプロットすることにより，容易に価値が高くコストが低いグループ，価値が低いがコストが高いグループなどが区別でき，最終的な優先度を決定することが可能となる．

ステップ 9．要求を検査する

「要求の検査（inspection）」では，特定の決められた形式に従い，その妥当性をレビューする．その方法には，Fagan 流検査（Fagan Inspection：IBM の Fagan 氏が提唱する厳密な検査手法）からピアレビュー（Peer review：仲間同士のレビュー）まで，システムが求める要求の厳密性や複雑度に応じて，さまざまなレベルで行うことができる．一度この検査が通れば，その組織は，優先度付けがされたセキュリティに関する最初の要求集合を得ることとなる．この要求は，また，一部が不完全で，通常，後に再度検討すべきものが残っていることが多い．なぜならば，最初に列挙された要求は，粒度や抽象度がまちまちで，その間の一貫性を一度のプロセスで取ることは難しい．何度か開発プロセスを繰り返すことで，要求が徐々に洗練化することが重要になる．組織は，どの部分が特定のアーキテクチャや実装に依存するかを区別し，また，どれが再度検討されるべき項目かを理解し，設計のプロセスに入る必要がある．

SQUARE-Lite：軽量版 SQUARE

SQUARE の事例研究を通して，その実践は組織の一部に多くのコストがかかることが明らかになった．実験の結果，SQUARE の全プロセスをやり遂げるには，2，3 カ月まで及ぶこともあり，実際に多くの組織がそれだけの時間，コストを掛けることが難しかった．そのため，すでに要求工学の何らかのプロセスを実践している多くの組織でも，セキュリティの要求のためだけに完全に別ものと思われる SQUARE の要求獲得プロセスを最初から再度行おうとはしなかった．

この結果を踏まえ，CMU では，SQUARE のプロセスを見直し，どのステップが既存の要求工学のプロセスに適合するかを調査した．同時に，SQUARE の全プロセスに時間を費やしたくない組織に対しても，少なくとも何らかのメリットが感じられる簡素化したアプローチを模索していた．このような背景で，SQUARE プロセスから効果があると思われる 5 つのステップから構成される SQUARE-Lite が生み出された．SQUARE-Lite では，SQUARE 中の，ステップ 1，2，4，6 と 8 を行う．表 -1 の Lite の番号がそれらのステップに対応している．

しかしながら，最近の事例研究の結果，プロセスがきちんと整備されていない組織では，いくら軽量にしても SQUARE を実践する利益をほとんど得ることができないことが判明し，何らかの要求獲得プロセスを実践している組織にのみ SQUARE を利用するように勧めている．

これまでの活動と今後の展望

CMUでは，Cylabとの共同作業の中で，SQUAREのためのプロトタイプツールやワークショップ，チュートリアル，教育教材などを開発し，公開している．日本でのSQUAREに関する最初の講演は2008年6月に国立情報学研究所で行われた（詳細は，http://sse-project.org/を参照のこと）．さらに，SQUAREをサポートする現在のプロトタイプツールを，より強固にするような開発が続けられている．

SQUARE-Liteによって，コストをかけずに要求を獲得したい組織にも対応可能となった．しかし，個々の組織で実践しているプロセスに，どのようにSQUAREのステップを適合させるかを一意に定めるのは難しい．たとえば，これまでの事例研究の結果，要求の獲得・整理・優先度付けはシーケンシャルに行うよりも適宜同時に行ったり，スパイラルに行ったりするほうが効率がよいこと が分かっている．すなわち，SQUAREの中の重要なステップを抜き出すだけではなく，それらと既存のプロセスをどのように融合し，要求を洗練するかにはバリエーションがある．そこで，SQUAREの適用可能性についても研究は進んでいる．具体的には，SQUAREを標準の開発ライフサイクルプロセスに統合する方法は，技術報告書[4]で議論されている．また，今後も実践で用いられている既存のさまざまなライフサイクルプロセスへの統合に関しても引き続き研究する予定である．

既存のプロセスとの融合のみではなく，脆弱性・リスク分析手法や要求獲得手法などさまざまな要素技術に関して，SQUAREとの親和性を確認する必要がある．現状，各ステップで使える代表的な技術とその事例を示しているが，適切な技術を適切に使いこなすためのガイドラインとしては，まだ不十分である．さらに，事例研究を通して，技術の選択の判断基準，および，組み合わせる方法を明確にしていく必要がある．

現在，CMUでは，プライバシーを扱うことができる要求工学を研究・開発し，SQUAREのプロトタイプツールに統合している．そこでは，さらにプライバシーに関する要求獲得をより全面的にSQUAREに統合し，拡張する予定になっている．

参考文献

1) Mead, N. R., Hough, E. and Stehney, T. : Security Quality Requirements Engineering (SQUARE) Methodology (CMU/SEI-2005-TR009). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University (2005). http://www.sei.cmu.edu/publications/documents/05.reports/05tr009.html

2) Hubbard, R., Mead, N. and Schroeder, C. : An Assessment of the Relative Efficiency of a Facilitator-Driven Requirements Collection Process with Respect to the Conventional Interview Method, pp.178-186. Proceedings of the 4th International Conference on Requirements Engineering. Chicago, IL, pp.19-23 (June 2000). Los Alamitos, CA : IEEE Computer Society Press (June 2000).

3) Karlsson, J. and Ryan, K. : Cost-Value Approach for Prioritizing Requirements. IEEE Software 14, 5, pp.64-74 (Sep./Oct. 1997).

4) Mead, N. R., Viswanathan, V., Padmanabhan, D. and Raveendran, A. : Incorporating Security Quality Requirements Engineering (SQUARE) into Standard Life-Cycle Models (CMU/SEI-2008-TN-006). Pittsburgh, PA : Software Engineering Institute, Carnegie Mellon University (2008). http://www.sei.cmu.edu/publications/documents/08.reports/08tn006.html

5) Wood, J. and Silver, D. : Joint Application Design : How to Design Quality Systems in 40% Less Time. New York, NY : John Wiley & Sons (1989).

（平成21年2月6日受付）

Nancy R. Mead ▶nrm@sei.cmu.edu

カーネギーメロン大学 Software Engineering Institute（SEI）．Computer Emergency Response Team（CERT）シニアメンバ．カーネギーメロン大学 the Master of Software Engineering，および，the Master of Information Systems Management program 兼務．IEEE，および，IEEE Computer Society フェロー．ACM 会員．

吉岡 信和（正会員） ▶nobukazu@nii.ac.jp

1998年北陸先端科学技術大学院大学情報科学研究科博士後期課程修了．博士（情報科学）．同年（株）東芝入社．2002年より国立情報学研究所に勤務，2004年より同研究所 特任助教授，現在准教授．エージェント技術の研究，ソフトウェア工学の研究に従事．日本ソフトウェア科学会，電子情報通信学会各会員．