Software [In]security: The Building Security In Maturity Model (BSIMM)

informIT

By Gary McGraw,Brian Chess,Sammy Migues

Date: Mar 16, 2009

Article is provided courtesy of Addison-Wesley Professional.

Return to the article

Gary McGraw, Brian Chess, and Sammy Migues describe the genesis of the Building Security In Maturity Model, its foundation in real world data, and the benefits of using it as an empirical yardstick for measuring your own software security initiative.

Confessions of a Software Security Alchemist

On March 4th we released the Building Security In Maturity Model (BSIMM) under a Creative Commons license (and slightly ahead of schedule). Those of you who follow this column know that we built the BSIMM by gathering real data from nine large-scale software security initiatives. Seven of the nine companies we studied graciously agreed to let us identify them: Adobe, The Depository Trust & Clearing Corporation (DTCC), EMC, Google, Microsoft, QUALCOMM, and Wells Fargo. We could not have done this empirical work without the cooperation of the nine, including the "two who cannot be named."

The BSIMM project adhered to one hard and fast scientific rule: only activities observed in the field could be added to the model. We started with a software security framework and a blank slate. As a result, BSIMM is the world's first software security yardstick based entirely on real world data and observed activities. Whether you run a software security initiative today or are charged with starting one tomorrow, you are likely to find the BSIMM incredibly useful.

Empiricism Over Alchemy

A handful of millennia ago — say, around 400 BCE — a number of particularly inquisitive souls spent much of their time working on alchemy. Some historians of science argue that alchemy evolved into chemistry. (The term "evolved" might be a bit of an overstatement. McGraw's dad was a chemist, and he claimed to "make potions" for a living.) Though the elusive lead-into-gold recipe remains out of reach, empirically-grounded chemistry has served the modern world well. The time has come for software security to make the same shift away from alchemy towards empiricism.

Early work in software security, including our own, concerned itself with advocacy and evangelism. We needed to convince the world that we had a serious problem. We succeeded.

In 2006, software security found itself embodied in three major methodologies: Microsoft SDL, Cigital Touchpoints, and OWASP CLASP. Not surprisingly, these three methodologies are very much like religions — charismatic leaders, articles of faith, and some basic best practices. If you stand back and squint, the three software security religions look basically the same.

Both early phases of software security made use of any sort of argument or "evidence" to bolster the software security message, and that was fine given the starting point. We had lots of examples, plenty of good intuition, and the best of intentions. But now the time has come to put away the bug parade boogeyman, the top 25 tea leaves, black box web app goat sacrifice, and the occult reading of pen testing entrails. The time for science is upon us.

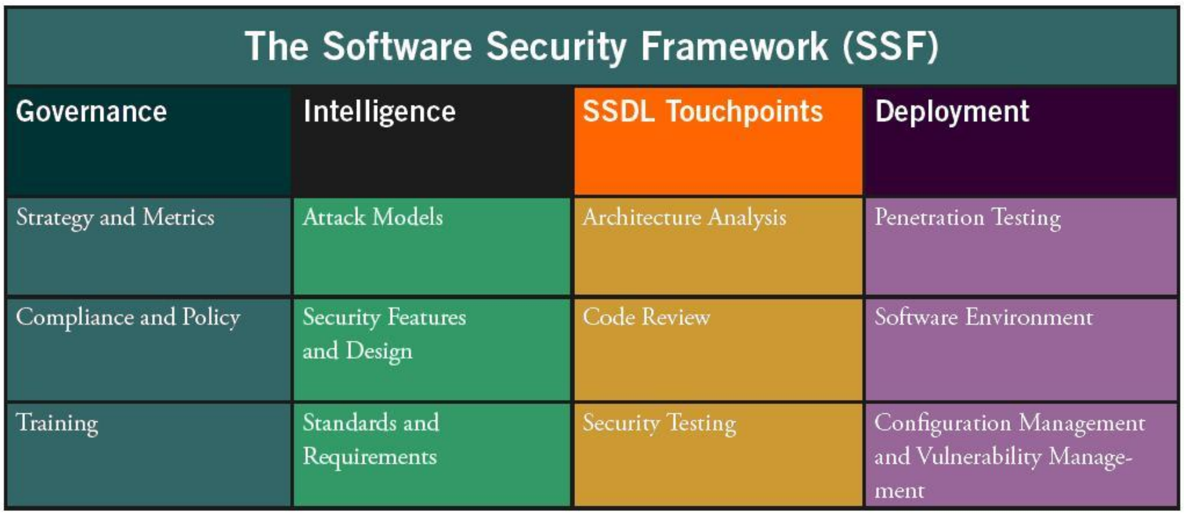
We are aware of at least 35 software security initiatives underway. We chose to study nine. Our model is based on empirical evidence and data observation. Each of the 110 activities we identified was observed in the field. The example for each activity in BSIMM is a real example. The 110 activities are not "best practices;" they are actual practices. Science.

BSIMM Basics

To give you a taste of how the BSIMM is constructed, we'll dive down through the model from the highest level. We introduce the Software Security Framework (SSF) in an informIT article where you can read more.

Incidentally, judicious use of the BSIMM requires careful use of the SSF to structure the data-gathering interview. Simply waltzing down the list of 110 activities and asking, "do you do such and so?" is not an appropriate data-gathering mode. In our early work applying the BSIMM, we have already noticed a "Soviet revisionist history" problem with self-reporting that we will need to account for as the model evolves. "Hey, that's a good idea!" becomes "Didn't we try that once?" becomes "We've totally been doing that forever." One way to avoid hyperbolic reporting issues is to structure the interview properly by discussing all of an organization's SSG activities in context, and then measuring the knowledge gained with the BSIMM yardstick. Another is to have the interview carried out by an experienced BSIMM interviewer.

Here is the SSF:



The Software Security Framework (SSF)

Governance	Intelligence	SSDL Touchpoints	Deployment
Strategy and Metrics	Attack Models	Architecture Analysis	Penetration Testing
Compliance and Policy	Security Features and Design	Code Review	Software Environment
Training	Standards and Requirements	Security Testing	Configuration Management and Vulnerability Management

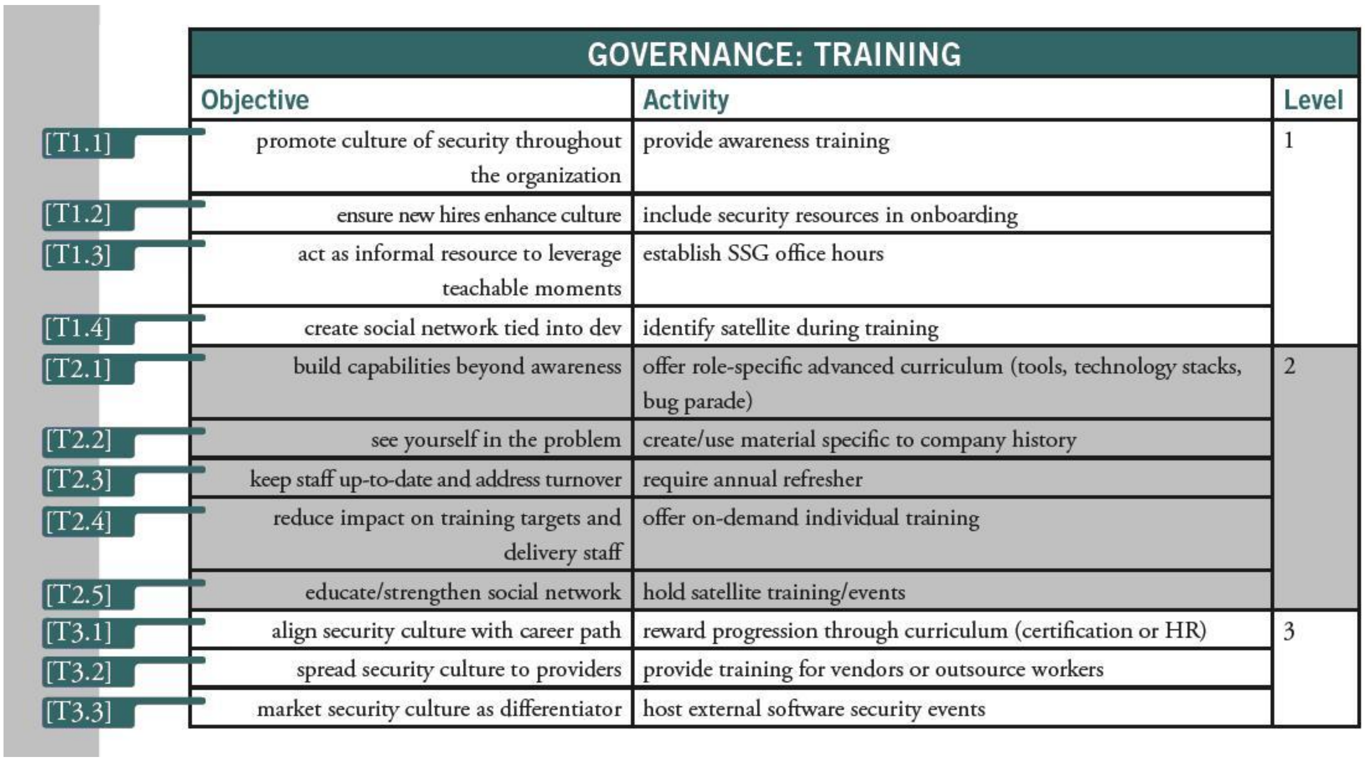
Figure 1.

Each of the twelve practices in the SSF has an associated set of activities described in the BSIMM. There are 110 activities. The BSIMM document provides a description of each activity, including *real examples* of each activity as observed among the nine. Special note: all 110 activities are actually at work somewhere (even though we don't expect all 110 to be employed in any single organization)!

The BSIMM website also has an interactive SSF. You can explore the practices and the activities by clicking around.

To make this especially clear, we did NOT observe all 110 activities in all of the nine. In fact, only ten activities were observed in all nine initiatives. However, we DID observe each of the 110 activities in at least one of the nine (and in most cases, more than one). In that sense, even though we stuck to our data-driven guns, the BSIMM is an inclusive model.

For each of the twelve practices, we have constructed a "skeleton" view of the activities divided into three levels. As an example, the skeleton for the *Training* practice is shown below:



GOVERNANCE: TRAINING			
	Objective	**Activity**	**Level**
[T1.1]	promote culture of security throughout the organization	provide awareness training	1
[T1.2]	ensure new hires enhance culture	include security resources in onboarding	
[T1.3]	act as informal resource to leverage teachable moments	establish SSG office hours	
[T1.4]	create social network tied into dev	identify satellite during training	
[T2.1]	build capabilities beyond awareness	offer role-specific advanced curriculum (tools, technology stacks, bug parade)	2
[T2.2]	see yourself in the problem	create/use material specific to company history	
[T2.3]	keep staff up-to-date and address turnover	require annual refresher	
[T2.4]	reduce impact on training targets and delivery staff	offer on-demand individual training	
[T2.5]	educate/strengthen social network	hold satellite training/events	
[T3.1]	align security culture with career path	reward progression through curriculum (certification or HR)	3
[T3.2]	spread security culture to providers	provide training for vendors or outsource workers	
[T3.3]	market security culture as differentiator	host external software security events	

Figure 2.

As you can see, there are eleven activities under the *Training* practice divided into three levels. Levels roughly correspond to maturity in that practice. Regarding levels, it is not necessary to carry out all activities in level 1 before moving on to level 2 or level 3 activities; however, the BSIMM levels correspond to a logical progression from "straightforward" to "advanced."

In the body of the BSIMM (and on the website) is a paragraph describing each activity. Here is an example of the description for Activity T1.3:

[T1.3] Establish SSG office hours.

The SSG offers help to any and all comers during an advertised lab period or regularly scheduled office hours. By acting as an informal resource for people who want to solve security problems, the SSG leverages teachable moments and emphasizes the carrot over the stick. Office hours might be held one afternoon per week in the office of a senior SSG member.

On page 49 of the BSIMM model, we report the number of times each of the 110 activities were observed in the field among the nine. By referring to that chart, we can note that five of the organizations we studied perform this activity.

Software Security 2.0 — An Empirical Yardstick

The most obvious way to use the BSIMM is as a yardstick for measuring your own software security initiative. You can do this by noting which of the activities you routinely carry out and which you don't, and then thinking about the level of maturity attained by your organization as evidenced by the level of the activities you carry out.

We are collecting a set of data that is growing over time with a 110-bit activity vector for each organization. Using these data, it is possible to determine where you stand and how what you are doing compared to others.

In the BSIMM we released two analyses that are useful for this comparison. The first is the average over the nine, which shows a "high water mark" maturity level for each of the twelve practices averaged over the nine. That is, if a level three activity was observed in a practice that practice is noted as a three regardless of the number of activities in levels one and two. We sliced the data many ways, and the high-water mark turned out to be both useful and easy to calculate.

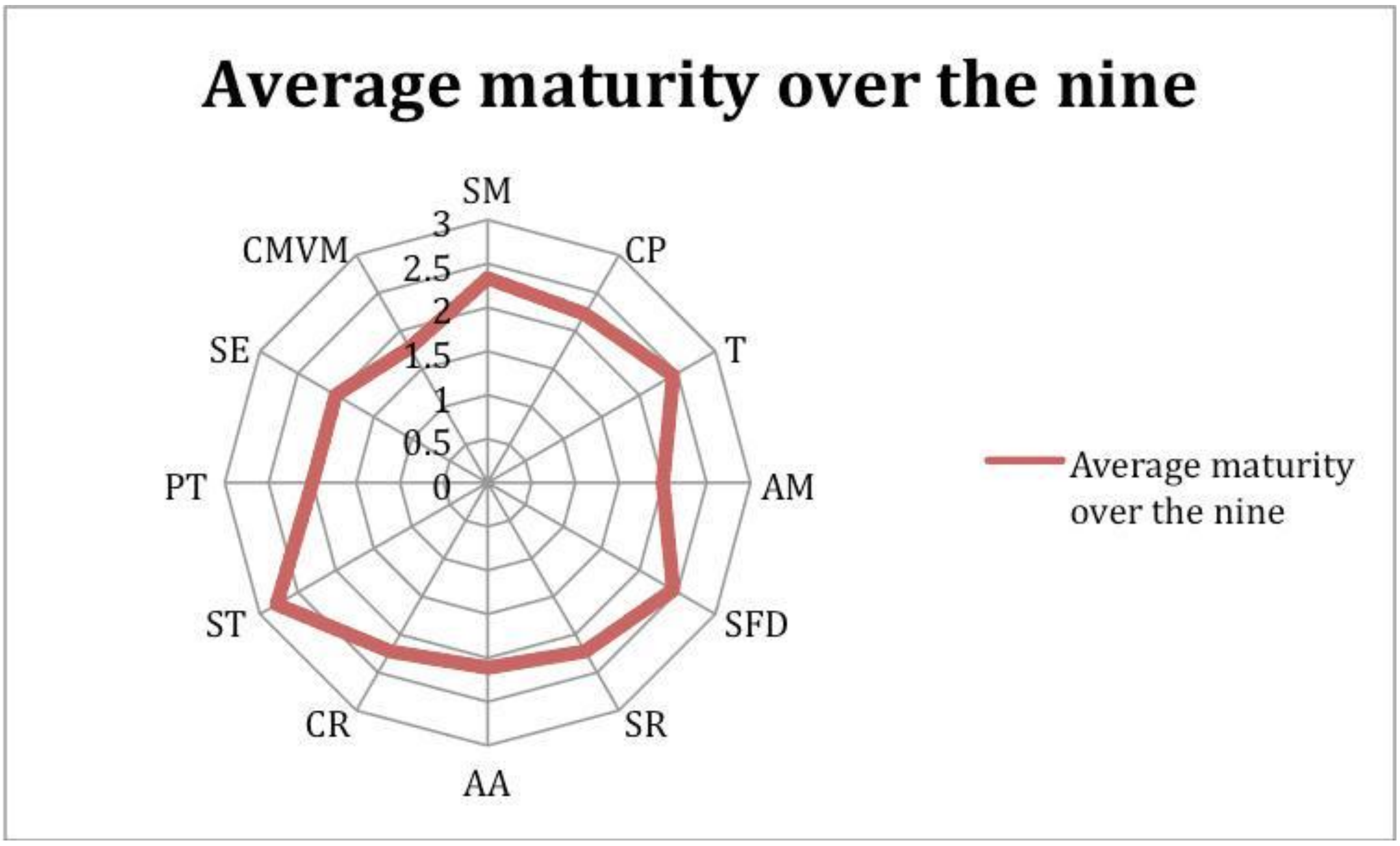
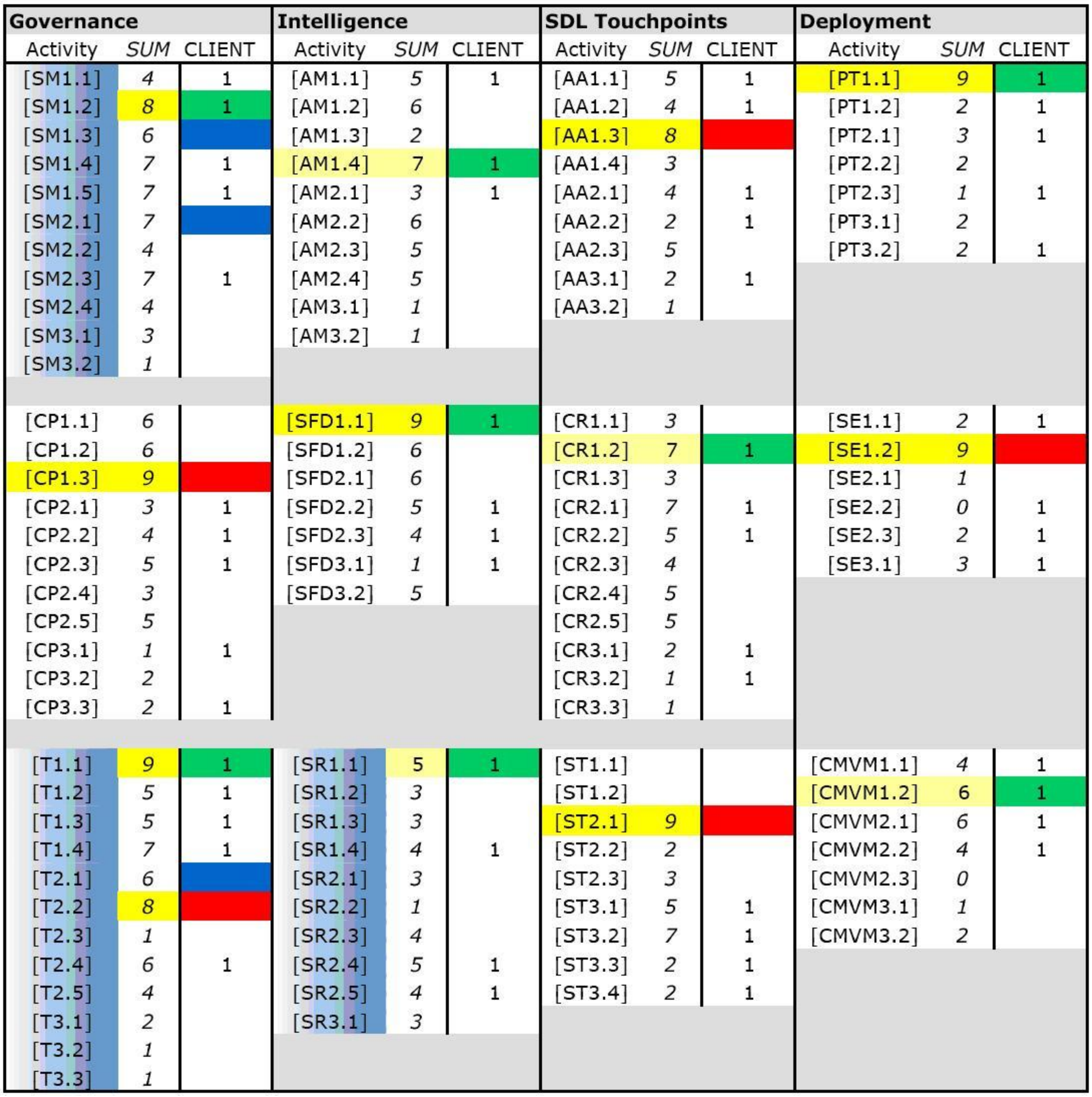


Figure 3.

By computing your own high water mark score and laying it over the graph above, you can quickly determine where you may be ahead of the game and where you may be behind. This information becomes valuable when you switch from yardstick-mode to software security initiative planning-mode, adopting some of the BSIMM activities based on your local goals, your assessment of software security risks, and your organization's culture.

A deeper analysis is also possible and is shown here (special note, the client data are FAKE):



Governance			Intelligence			SDL Touchpoints			Deployment		
Activity	SUM	CLIENT	Activity	SUM	CLIENT	Activity	SUM	CLIENT	Activity	SUM	CLIENT
[SM1.1]	4	1	[AM1.1]	5	1	[AA1.1]	5	1	[PT1.1]	9	🟩 1
[SM1.2]	8	🟩 1	[AM1.2]	6		[AA1.2]	4	1	[PT1.2]	2	1
[SM1.3]	6	🟦	[AM1.3]	2		[AA1.3]	8	🟥	[PT2.1]	3	1
[SM1.4]	7	1	[AM1.4]	7	🟩 1	[AA1.4]	3		[PT2.2]	2	
[SM1.5]	7	1	[AM2.1]	3	1	[AA2.1]	4	1	[PT2.3]	1	1
[SM2.1]	7	🟦	[AM2.2]	6		[AA2.2]	2	1	[PT3.1]	2	
[SM2.2]	4		[AM2.3]	5		[AA2.3]	5		[PT3.2]	2	1
[SM2.3]	7	1	[AM2.4]	5		[AA3.1]	2	1			
[SM2.4]	4		[AM3.1]	1		[AA3.2]	1				
[SM3.1]	3		[AM3.2]	1							
[SM3.2]	1										
[CP1.1]	6		[SFD1.1]	9	🟩 1	[CR1.1]	3		[SE1.1]	2	1
[CP1.2]	6		[SFD1.2]	6		[CR1.2]	7	🟩 1	[SE1.2]	9	🟥
[CP1.3]	9	🟥	[SFD2.1]	6		[CR1.3]	3		[SE2.1]	1	
[CP2.1]	3	1	[SFD2.2]	5	1	[CR2.1]	7	1	[SE2.2]	0	1
[CP2.2]	4	1	[SFD2.3]	4	1	[CR2.2]	5	1	[SE2.3]	2	1
[CP2.3]	5	1	[SFD3.1]	1	1	[CR2.3]	4		[SE3.1]	3	1
[CP2.4]	3		[SFD3.2]	5		[CR2.4]	5				
[CP2.5]	5					[CR2.5]	5				
[CP3.1]	1	1				[CR3.1]	2	1			
[CP3.2]	2					[CR3.2]	1	1			
[CP3.3]	2	1				[CR3.3]	1				
[T1.1]	9	🟩 1	[SR1.1]	5	🟩 1	[ST1.1]			[CMVM1.1]	4	1
[T1.2]	5	1	[SR1.2]	3		[ST1.2]			[CMVM1.2]	6	🟩 1
[T1.3]	5	1	[SR1.3]	3		[ST2.1]	9	🟥	[CMVM2.1]	6	1
[T1.4]	7	1	[SR1.4]	4	1	[ST2.2]	2		[CMVM2.2]	4	1
[T2.1]	6	🟦	[SR2.1]	3		[ST2.3]	3		[CMVM2.3]	0	
[T2.2]	8	🟥	[SR2.2]	1		[ST3.1]	5	1	[CMVM3.1]	1	
[T2.3]	1		[SR2.3]	4		[ST3.2]	7	1	[CMVM3.2]	2	
[T2.4]	6	1	[SR2.4]	5	1	[ST3.3]	2	1			
[T2.5]	4		[SR2.5]	4	1	[ST3.4]	2	1			
[T3.1]	2		[SR3.1]	3							
[T3.2]	1										
[T3.3]	1										

Figure 4.

The table above is organized to roughly mirror the SSF. The SUM column shows how many of the nine performed each activity (from BSIMM page 49). Those activities where the organization carries out one of the activities that everybody does, are shown as green blocks. Those activities where the organization does not do one of the things that everybody does are shown in red blocks. Those practices where the data show that the organization under review is "behind" the average are shown with the blue swath over the activities in the practice.

In the fake client data above, the pretend organization should think hard about the red blocks in *compliance and policy*, *training*, *architecture analysis*, *security testing*, and *software environment*. Red doesn't mean you're negligent, but it does make you an outlier. Like your mother always told you, just because everyone else is doing it doesn't mean you have to do it too, but it is prudent at least to know why you're not doing it.

Blue shift practices for this pretend organization include: *strategy and metrics*, *training*, and *security requirements*. Those "popular" activities in blue shift practices (where more than five of the nine carry out the activity) are shown as blue blocks. The blue shift activities might be a way to accelerate a program quickly within a given practice.

Everyone is a Special Snowflake (Not)

One of the surprises of the BSIMM work was that industry vertical has less impact than we thought it would on a software security initiative. We observed data from 4 ISVs (Microsoft, EMC, Google, and Adobe). We also observed data from four financial services firms (Wells Fargo, DTCC, and two that remain un-named). Our intuition was that we would need to build two models, one for ISVs and one for Financial Services firms. The data show that intuition is wrong.

Here is a spider chart showing the ISV high water mark average over the Financial Services high water mark average (these data are real):

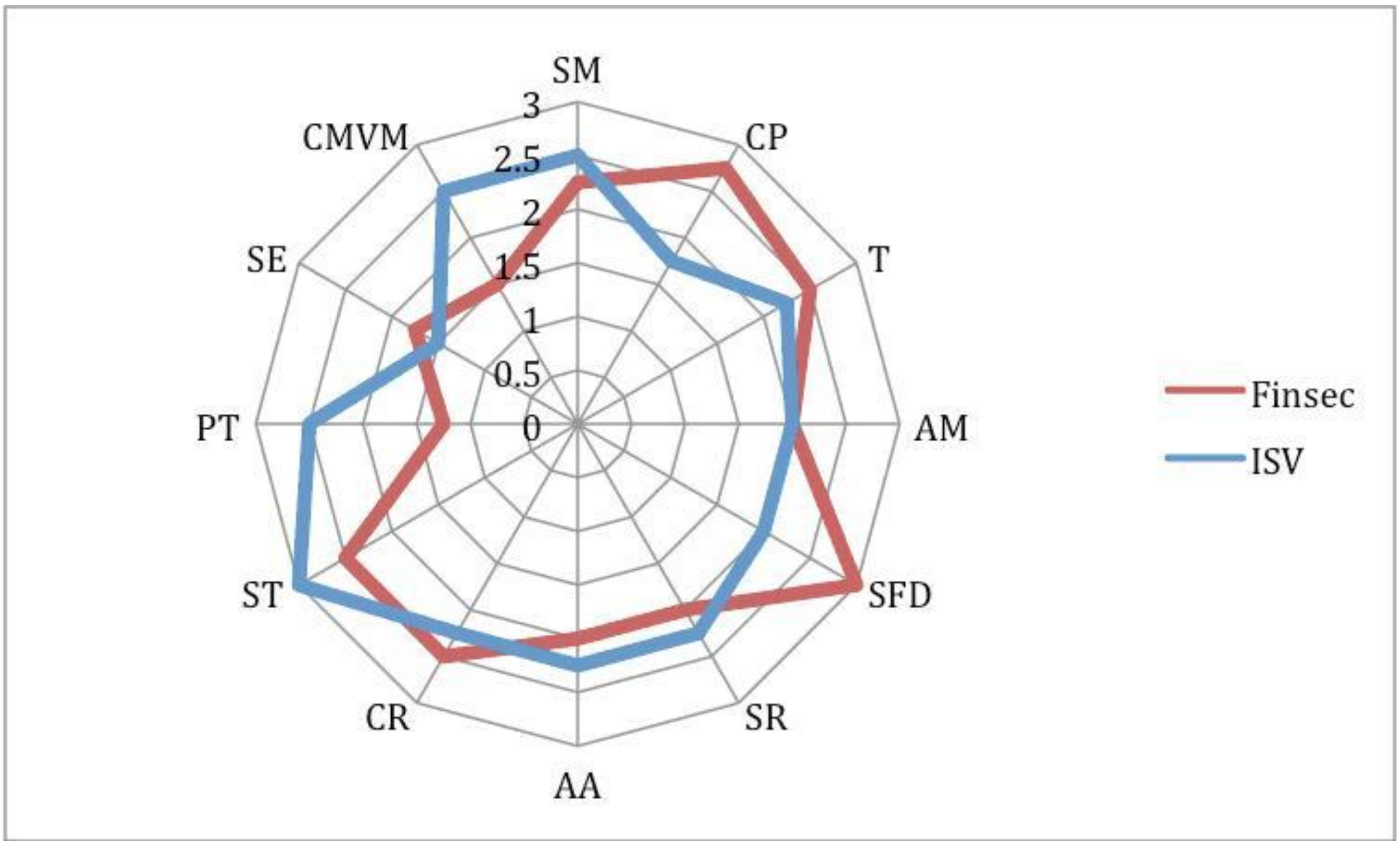


Figure 5.

As you can see, the most apparent feature is the broad overlap between verticals. There is one difference worth noting: the ISVs studied have a tendency to emphasize testing (both *security testing* and *pen testing*) and *configuration management/vulnerability management* over the financial services firms. The financial services firms have a slightly greater emphasis on *compliance and policy*, *security features and design*, and *training*. This makes perfect sense if you think about regulatory compliance versus market perception issues affecting the two verticals.

The data show that the BSIMM is useful as a yardstick for software security initiatives even from diverse verticals. As we gather more data, both from larger, established programs and from smaller, newer programs, we will determine just how robust this feature of the model is in practice.

This is an important result, because it cuts through one of the software security myths: the myth of the special snowflake. Jim Routh, one of the study participants and the CISO of DTCC puts it this way, "What we discovered is that a mature software security program for a financial service firms is very similar to one for an ISV like [Microsoft]. In fact they can and do use the same methods, techniques and tools for secure software development. The maturity for both entities can now be measured the same way with the same framework. This helps our firm when it communicates to ISVs and requests artifacts from the software security program. We always assumed that the ISV world of software security was different."

Put concisely, the BSIMM is an extremely useful yardstick for software security programs that can provide an important measuring tool.

The Metrics/Culture Problem

We spoke to each of the nine about metrics, and previously reported some of our findings regarding the misuse of metrics. All of the nine have robust metrics systems in place and note the importance of good metrics to their success (and decry the distracting impact of bad metrics). Despite the common ground seen in the various SSG activities, we observed no common metrics shared among all of the nine.

Our hypothesis is that metrics are valuable in a particular organizational culture. Furthermore, transferring a metrics program from one culture to another seems much like organ transplant — chances of rejection by the host are high.

This is discouraging, because we all would like some common metrics that work for any organization, especially metrics that we can use to show that software security is actually improving. Meanwhile, we will settle for measuring activities under the assumption that such activities result in more secure software. The participants firmly believe that their SSG activities are a fundamental reason for their software security improvements, and we have no reason to doubt them.

Dawn in the Age of Enlightenment

Please download the BSIMM and use it yourself: http://bsi-mm.com. Use the contact form to tell us about your experience.