#becrypt

Paradox through MITRE ATT&CK

Using the MITRE ATT&CK Framework to assess operating system security

The MITRE ATT&CK framework provides an approach to assessing the contribution of security technologies such as Paradox within broader architectures.

Paradox is a security-focused operating system developed in collaboration with UK Government to meet the growing need for robust endpoint protection against elevated cyber threat. Paradox is designed specifically for providing secure access to cloud and online services, and as a light-weight operating system is able to support a number of security features that would be difficult or impossible to implement for a general-purpose operating system.

Initially deployed within government classified environments, the functionality and security features of Paradox have broader relevance across the critical national infrastructure (CNI).

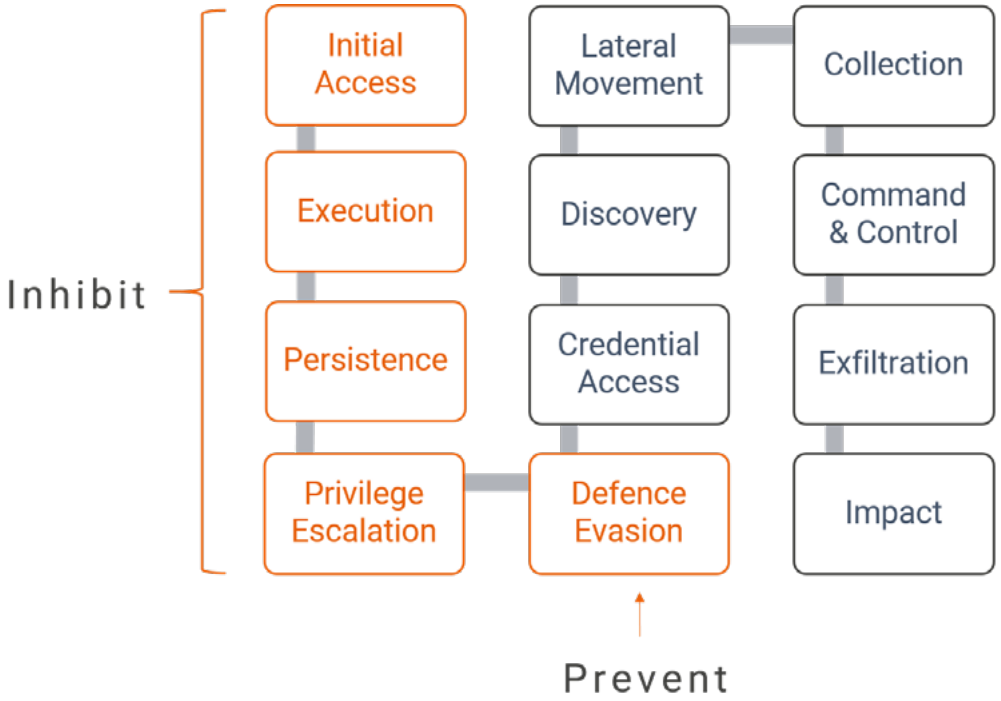
However, assessing the relevance of Paradox for security architects within the CNI can be a challenge without a commonly accepted framework.

The increasingly referenced MITRE ATT&CK framework provides an approach to assessing the potential contribution of security technologies within broader architectures. MITRE ATT&CK is a knowledge base of techniques that are used by adversaries through the common stages of a compromise, providing information relevant to both penetration testers and defenders for understanding the nature of how adversary techniques typically work.

Techniques, such as gaining access to user credentials, represent how an adversary achieves higher-level tactical objectives, such as moving laterally or extracting data. The ATT&CK framework outlines mitigations that may be deployed for each identified technique, and highlights their relevance to general purpose operating systems.

As Paradox is a Linux-based operating system, the Linux subset of ATT&CK tactics shown below provides a starting point to assess the relevance of the additional mitigations introduced by the Paradox security architecture.

This paper provides an overview of the novel security mitigations enforced by Paradox for the five adversarial tactics ranging from Initial Access to Defence Evasion.



Linux ATT&CK subset v Paradox Security Objectives

Initial **Access**

Definition: **Techniques that use various entry vectors to gain their initial foothold**

Paradox Mitigation: **Limited User Model**

A common technique employed to achieve an initial foothold is to exploit valid user or administrative accounts. Compromised credentials may allow an adversary to avoid using malware or tools in conjunction with the legitimate access that compromised credentials provide, making it harder to detect their presence. **For Paradox, users have no knowledge of an operating system (OS) password**, as a fixed operating system account is used for all interactive sessions, with account permissions defined at build time.

Significantly, there is actually no Admin (root) password.

By design Paradox limits the user's ability to reconfigure client software. All software installation and configuration is centrally managed, with access to the corresponding functionality removed from the interactive user. Equally, policies, such as those that control access to removable media are centrally managed, allowing media device use as an initial access vector to be controlled.

Execution

Definition: **The adversary is trying to run malicious code**

Paradox Mitigation: **Paradox prevents arbitrary code execution**

Execution consists of techniques that result in adversary-controlled code running on a target system. For Paradox, all valid executable files are known and are signature checked before access.

Paradox uses 'mount' permissions defined at system build time to determine how system and data partitions are used. **Anything that is writable is not executable, and anything that is executable is not writable.** This prevents an adversary modifying an existing executable within the system partition, or creating a new executable on a data or removable media partition, without first circumventing the permissions model.

To detect attempts at modifying or creating in-memory executables, relevant system calls are monitored and blocked. Additionally, arbitrary user interaction is constrained restricting a user's ability to run arbitrary commands. A standard Terminal window does not exist, and applications such as File Explorer are patched to prevent right-click command execution.

Persistence

Definition: **The adversary is trying to maintain their foothold**

Paradox Mitigation: **Paradox uses a read-only file system**

Persistence consists of techniques that adversaries use to keep access to systems across restarts, changed credentials, and other interruptions that could cut off their access. The use by Paradox of a system partition that is mounted as read-only inhibits an adversary's ability to use storage to gain persistence. However, Paradox avoids the drawbacks of typical 'read-only' systems, such as 'Live CDs'. The Paradox system partition may be actively updated (replaced) by one of Paradox's System Services which acts under control of the management platform.

Authorised applications downloaded from the management platform are cryptographically validated before being mounted and subsequently executed.

User settings are stored in a writable data partition for which build-time permissions prevent execution. However, **applications or users do not write directly to storage.** Paradox uses its System Services to write data. Services do not run as root, but with appropriately limited privileges defined as part of system build.

For further control over the nature of the data persisted, the System Services validate the simple data types that are written in encrypted form form to the data partition, ensuring compliance with defined allowable data types.

Privilege Escalation

Definition: **The adversary is trying to gain higher-level permission**

Mitigation: **Standard support for permissions management is removed**

Privilege Escalation consists of techniques that adversaries use to gain the higher-level of permissions on a system needed for the remaining stages of attack. As outlined above, for Paradox no root password is available locally, and all administrative actions are performed by Paradox System Services. Paradox uses a stripped down version of Ubuntu, using standard tools to create a minimal build. This helps reduce the attack surface, but more importantly makes it possible for Paradox to **enumerate and control all files that should be executable, and strictly enforce which files may obtain root privileges.** The standard mechanism to change privileges is removed from the system (Linux Access Rights Flags).

Moving from commodity attacks to elevated threat

The security controls outlined above provide robust protection against commodity attacks, and extend to protecting against a range of targeted attacks. However, for an elevated threat scenario, a highly sophisticated adversary with access to multiple zero day kernel exploits, and significant amounts of time and money, even these defences may be susceptible to prolonged attack. Consequently, **the primary security objective for Paradox, as defined by government, was to ensure that in such a scenario, the attack can be detected.**

The 'gold standard' for compromise detection is during system start, where non-operating system, and indeed non-software components can be used. However, with general-purpose operating systems, current health measurement techniques are limited. For example third-party device driver validation relies on certificate checks that are susceptible to compromise and standard white-listing, anti-virus and anomaly detection techniques can be evaded. The result for standard operating system architectures is only being able to achieve confidence that initial operating system components are not compromised, but having less confidence in the health of the overall system.

Defence Evasion

Definition: **The adversary is trying to avoid being detected**

Paradox Mitigation: **Device health is based on measurements of ALL firmware and software**

Defence Evasion consists of techniques that adversaries use to avoid detection throughout their compromise. **Paradox ensures that any compromise to any firmware, operating system or application software would be detected on a system start.** Paradox makes use of Secure Boot, the now pervasive industry security standard to make sure that a device boots using only known good software. When a device starts, the firmware checks the signature of boot software, if the signatures are valid the firmware gives control to the operating system. The Paradox boot-loader cryptographically validates operating system kernel and system files with keys protected by a hardware root of trust (the Trusted Platform Module).

When running, Paradox cryptographically validates every block that is read from storage (using a process called DMVerity), ensuring all subsequent cryptographic validation is chained to the hardware root of trust. Individual applications are cryptographically signed, and validated with certificates managed by the Paradox management system, and therefore not susceptible to typical third-party certificate exploits. This approach uniquely allows all operating system files, kernel drivers and applications to be included within strong cryptographic signature validation.

Defence evasion also extends to client-server interaction, allowing protected services to ensure that they are only accessed by devices in a known healthy state. Paradox supports an independently validated remote attestation protocol. Device Health and Identity Management is implemented by the Paradox Authentication Service, which may be deployed to protect exposed services as part of a Zero Trust Architecture combining user and device signals for service access policy. Additionally, the Paradox Authentication Service is compatible with the latest High-Assurance cross domain gateways, providing hardware-based validation of the integrity of all device management traffic.

Summary

The MITRE ATT&CK provides a useful framework to position and assess security controls implemented by Paradox. The analysis outlined above demonstrates that being a light-weight operating system focused on providing secure access to cloud and online services, Paradox is able to implement a number of mitigations that would be difficult or impossible to achieve with a general-purpose operating system.

The mitigations described provide significant defence against commodity attacks, but ensures that defender's ability to ultimately detect even an advanced attack is based on the strong properties of cryptography.

paradox

About Becrypt

Becrypt is an agile London-based UK SME with 20 years cyber security expertise, established through the development and delivery of cloud, mobile and endpoint platforms. We supply governments and security-conscious commercial organisations, large and small, with a range of security solutions and services - from funded research, to commercially available products and flexible managed services. Becrypt have worked with UK Government and platform vendors to pioneer and deploy device health identity management products and services, based on the NCSC CloudClient Architecture.

https://www.becrypt.com/uk/products/paradox/

Becrypt Ltd
Artillery House
11-19 Artillery Row
London SW1P 1RT
UK Company Number 4328430
Tel: 0845 838 2050

#becrypt