ComputerWeekly.com TechTarget

E-guide

CW+ Content

How to achieve container security best practice

In this e-guide

In this e-guide:

The adoption of containers in the enterprise promises great organisational efficiency advantages, but as with any fast-evolving technology, their implementation brings new challenges to cyber security teams for a number of reasons.

In a recent series of articles, Computer Weekly's long-running Security Think Tank assessed some of the issues around this tricky problem and sought to answer the question, what do CISOs need to know to secure containers?

In this e-guide, we will explore some of their thoughts. First, PA Consulting experts Alan Taberham and Niall Quinn set out their ideas on what container security best practice looks like, while the British Computer Society's (BCS's) Petra Wenham weighs in on how the evolution of container security means CISOs must evolve their thinking, too.

Then, Paddy Francis of Airbus Cyber Security outlines how good container security practice begins with good DevOps

practice, while Paul Holland of the Information Security Forum (ISF) argues in favour of embedding security-by-design as early as possible in the development process, and Turnkey Consulting's Andrew Morris shows us why with a little forethought, securing containers need not be too taxing.

Finally, we go in-depth on the relationship between zero-trust cyber security models and containers, calling on multiple experts to find out why how to apply zero-trust models to containers, and why the two make excellent bedfellows.

Alex Scroxton, Security Editor

In this e-guide

Security Think Tank: Four steps to container security best practice

Alan Taberham and Niall Quinn

Container concepts began in Linux systems and were made mainstream by Docker in 2013, which launched containerisation into the global developer community. Advances in the orchestration layer continue to mature and broaden container capabilities – especially within the hyperscale cloud supplier platforms and microservice architecture, such as Netflix or Paypal.

For CISOs looking to help their business safely adopt or continue to use this technology, they must ensure they are equipped to deal with the threats and risks they present. The resultant growth in the complexity and size of IT estate is not unique to containers, but there are four areas a CISO should be considering:

1. Ensure a code pipeline mentality within the security team, using DevSecOps to keep pace and avoid being overwhelmed with manual rebuilds

Patching a containerised application, external dependencies and the application code, requires an update to the base image and a recreation and redeployment of the container. Maintaining the implementation of updates is critical and ensuring security experts are part of your developer teams is key to staying on top of this challenge.

As with any DevSecOps pipeline, you should also take precautions around leaking hard-coded credentials which are embedded within the container images, scanning for vulnerabilities and determining the level of trust in the dependencies packaged with the software. All these activities that help improve the detection of vulnerabilities save the organisation money. Also, don't forget that in order to patch, you need to be able to replace, stop and restart a container.

2. Implement configuration management and security tools that can cope with the scale

Effective configuration management is crucial. Orchestration services (Kubernetes, AWS Elastic/Azure Container Service), container native configuration management databases (CMDBs) such as Configuration Management by MicroFocus, and a labelling/tagging policy for containers assist with these challenges. Organisations also require a parallel approach for managing the networking security, logging, host OS and container security.

You need a way to protect containers from threats both outside and within your container ecosystem. A macro-level method is to deploy risk zones (or pods, in Docker language) where containers can freely talk to each other within that zone, but have firewall rules on the boundary of the zone. A micro-level method is to deploy agents with the container image to allow dynamic updates or build firewall rules into the CI/CD pipeline. Either method needs a standardised approach across the IT estate, coupled with automated compliance reporting.

3. Implement container resource controls, and host blast radius protections

Availability and scalability are two reasons why organisations have adopted containerisation technology. This presents governance challenges and the need for effective resource management. Applying resource limits to hosts will increase container capacity and allow for performance increases, resulting in reduced running costs and security risks.

Embedding host protection resource management controls within any container architecture will reduce configuration vulnerabilities and critical risks such as Kernel Panic, which can crash hosts and subsequent containers.

Deploying containers in the cloud allows organisations to simplify many security challenges that would otherwise require more manual processes – host management, easier security mechanisms, automation and scaling. You can significantly reduce the impact radius and overall response times to security incidents with automated actions and alerts to developers and the necessary security teams.

A concern for anyone deploying container-hosted applications is the risk of an attacker gaining access to the underlying container infrastructure through a vulnerable application. Management of container privileges, and having a policy on principle of least privilege, is a simple but effective way to reduce this risk and prevent root-level access in the event that an application is exploited.

For all organisations with containerised environments, it is vital to keep an up-to-date risk register covering all potential security risks. This enables essential security teams to monitor and develop underlying issues that could lead to a security breach.

4. Apply the best practice cyber security guidance

The most likely route of attacks and incidents is where fundamental and basic principles are not followed. This is often the result of outdated or non-existent disaster recovery and failover plans, which mean incidents are poorly managed and the organisation fails to recognise that tried and tested procedures are a vital resource in incidents where there are time pressures.

The NIST 800-190 *Application container security guide* provides best practice on dealing with the most common threats, including:

- Major risks for core components of container technologies.
- Countermeasures for major risks.
- Container threat scenario examples.
- Container technology lifecycle security considerations.

By automating where possible and developing a strong cyber security culture, containers provide the capability to develop a security architecture that responds to business development and enables you to keep on top of the ever-increasing regulatory burden. By thinking about these four areas, you can put the necessary safeguards into place and make best use of containers to support your business and security objectives.

Alan Taberham and Niall Quinn are cyber security experts at PA Consulting.

Security Think Tank: Container security is evolving, so must CISOs

Petra Wenham,

A number of articles on containerisation have been published over the past couple of years. I wrote one on Linux implementations in January 2019, but since that time, Microsoft has been flexing its muscles in that area and there have been a number of new entrants.

Prior to containerisation, the only option was to virtualise the server hardware, and in the process create multiple versions of the hardware – each virtual server then needed to run its own licensed operating system. In containerisation, rather than virtualise a host server's hardware, you essentially virtualise a server's operating system (OS). This, in turn, can lead to greater efficiency in an IT infrastructure as containers are much smaller than a virtual server running its own OS.

In earlier containerisation, the containers were operating system dependant, so a container written for a Linux distribution would not run on a Microsoft system, for example. However, things have moved on apace, and you now can get OS virtualisation software that is not only available for different OS platforms, but offers a common and consistent set of container support functions, such as application programming interfaces (APIs).

In this way, containers become portable between differing OS platforms. You can, of course, run containerisation on virtualised servers, and typically that is what you would get when running IT in the cloud.

Life is getting quite complex for the chief information security officer (CISO), with containers running on virtualised servers potentially in a cloud supplier's remote datacentre. The question is, how does the CISO safeguard the company's data? It's back to basics, together with a realisation that we are dealing with multiple layers of software. The CISO's job therefore includes, as a main function, ensuring the basics are in place and being adhered to. Those basics can be summarised as:

- Having formal policies, procedures, standards and work practice documentation in place. These should be easy to access (intranet, for example) and regularly maintained to ensure that:
  - The latest vendor-supported software or firmware is being employed, not only at the OS level but at the virtualisation level, server hardware and application level, where appropriate;
  - All software is routinely patched, with security patches applied as a priority;
  - All software is configured, not just for function and performance, but also for good security;
  - Staff are trained and competent not only to undertake effective configuration of the various levels of software, but also to

understand the interaction between the various software levels – for example, server hardware BIOS, virtualising server hypervisor, server operating system, OS virtualising software and the containers themselves;

  - That comprehensive monitoring and management systems are in place together with incident reporting, investigation, management and resolution processes.
- Having audit mechanisms in place to regularly check that the policies, procedures, standards and work practices comply with company governance and compliance requirements, are being used and are fit for purpose.
- Ensuring that all systems are regularly and independently checked by external professional companies for security, not only from the internet, but also at the infrastructure and server level.
- Where some or a majority of a company's IT is outsourced, the CISO must ensure that:
  - Contracts accurately reflect a company's policies and standards, and appropriately addresses a company's governance and compliance requirements. The company must have these as a company cannot outsource its compliance responsibility (for example, the General Data Protection Regulation and the Data Protection Act 2018);
  - Security is covered in contracts in detail;
  - The contract allows for independent testing of the company's outsourced IT;
  - The interface between the outsourcer and the company is clearly identified and covers not just operation and management issues, but also has a clear definition of which party is responsible for what functions. This is particularly important for security monitoring and incident reporting and management.

Security Think Tank: Container security starts with good DevOps practice

Paddy Francis,

It is easy to see why the use of containerisation has increased rapidly in line with the increase in cloud services and digital transformation initiatives. Their use allows rapid development and deployment, portability and scalability, and – in some ways at least – more security.

However, the use of containers is a radical change in the approach to developing and deploying applications, and in the infrastructure used to manage them. As with any radical change, the approach to security needs to change, taking advantage of the security properties of containers while addressing the new problems they bring.

Containers provide virtualisation of the operating system (OS), rather than of the hardware as in traditional virtualisation, and are compiled with the application and any dependent programs, libraries and so on, required by the app.

The compiled container is therefore fully self-contained, only needing to access the OS using the specific OS calls necessary for the application to run. Containers are also confined to running in user space. These aspects make it more difficult – but not impossible – for an attacker to compromise the OS, and hence other containerised apps running on the same OS.

On the negative side, additional layers of abstraction mean traditional security tools cannot monitor and protect containerised apps.

Also, the production environment contains the orchestration software which provides scalability by spinning up containers as required and the registry storing the images. The protection of the orchestrator and repository are also security concerns in terms of integrity of the app images and availability of the orchestrator to generate the services.

Using microservices

Another consideration is the use of microservices. This builds on traditional ideals of modularisation of software, but breaks down an application into a number of separate microservices, each of which can be developed separately using different software environments but communicate with each other (typically over https) to provide an overall service.

While having some of the benefits of scalability and agility, and a similar development approach as containers, different microservices making up the same application can run in a container, on bare metal, on a host OS, or in the cloud. They are typically used for distributed and scalable networking applications (load balancing, for example) and can also be used for security monitoring applications with the ability to monitor inside a container.

Protecting the host OS

In the production environment, the key security considerations are protection of the host OS, protection of the orchestration and registry infrastructure and monitoring of containers. The host OS only needs to respond to calls from containers and the orchestration system, and therefore should be hardened in line with recognised guidelines by removing unnecessary services, and so on.

In addition, regular vulnerability scans of the host should be carried out to detect and fix emerging vulnerabilities. A least-privilege model should also be adopted to limit access to the orchestrator and container registry. Also, any front-end services should be secured from attack using application whitelisting. These and other measures should take account of and defend against the OWASP Top 10 most common web attacks.

Monitoring of the containers themselves is more problematic, and currently the best approach is probably behavioural monitoring of the apps against a previously established secure state, together with monitoring of the communications between them at the network layer.

Security in the development environment

The security of the production environment is only one part of the lifecycle, however, and the security and practices in the development environment are equally important. Traditional waterfall or agile development methods produce a single, monolithic app, which will be rigorously tested and probably deployed for a significant time without being updated.

The DevOps process used for containerised app and microservices development, however, is a continuous development process, which provides updated functionality on an ongoing basis with new versions apps deployed as they become available.

This has advantages and disadvantages from a security point of view – while every new iteration of the app could introduce new vulnerabilities, when a problem is found, it can be fixed quickly without a long patch cycle.

Automated testing

Good development practices such as establishing coding standards and code complexity rules are a first step, but in a fast-paced DevOps environment automated security testing is essential to police standards and ensure vulnerabilities are eliminated as far as possible before deployment.

Testing should be done on external and open source code where possible, as well as in-house-developed code. While some testing can only be done on a completed application, testing should be done as early as possible in the development cycle. Static code analysis can pick up violations of coding standards and potential vulnerabilities like unprotected buffer overflows. Because it doesn't need executable code, static code analysis can be run overnight, on code written during the day.

Dynamic code analysis should be done on compliable code, but again can be an integral part of development as well as final release testing. Other security testing can only be carried out on the full app, including fuzzing and penetration testing. While penetration testing and, to some extent, fuzzing are generally performed manually, solutions using artificial intelligence (AI) are now emerging to help speed up the process, but today at least, a skilled pen tester will be needed.

Though not specific to containers and DevOps, supply chain security and security of the development environment are also important factors applicable to any software development, as is management of open source software use and licensing.

As with any new technology, there will be some aspects that can be exploited to deliver improved security, and others that give us new security challenges. The first thing we need to do is understand the technology and the environment in which it is operating, so we can identify critical assets that need to be protected and the new security approaches we need to develop or adopt to protect them.

Security Think Tank: ‘Shift left’ to secure containers

Paul Holland, Principal Research Analyst

The cloud is becoming a vital part of many organisations’ IT roadmap and transformation programme. The current global situation of remote working has helped to drive this move to the cloud for many.

One common method for setting up applications in the cloud environment is to use containers, which are a form of virtualisation but without the traditional hypervisor or the need for a guest operating system (OS) such as Windows Server. The build process and the requirements for the application are much lighter, allowing the application to run much faster since there is no guest OS to consume memory and processor time.

As each container tends to host just the one application, organisations will be responsible for many more containers as compared to virtual machines (VMs). The adoption of cloud services and containers allows for a fast pace of change and automation. But security practices need to be tailored to take all of this into account, especially since the use of containers makes it harder to run traditional security tools such as antivirus as there is nowhere to host it.

This is not to suggest a need for a dramatic shift in how security best practices are implemented – rather a refinement and change in focus on when, where and how to apply them. With agile development and DevOps, many developers are now more involved in the support of the applications they build and thus becoming a jack of all trades – this includes understanding and embedding security into their builds.

Training in secure coding methods (such as the OWASP Top 10) is the most important aspect here – eliminating vulnerabilities early so that containers are secure by design. Another key measure is to adopt a 'shift left' policy for development, whereby the responsibility for security is embedded earlier in the development process – in other words, to the left.

The theory of the shift left policy is that the developers rather than security analysts now check for vulnerabilities. This is supposed to empower the developer to find and fix issues at an early stage of the software development lifecycle and thereafter on a continual basis, as opposed to when the work is complete and a penetration test is performed at the last moment. Theoretically, this should make fixing things cheaper, faster and with less of a burden on the operational teams and infrastructure.

Application level security has therefore become vital priority for chief information security officers (CISOs). It should include implementation of technical solutions such as web application firewalls (WAF), which would ideally link into a Security Operations Centre (SOC) to help monitor for anomalies.

Code reviews should also be conducted, whether that be an internal peer review, external expert review or software review. Such reviews can spot vulnerabilities before code is made live within applications.

In the context of agile development and DevOps, speed is often a measure of success, but secure development of applications should also form part of the criteria for determining whether a sprint is successful. CISOs need to realise that developers should be granted time to develop securely and not judge their performance solely by the time to build.

Securing containers is not a one stop shop but a multi-faceted undertaking. Combining the above into a cohesive plan and creating a secure development lifecycle that is enhanced with technical monitoring will provide the CISO with assurance that containers can be used securely and effectively in an organisation's IT environment.

In this e-guide

Security Think Tank: Securing containers needn't be taxing

Andrew Morris, Managing Consultant

Until relatively recently, security appliances were provided by their suppliers in physical blades that were installed on an organisation's system. Today, this software is increasingly likely to be provided in containers.

At their core, containers are isolated collections of software, gathered together into a working package that can be configured and managed independently, both of other containers and the hardware on which they run. As well as being easily deployed on any virtual or physical server, the segregated nature of containers allows rapid development and ongoing maintenance because they can be moved from server to server dynamically without their operation being disrupted or software compatibility being an issue.

As with any emerging technology, containerisation introduces substantial opportunities for organisations to enhance the effectiveness and efficiency of their processes and activities.

At the same time, it doesn't come with "best practice" ways of ensuring it is secure, and as the technology itself and methods used to deploy it continue to evolve, the challenge is to determine the new risks it introduces and the resulting mitigation strategies that need to be implemented to ensure that the security posture of the enterprise is not compromised.

This is exacerbated by the "black box" nature of containers, in which only inputs and outputs are visible.

Despite the container concept being relatively new, some standard security practices are effective in managing the risks. As containers are only as secure as the software on which they are running, security assurance processes need to be engaged at the start of the development life cycle of the container and continued throughout. This includes assessments to ensure that none of the software libraries used house any known vulnerabilities and checks that access permissions within the container are appropriate and not running at elevated levels.

However, while it is relatively straightforward to use vulnerability management tools to assess whether a piece of software is secure, containers often consist of multiple technology stacks (from web servers, virtual machines and databases), and interconnected software installations. Rather than viewing security assessments on a container as a single vulnerability test, these need to be performed as they would on a new application so that all software components are checked.

Security by design

These assessments also need to be performed from day one of a project, which requires them to be built in at the design and requirements stage; the later in the process they are introduced, with this sometimes being at go-live or after production has started, the greater the likelihood of security flaws arising or causing unexpected business disruption.

For software being delivered by a third party, checks should focus on the developer, as part of the normal security supplier assessment process. They also need to look at the level of control that the organisation using the container will have and how this will operate; determining if the container requires full root permissions to run, or whether permissions can be controlled to restrict access to resources on the server, for example.

Zero trust policies

Treating containers like micro zero-trust environments is another viable approach to security. Restricting what they communicate with and authenticating and authorising all requests and commands reduces the amount of damage should compromised, or less than secure containers, be introduced into the organisation's IT landscape.

Monitor and patch

Containers should be integrated with any existing monitoring processes to identify unexpected events or indicators of compromise. Ideally a baseline will be provided by a full list of expected behaviour and data flows established during development.

Because the nature of containers means it is not always possible to look at the traffic or logs within them, monitoring needs to focus on the interaction of the overall container with physical resources (such as servers and storage) and the data flows entering and exiting it.

Ensuring software is current (and therefore as secure as possible) requires containers to be included in change and patch management processes. This is especially critical for any virtualisation software being used, as this is where criminals will often look for flaws to exploit other components running on it.

Off-the-shelf solutions

With the growth in the adoption of containers, software solutions that specifically monitor container security have also been developed. But before choosing this route, organisations need to evaluate the cost when weighed up for the volume of containers within their IT environment and critical processes, as well as take into consideration that the technology is still changing and adapting; relying solely on a single tool, even it is the most effective option now, might not meet long term goals.

Efficiency enabled by security

Developers themselves provide the strongest defence when securing containers as they can instigate safeguards such as following well-known design practises and formulating containers with maintainability and security in mind. And in adopting security-by-design principles they can reduce the potential risk to the organisation by decreasing or eliminating the number of issues that need resolving once the container has gone live.

In addition, CISOs need to ensure proven secure software is selected for use within a container, code audits and vulnerability analysis are performed during development, and that all interfaces and data collection are mapped.

These steps will dramatically reduce the overheads required by the security function and enable the organisation to realise the full efficiency benefits offered by container technology.

In this e-guide

How to apply zero-trust models to container security

Nicholas Fearn,

Organisations are increasingly replacing archaic software development approaches with containers, which allow them to develop, deploy and scale applications much more quickly than traditional methods.

But despite these benefits, containers are not perfect. Their adoption has also resulted in new challenges for security teams, particularly around data protection, container image vulnerabilities, cyber attacks, unauthorised access and a whole host of other risks. Could zero-trust models mitigate these? And if so, how can organisations apply them to container security?

Although containers provide greater efficiency and scalability for development teams, they can have significant implications for security. Often, traditional perimeter-centric security models are not suitable and new approaches are needed.

Kevin Curran, IEEE member and professor of cyber security at Ulster University, says: "The dynamism of containers can cause problems for traditional security environments, due to the complexity involved in networks, overlays and dynamic IPs, mixed with the limitations of traditional firewalls which struggle to identify nefarious activity.”

But that is where zero-trust security models can help. Curran explains that when combined with policies based on identities of workloads, they allow enterprises to build a picture of what is communicating over their network. “Here, zero-trust based on identity can prevent compromised workloads from communicating as each identity will not be recognised,” he tells Computer Weekly.

“The need for a zero-trust security model has arisen in part because enterprises no longer tend to host data in-house, but rather through a variety of platforms and services that reside both on- and off-premise, with a host of employees and partners accessing applications via a range of devices in diverse geographical locations. This means the traditional security model is no longer fit for purpose.”

Curran says zero-trust security can be implemented by updating network security policies, validating each device logging into the network, securing networks with a variety of network, perimeter and microsegmentation, implementing multifactor authentication and conducting periodic reviews of user access.

He adds: “The main applications for zero-trust security require new approaches, such as using network/microsegmentation based on users and locations. It also requires enforcement of identity and access management [IAM], next-gen firewalls, orchestration, multifactor authentication and file system permissions.

"Ideally, this is something that is done slowly in steps, as it entails pilot projects and tweaks in a lab environment before deploying. It is crucial to ensure that the zero-trust infrastructure is seamless for employees."

Catalyst for zero-trust

Many experts believe the need for zero-trust security models is growing along with the increased adoption of containers across the enterprise landscape. Neil Thacker, CISO of software firm Netskope, agrees with Curran that such models are paramount for security teams deploying containers.

Thacker says: "Cloud-based applications and container-based applications – not to mention cloud-based, container-based applications – are a further catalyst for interest in zero-trust network access [ZTNA], specifically because of the disregard both cloud apps and containers have for traditional perimeter approaches to security."

He says security teams need consistent security controls across all applications as a fundamental rule, regardless of whether they are based on a traditional stack, are virtualised or hosted in containers.

"While security must not stand in the way of the inherent benefits of containers, such as portability, the controls and methods of securing access to containers is key," says Thacker. "Firewalls aren't useful because they are not app-aware, and even next-gen firewalls that apply controls to the application layer still require illogical network arrangement and overly permissive security policies to account for the rapid changes of network IP addresses within containers.

"This is why cloud-based ZTNA appeals to organisations, because instead of restricting connectivity and restricting the potential benefits that containers offer, ZTNA can prioritise the application, however and wherever it is hosted."

Making containers impenetrable

Containers may be a powerful tool for developers, but they are becoming a security nightmare as cyber criminals increasingly target them. By gaining unauthorised access to containers, hackers can cause all sorts of mischief, potentially across a large virtual environment.

David Warburton, senior threat evangelist at application threat specialist F5 Labs, says: "If an attacker can leverage vulnerable code within a container, they may be able to impersonate that service and access data never intended to be made available. Decades-old vulnerabilities, such as injection attacks, apply just as much to modern code running inside a container as they do traditional, monolithic apps.

"The difference now is that containers, and the microservices they provide, have exponentially increased the surface area available for attack, putting data at greater risk. In addition, network-related problems, such as access control, load balancing and monitoring, that had to be solved just once for a monolith application, must now be handled separately for each service within a cluster."

By applying zero-trust models, security teams can mitigate these threats. Warburton adds: "A key tenet of zero-trust is that every single request should be secured, regardless of who or where it came from. This model needs to be applied to containers so that all communications are encrypted, even those between internal services."

To prevent unamortised access, organisations must enforce strong authentication mechanisms for their containers. "Mutual digital certificates should be used to ensure only trusted containers can communicate with one another. Finally, strong, role-based access control is needed to ensure only authorised users and services are performing actions that they have explicitly been given permission for," says Warburton.

"Create a service mesh security to be handled in a more efficient way by combining security and operations capabilities into a transparent infrastructure layer that sits between the containerised application and the network. Emerging today to address security in this environment is the convergence of the zero-trust approach to network security and service mesh technology."

Sandy Carielli, principal analyst at market research company Forrester, warns of a disconnect between developers adopting containers and security teams left to pick up the pieces. She says: "Development teams are eager to adopt containers due to their scalability and cost-efficiency, but one of the realities is that dev makes the containerisation decision, and then security finds itself going along for the ride and figuring out the security implications and requirements." Overstuffed images, in particular, are a major challenge in container security, says Carielli. "Developers typically pull images from repositories, and those images contain more tools, features and permissions than the developer needs for their particular use case," she explains.

"However, dev teams rarely have time to scale down the image to just the essentials. DevSecOps teams need to set time aside to look at the images they are using and remove the functions and permissions that they don't need. As a basic example, don't run containers with root permissions."

Carielli says microsegmentation is another aspect of zero-trust that applies to containers. "Organisations leverage application microsegmentation tools to evaluate both north-south and east-west traffic and manage the flow of data among application components – these could be containers, APIs or serverless functions," she says.

"Runtime container security tools map the flow of data between containers, allow you to set policy on how containers interact, and can spin down containers that unexpectedly change configuration."

Creating an effective security strategy for containers

When deploying containers, organisations are effectively exposing themselves to myriad security problems that must be mitigated if they want to get the most out of these technologies.

Benoit Heyndericks, principal analyst at the Information Security Forum, says: "The lightweight nature of containers removes the need for traditional IT infrastructure security controls such as a constant patching cycle and the extreme reliance on the firewall for protecting a network-based perimeter.

"But it brings new types of risks due to the rapid lifetime of containers, while adding increased networking complexities and placing emphasis on the need to apply secure design principles early on, such as secure coding practices."

With a zero-trust model, organisations can ultimately create an effective security strategy for containers, says Heyndericks. "By focusing on authenticated identities, least privilege principle, defined microsegmentation, traffic monitoring and logging, the model relies on the principle of 'never trust, always verify'.

"This is a paradigm shift from traditional security models and can only be addressed by deploying it in a phased and defined manner, focusing on specific groups of applications, such as the most sensitive ones for a start."

Heyndericks says organisations applying zero-trust models to container security should be supported by strong coding practices for all application development activity. This, he says, will put the organisation in a strong position to respond to the growing demand from developers to use rapid deployment platforms such as application containers.

Heynderickx adds: "Modern businesses can therefore benefit from using agile development technologies to deploy secure applications in a fast manner for their demanding customers."

Understandably, organisations want to roll out new software quickly and efficiently to stay ahead of the curve and achieve competitive advantage. So containers are the perfect answer. However, their adoption has resulted in clear security challenges, and it is crucial that firms take steps to address these if containers are to return value on investment. Therefore, developers must work with security teams when they look to adopt and use containers.

In this e-guide

Getting more CW+ exclusive content

As a CW+ member, you have access to TechTarget's entire portfolio of 140+ websites. CW+ access directs you to previously unavailable "platinum members-only resources" that are guaranteed to save you the time and effort of having to track such premium content down on your own, ultimately helping you to solve your toughest IT challenges more effectively—and faster—than ever before.

Take full advantage of your membership by visiting www.computerweekly.com/eproducts

Images; stock.adobe.com