IT Security in the brave new world of Agile and DevOps

Abstract

The advent of Agile and DevOps practices in the development of IT systems is changing not only the pace at which new features are deployed but also the way of managing in practice IT security and controls. Together with new opportunities, Agile and DevOps bring to IT security management and governance new risks and the requirement to find new continuous and adaptive ways to provide to the business the security of IT systems.

The security of IT systems is strictly tied to two major features of the IT systems themselves:

- the presence or absence of IT security vulnerabilities
- the correct design and deployment of IT security measures

and both of them depend on the development process of IT systems.

Indeed IT security vulnerabilities can be avoided by carefully designing and implementing the IT systems with good security practices, and IT security measures can be deployed only if the IT systems are designed and implemented with features that support them.

If on one side it is well known that the success of IT security is strictly tied to the development of the IT systems, at the same time the process of developing an IT system in a secure way is seldom successful, as we all very well know. It is enough to read specialized magazines, blog and even normal newspapers to learn about security bugs, attacks, incidents etc., due to the failure of security in some IT system or product.

The attention to security by the IT industry has greatly improved in the last 10 years, but at the same time IT systems and products have become much more complex and attackers have become much more skilled besides being motivated by financial gain and crime.

As of today, Cyber Crime is the 2nd most reported economic crime affecting businesses worldwide [1] and the cost to worldwide business of Cyber Crime is estimated between 400 Billion USD and 2

Trillion per year [2,3].

This is an obvious indication that building security in IT systems and products is not at all easy. To understand why we should briefly look into the evolution of the processes and techniques to develop IT systems.

SDLC and Waterfall

The Systems Development Life Cycle (SDLC) is the traditional approach to the management, engineering processes and techniques used to develop IT systems.

Going back 30 or more years, the high level process to develop an IT system was based on these steps:

- definition of the requirements, in particular of the features requested by the final user
- design of the technical specifications to implement the requirements
- development of the IT system, in particular of the software
- test of the final product to verify that it satisfies all the initial requirements
- sale or deploy in production of the final product.

In the traditional approach, usually going under the name of “Waterfall”, these steps are executed sequentially, one after the other. This makes sense if the product to build has few, carefully selected and well designed features, and at the end of the development cycle it should be carefully well tested to assure that the product sold in tapes, discs or other types of hardware does not have defects. Moreover the product was usually manufactured at one time in large numbers and then sold for a long period of time, even of a few years.

This approach worked well 30 and more years ago, but in the '90s things started to change in two respects:

- IT products, in particular software, became very large with an extremely high number of features
- the development cycle became much shorter, and new features were requested by business (and customers) at an increasing speed.

These two points are practically at contradiction since the larger becomes the product and the number of features, the longer is the development time and not vice-versa. Indeed the tension between these two points was a major contributing factor to the monumental failures of some IT projects in particular in the '90s and 2000's [4].

The Manufacturing Way

But these problems are well known in the manufacturing sector. Since the time of Henry Ford and the introduction of the Assembly Line, manufacturing has been developing approaches to increase production, reduce waste, reduce production time and improve quality at the same time. Many approaches and methodologies have been proposed and adopted in manufacturing to improve the productivity. Some of these approaches and methodologies are (in alphabetical order): Drum-Buffer-Rope (DBR), Just In Time (JIT), Kanban, Lean Manufacturing, Six Sigmas, Theory of Constraints, Toyota Way (or Toyota Production System – TPS), etc.

IT systems are quite different from a manufacturing assembly line, but many ideas, principles, approaches proposed and adopted in manufacturing companies, have been adapted to the development of IT systems with varying degrees of success.

Agile Software Development

The first changes proposed and by now extensively adopted, concern the approach and methodologies for software development. These approaches go under the general name of Agile Software Development [5]. The main purposes of these approaches and methodologies are two: to develop applications in much shorter time and to be able to be closer and more reactive to the always changing business requirements. This can be achieved by:

- having short developing cycles (or "sprints"), days or weeks instead of months or years
- producing few features at the time, instead of a full product
- working in small groups with very good internal communication, also with the final users
- adopting as much as possible tools to automate large part of the developing process: tools for writing, building, testing etc. the software.

This implies that software products are created in an incremental and evolutionary way so to respond quickly to the business requests.

And this has the obvious drawback that requirements keep changing, and often it is difficult to have a detailed, clear, full description of the product.

The DevOps Approach

Traditionally development and operations (that is the daily running of production IT systems) have

been separated groups. New IT systems produced by the development teams would be installed in production under the care of the operations teams, by means of Change Management which is a critical and often long procedure. The development and operations teams often have very little contacts, and usually development personnel is completely barred from accessing systems in production.

But the traditional Change Management processes cannot keep up the pace if the development teams release new applications every few days or even every few hours or minutes.

Moreover, changing very often IT systems is very risky: stability is fundamental for the correct daily operation of IT systems. Many small and frequent changes can destabilize the IT systems and can lead to very important incidents, with loss of availability, security, compliance, integrity, confidentiality and data.

But business requires that the IT systems change at the business speed, not at, from a business point of view, an "artificial" speed imposed by organizational and technical requirements of the IT department.

To be able to release in production new software even every few minutes,[1] development and operations must collaborate closely and the Change Management process must be redesigned and rendered as much as possible automatic.

The principles and approaches to adopt are similar to the Agile ones for software development, but the implementation techniques are more similar to what is done in manufacturing and assembly lines. This is what the DevOps movement and approach stand for [6,7].

What about IT Security?

Agile and DevOps bring both new opportunities and new risks to IT security.

The new approaches to the development and maintenance of IT systems brought by Agile and DevOps offer various opportunities to improve IT security.

First of all, a much more frequent maintenance of IT systems with updates and upgrades of hardware and software, is a inevitable consequence of adopting these approaches. From the point of IT security, the first security defence is to keep all software up-to-date and this will come as a by-product of Agile and DevOps. IT security practitioners must exploit this opportunity to guarantee that all operating systems and applications receive in short time all security patches.

Having a dynamic environment as the one provided by Agile and DevOps allows to deploy new

1 John Jenkins, Amazon's former lead engineer, in 2013 went on record saying that in May 2011 Amazon pushed new code to production about every 11.7 seconds during the work week [8].

security features in short time. Indeed, by treating security features in the same way as business features, it is possible to continuously improve the security of the IT systems in an incremental and evolutionary way. Again IT security practitioners should see that the release of new or updated security features becomes a normal routine, included in the daily or hourly release in production of new software.

Moreover Agile and DevOps make it much easier to respond to and manage in very short time IT security incidents. These methodologies allow to deploy critical security patches and critical security fixes in production in a controlled way using standard procedures instead of emergency, ad-hoc ones. This makes the management of security incidents a much less risky procedure removing many of the uncertainties of the traditional IT emergency interventions.

In general, the adoption of these new approaches which make more dynamic and open to change the development and deployment of IT systems, is an opportunity to improve in a continuous way the security of the IT systems and to make the management of IT security a necessary component of every IT system.

New and Returning Risks for IT Security

As far as there are many opportunities to improve security of IT systems with the adoption of Agile and DevOps, there are also many new and returning risks.[2] Some of these risks are actually those that the traditional approach to IT system development mitigates. And some of them have already been mentioned in the previous sections.

First of all, in such dynamic environments as the ones created by Agile and DevOps, it becomes a difficult task to have a full, detailed and clear documentation of the IT systems, in particular of software applications.

In the traditional approach, the documentation is critical since development and most analysis, checks, audit are done only based on it. Documentation comes first, and the IT system must implement and adhere strictly to it. This gives a formal chain of control which should guarantee the quality, compliance and security of the product. In Agile and DevOps the relation between documentation and product is often inverted: the documentation describes what has been implemented based on the requests of the user.

The first very real risk is that to cut down time even more, the documentation is never prepared or describes the IT system at such a high level that it is not useful to verify quality, compliance and security. Procedures and tools must be adopted to guarantee that the relevant and adequate documentation is produced along side the IT product, and that the documentation is always

2 See [9, 10, 11, 12] for approaches to manage Agile and DevOps risks and controls.

maintained up to date. In practice this requires to automate most of the effort to produce documentation, which in large part is a catalogue of components (down to the smallest ones) and library of changes (described by what, by whom, how, when etc. a change is implemented).

Once the evolution of the IT systems is strictly traced and controlled, and a clear, detailed description of the IT systems in production is available, it is possible to evaluate quality, compliance and security. Moreover it is possible to introduce automated controls that guarantee that some security (or quality or compliance) measures are always in place and are active. This to prevent that new features will remove, dis-activate or render inefficient controls already in place.

DevOps re-designs the Change Management process to assure that it can deliver at the speed and in collaboration with the Agile development of new IT systems. In the traditional approach, Change Management is the moment where all crucial tests, including the quality, compliance and security ones, are performed or verified. By automating most of the Change Management process there is a very large risk that many quality, compliance and security tests are not performed. Indeed many of these tests, as usually formulated, are thought to be performed by humans, not machines. Eliminating most human intervention from this process, often implies also eliminating these tests.

This can have catastrophic consequences for integrity, confidentiality, availability, compliance etc. of the IT systems. But integrating controls and DevOps requires a change in the approach to IT security controls (and to IT audit). Traditionally controls are introduced in the requirements' documentation and checked in the test phase of development and in the Change Management process. Since these phases are dramatically modified in the Agile and DevOps approach, these controls must be introduced and verified differently.

Probably the best approach is to manage these controls as features of the IT systems, and develop and deploy them as any other IT feature. IT Security, as IT Quality, can be seen as another “user” of the IT systems with its own business requirements. In case, the unusual role of this “user” is that it is involved in all IT products. In the design of the security features must be also included the automated controls which guarantee that security is always active when the product is delivered in production.

Viewing IT security as another “user” of IT systems development, requires that developers and operations personnel get acquainted with the “security business”. In other words, all developers and operations personnel must understand at least the basic IT security principles, how to respect them and implement them in their own field of expertise. This can be done with an ongoing IT security awareness program, which must be a continuous and dynamic process, as Agile and DevOps are.

Moreover, in such dynamic environments as the ones created by Agile and DevOps, even if all possible processes to govern IT security have been implemented, there is always the risk that something will go unnoticed. It is then necessary that IT security tests, monitors and scans continuously IT systems for vulnerabilities and security weaknesses. Typical activities which should

be automated as much as possible and performed on a regular, daily basis, are vulnerability assessments, penetration tests, security code reviews etc. These should obviously be done on the production environment but also and more importantly, on the development and test environments so to catch possible IT security issues as early as possible.

Conclusions

Agile and DevOps change the development processes of IT systems to sustain the pace of current business introducing dynamism and elasticity to their evolution.

This opens many interesting opportunities for IT security, as well as for quality and compliance, but at the same time requires some drastic change to manage new and returning risks.

One of the main issues, at the same time a strength and a potential weakness, is the need of collaboration and communication. IT security awareness must become a central pillar to improve the security of IT systems, since everybody involved with the fast development and change of IT systems must be at least aware of the major IT security risks.

At the same time the approach of IT security practitioners should evolve: IT security should become incorporated into business features and be designed, developed and deployed together or similarly to real business features. IT security practitioners should be involved at every moment of the development of an IT system, and monitor it while in production. This should become a continuous, cyclical activity, mostly supported by tools and by the applications themselves,[3] instead of the traditional approach of writing requirements' documentation and testing products before the release in production or during yearly reviews.

References

[1] PwC, “Global Economic Crime Survey”

[2] McAfee, “Economic Impact of Cyber Crime II, 2014”

[3] Forbes, “Cyber Crime Costs Projected To Reach $2 Trillion by 2019”

[4] for a review, see the IEEE Spectrum series of articles on “Lessons From a Decade of IT Failures”, http://spectrum.ieee.org/static/lessons-from-a-decade-of-it-failures

[5] “Manifesto for Agile Software Development”, http://www.agilemanifesto.org/

[6] For a definition of “DevOps” see Patrick Debois "Agile 2008 Toronto"

---

3 See for example [12] for a list of 10 critical controls in Agile and DevOps environments.

http://www.jedi.be/blog/2008/10/09/agile-2008-toronto-agile-infrastructure-and-operations-presentation/ , and “Ghent 2009” http://www.devopsdays.org/events/2009-ghent/

[7] For a brief introduction to DevOps see for example Mike Loukides, “What is DevOps” and “Revisiting What is DevOps”, O’Reilly

[8] G. Lawton, “How Amazon Made the Leap to a DevOps Culture,” ServiceVirtualization.com, 5 September 2013, http://servicevirtualization.com/how-amazon-made-the-leap-to-a-devops-culture/

[9] Chong Ee, “Auditing Agile – A Brave New World”, Isaca Journal 2, 2016

[10] Alan Moran, “Risk Management in Agile Projects”, Isaca Journal 2, 2016

[11] “DevOps overview”, Isaca white paper 2015

[12] “DevOps practitioner considerations”, Isaca white paper 2015

Andrea Pasquinucci (PhD CISA CISSP)