On-Chain Dynamic Policy Evaluation for Decentralized Access Control

Yepeng Ding[1(✉)], Junwei Yu[2], Hiroyuki Sato[2,3], Tohru Kondo[1], and Yan Bai[4]

[1] Hiroshima University, Hiroshima, Japan
yepengd@acm.org, tkondo@hiroshima-u.ac.jp
[2] The University of Tokyo, Tokyo, Japan
yujw@satolab.itc.u-tokyo.ac.jp
[3] National Institute of Informatics, Tokyo, Japan
schuko@nii.ac.jp
[4] University of Washington Tacoma, Tacoma, WA, USA
yanb@uw.edu

Abstract. Existing decentralized access control solutions primarily cater to off-chain systems and adopt static policy evaluation mechanisms that fall short in addressing the unique requirements of on-chain environments. Challenges such as granularity, upgradability, and cost-efficiency in access control for on-chain systems remain unresolved. To address these challenges, we propose and formalize On-Chain Dynamic Policy Evaluation (OnDPE), a novel policy evaluation mechanism serving as an infrastructure for decentralized access control with flexible granularity and upgradability. Furthermore, we formulate an OnDPE-integrated Decentralized Access Control (OnDAC) framework tailored for decentralized applications on EVM-compatible blockchains.

Keywords: Decentralized access control · Smart contract · Decentralized applications · Blockchain · Decentralized computing

1 Introduction

Decentralized access control for on-chain systems, particularly Decentralized Applications (DApps), remains understudied. Recent investigation reveals that over $300 million in losses have been attributed to vulnerable access control flaws of DApps [5,12]. More broadly, attacks exploiting business logic, reentrancy, and input validation vulnerabilities leading to unauthorized access have indirectly resulted in losses exceeding $1.1 billion [5]. Such breaches have significantly undermined the health of the cryptocurrency market and have generated adverse effects on various secondary markets.

This research was partially supported by KAKENHI (Grant-in-Aid for JSPS Fellows) 22KJ0614 and National Science Foundation (NSF) Grants 2334196.

T. Zhu et al. (Eds.): ICA3PP 2024, LNCS 15254, pp. 337–346, 2025.
https://doi.org/10.1007/978-981-96-1545-2_21

Existing solutions for DApps fall short in addressing challenges such as flexible granularity, upgradability, and cost-efficiency due to their reliance on static policy evaluation. For example, OpenZeppelin[1], the most widely used on-chain access control library safeguarding over $15 billion in assets for DApps, provides only coarse-grained access control. Although its proxy upgrade pattern supports smart contract upgrades, the gas consumption is comparable to redeploying an entire smart contract, even when slight modifications pertain only to access control logic. Consequently, managing access control, such as updating access privileges and subject roles, remains highly costly. Additionally, customized access control mechanisms may offer flexible granularity by introducing modifiers in function signatures, embedding conditional or require statements in function bodies. Nevertheless, customization may also introduce potential risks and still faces challenges related to upgradability and cost-efficiency.

Motivated by tackling the challenges inherent in existing solutions, we propose On-Chain Dynamic Policy Evaluation (OnDPE), a novel policy evaluation mechanism fully operating on blockchains to provide flexible granularity and upgradability. Unlike traditional methods that rely on static policy evaluation mechanisms hardcoded into smart contracts and necessitate costly redeployment for administrative operations such as policy management, OnDPE allows for cost-efficient modifications to on-chain state variables without the need for redeployment. Moreover, OnDPE can function as a universal policy evaluation infrastructure for all on-chain access control frameworks with only a single deployed instance. Building on OnDPE, we formulate OnDAC, a decentralized access control framework that adheres to standardized terminology to define core components with an authentication layer leveraging a decentralized identity model and a multi-signature paradigm. OnDAC serves as a template for replication and further refinement in various access control scenarios.

2 Related Work

We have surveyed the recent decentralized access control frameworks and found that most are designed to secure off-chain systems, such as the Internet of Things (IoT) [4,8], smart health [6,15], and data sharing [10,11,16].

In the realm of IoT, the Decentralized Lightweight Group Key Management for Access Control (DLGKM-AC) [4] was proposed to address scalability and computational overhead challenges via a hierarchical architecture. DLGKM-AC mitigates the rekeying burden on core networks and supports scalable and flexible access control across multiple IoT groups. However, it lacks support for fine-grained access control, which is critical for addressing diverse security requirements within IoT environments. Later, Bloccess [8] was proposed as a sophisticated fine-grained access control framework for IoT. Bloccess implements a hybrid blockchain structure to enforce access policies with high granularity and security. Additionally, it incorporates robust identification and administration mechanisms to facilitate access control management.

[1] https://www.openzeppelin.com/.

In the context of smart health, decentralized access control frameworks are pivotal for protecting data privacy. Recent studies [6,15] explore fine-grained access control frameworks to safeguard electronic health records through decentralized mechanisms. These frameworks enable secure collaboration with data centers while ensuring the privacy and integrity of health data. For a broader application, Droplet [16] was proposed as a user-centric architecture that authorizes data sharing and allows users to define fine-grained stream-specific access control policies. Additionally, decentralized access control [2,10,11] incorporating self-sovereign identity [7] has been explored for decentralized data aggregation.

Furthermore, efforts to develop general-purpose decentralized access control frameworks are exemplified by works [3,13]. In [13], the authors proposed an on-chain static policy evaluation mechanism that translates XACML policies into smart contracts through a policy translation point. Each policy is embedded within a smart contract containing hardcoded policy evaluation logic. However, this method is not scalable for administrative operations such as updating policies. Each policy update necessitates redeploying new contracts and reconfiguring protected resources, resulting in low cost-efficiency and scalability. On the contrary, a dynamic policy evaluation mechanism was formulated in [3], which maintains a fixed cost for policy updates where policies are limited to role information. Consequently, this work allows only simplistic logic in policies and lacks the granularity required for more complex access control scenarios.

To the best of our knowledge, no existing study on on-chain dynamic policy evaluation comprehensively addresses the issues of granularity, cost-efficiency, and scalability. Our work aims to fill this gap by introducing a novel mechanism that dynamically interprets ACPs during smart contract execution, thereby overcoming the limitations of static policy evaluation mechanisms and improving the flexibility and cost-efficiency of decentralized access control frameworks.

3 On-Chain Dynamic Policy Evaluation

3.1 Access Control Policy

An ACP is a set of rules associated with a set of protected smart contract functions. These functions are managed by a context manager, which interacts with an evaluation engine to determine whether the invocation of these protected functions is permissible. The evaluation engine implements the OnDPE mechanism, while context managers control the authorization processes.

Given that OnDPE operates entirely on blockchains, we first formalize a blockchain as a distributed state machine. In this formalization, the blockchain transitions from one state to another through transactions that are validated and accepted by peer nodes within the decentralized network. Each peer node maintains an independent record of local states, necessitating a consensus mechanism to ensure that the network converges on a consistent record of global states. The concept of global states is formally defined in Definition 1.

Definition 1 (Global State). *The set of global states S of a distributed ledger consists of total functions $\sigma : V \mapsto R$, where V is a set of typed state variables and R is a set of ranges for variable types. Here, V consists of four subsets:*

- *V^b for Boolean state variables, such that $\forall v \in V^b : \sigma(v) \in C^b = \{\top, \bot\}$,*
- *V^n for numeric state variables, such that $\forall v \in V^n : \sigma(v) \in C^n = \mathbb{N}$,*
- *V^s for string state variables, such that $\forall v \in V^s : \sigma(v) \in C^s = \{\texttt{0x20}, \ldots, \texttt{0x7E}\}^*$, and*
- *V^d for address state variables, such that $\forall v \in V^d : v \in C^d$ is a valid address.*

The rules of ACPs are described in propositional logic about the global states of a blockchain. We formally define ACPs in Definition 2.

Definition 2 (Access Control Policy). *An access control policy is a propositional formula over the set E^a of atomic propositions about the global state of a distributed ledger $\mathfrak{L}$, where the formation rule for E^a is defined by*

$$a ::= b \mid n_0 = n_1 \mid n_0 \leq n_1 \mid s_0 = s_1, a \in E^a$$

where

- ***Boolean expression*** $b ::= c^b \mid v^b \mid v_f^{b,m^d} \mid m^b$ *with* $c^b \in C^b, v^b, v_f^{b,m^d} \in V^b, m^d \in M^d, m^b \in M^b, b \in E^b$,
- ***Numeric expression*** $n ::= c^n \mid v^n \mid v_f^{n,m^d} \mid m^n \mid n_0 + n_1 \mid n_0 - n_1 \mid n_0 \times n_1 \mid n_0 \div n_1 \mid n_0 \bmod n_1$ *with* $c^n \in C^n, v^n, v_f^{n,m^d} \in V^n, m^d \in M^d, m^n \mid M^n, n, n_0, n_1 \in E^n$, *and*
- ***String expression*** $s ::= c^s \mid v^s \mid v_f^{s,m^d} \mid m^s \mid s_0 \oplus s_1$ *with* $c^s \in C^s, v^s, v_f^{s,m^d} \in V^s, m^d \in M^d, m^s \in M^s, s, s_0, s_1 \in E^s$.

$M = M^b \uplus M^n \uplus M^s \uplus M^d$ is a typed policy parameter set that functions as placeholders for the unknown values at the point of policy initialization.

A state variable represented by $v_f^{t,d}$ with $t \in \{b, n, s\}, d \in M^d$ stores the returning value of type t of the pure function f associated with the address d.

An atomic proposition can be categorized as Boolean, numeric, or string based on its composition of E^b, E^n, E^s, respectively.

ACPs are managed by an ACC in an access control system. An ACC consists of a global state, a set of protected smart contract functions, a set of ACPs, and a binding function mapping actions to ACPs, such that under the global state, a protected smart contract function can only be triggered if its bound ACP is satisfied. In this manner, OnDPE is independent of ACCs and can adapt to various access control systems. Formally, we define ACCs in Definition 3.

Definition 3 (Access Control Context). *An access control context Γ of a distributed ledger $\mathfrak{L}$ is a quintuple $\langle \varepsilon, F, P, \omega, \rho \rangle$ where*

- *ε is the address of a deployed evaluation engine,*
- *F is a set of protected smart contract functions of $\mathfrak{L}$,*

– *P is a set of ACPs stored on $\mathfrak{L}$,*
– *$\omega : P \mapsto \mathcal{E}$ is an argument binding function, and*
– *$\rho : F \mapsto P$ is a policy binding function.*

Here, for any $p \in P, \omega(p) : M_p \mapsto C^b \uplus C^n \uplus C^s \uplus C^d$ evaluates the policy parameters M_p of p to concrete values in their corresponding ranges.

Policy parameters enhance the flexibility of decision-making and policy control. For decision making, policy parameters can serve as placeholders for subjects and environmental factors, allowing the same ACP to yield different outcomes for different subjects and under varying conditions over time. This dynamic evaluation capability enables the system to adapt to changing circumstances and user contexts seamlessly. For policy control, policy parameters facilitate the setting of dynamic values, enabling ACPs to be updated and refined without the need for deploying entirely new policies. This capability ensures that ACPs can be fine-tuned to meet evolving security requirements and operational needs and enhance the agility and responsiveness of access control.

3.2 Mechanism

OnDPE involves two distinct roles: access control administrator and subject. Administrators manage decentralized access control systems enabling OnDPE via context managers, while subjects invoke protected smart contract functions after being authorized by context managers.

Context Manager. A context manager refers to a smart contract that implements an ACC $\Gamma = \langle \varepsilon, F, P, \omega, \rho \rangle$ defined in Definition 3 and allocates a set of state variables to store ε, P, ω, and ρ.

Management. The administrator manages a decentralized access control system via a context manager controlling an ACC Γ. Administrators deploy ACPs P into the state variables of the context manager by default. Also, administrators can maintain references to ACPs deployed on any accessible smart contracts by storing their addresses. Similarly, administrators register the addresses of protected smart contract functions F and associate them with specific ACPs P within the state variables of the context manager. Additionally, the administrator determines the address of an evaluation engine responsible for enforcing the evaluation processes under Γ.

Administrators can modify the rules governing the administrated access control system by updating elements of Γ. For instance, they can reassign an ACP $p \in P$ to a protected function $f \in F$ by updating the mapping function $\rho(f) = p$. This flexibility allows administrators to dynamically adjust ACCs to meet changing requirements.

Dynamic Argument Binding. A context manager dynamically updates the argument binding function ω of its controlled ACC Γ to ensure ACPs are parameter-free while processing access requests. Each access request under Γ carries a target protected function $f \in F$ and an argument function η. The dynamic argument binding procedure is shown in Algorithm 1.

A context manager dynamically updates the argument binding function ω of its controlled ACC Γ to ensure that ACPs are parameter-free while processing access requests. Each access request under Γ includes a target protected function $f \in F$ and an argument function η. The procedure for dynamic argument binding is detailed in Algorithm 1.

Algorithm 1. Dynamic argument binding

Require: $\omega \in \Gamma, f \in F, \eta$
Ensure: Parameter-free p or exception
  $p \leftarrow \rho(f)$
  for $m \in Keys(\omega(p))$ **do** ▷ For each policy parameter m of p
    if $\eta(m) = \varnothing$ **then**
      Throw exception *Argument m is missing*
    else
      $p \leftarrow Replace(p, m, \eta(m))$ ▷ Replace the occurrences of m in p with $\eta(m)$
    end if
  end for
  return p

Evaluation Engine. An evaluation engine is a smart contract that interprets ACPs and can serve multiple context managers.

As shown in Fig. 1, the evaluation engine cooperates with an ACC to evaluate ACPs. The evaluation engine exposes *Evaluate* function for context managers to enforce evaluation processes and return a Boolean value as the evaluation result.

Specifically, an evaluation engine first evaluates primary forms of the ACPs normalized by the Tseytin transformation [17] and parameter-free check. Notably, an ACP is parameter-free if all its policy parameters are assigned specific values. Then the evaluation engine interprets the semantics of each clause in an ACP using short-circuit evaluation, i.e., the algorithm terminates and returns $\perp$ if any clause evaluates to $\perp$. As detailed in Definition 2, an ACP is constructed from three types of atomic propositions on the global state set S. Consequently, we can establish the semantics of these three types of atomic propositions and evaluate ACPs to Boolean values concerning logical connectives.

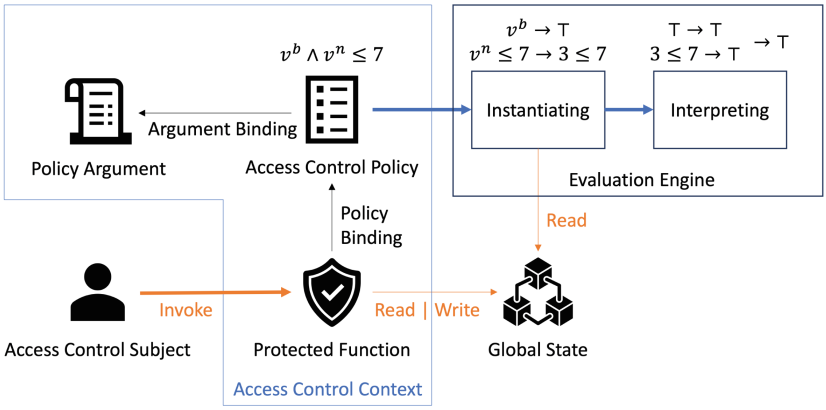

Fig. 1. Overview of the evaluation engine. Bold blue arrows indicate the input and output flow. (Color figure online)

4 Decentralized Access Control with OnDPE

We have developed an OnDPE-integrated Decentralized Access Control (OnDAC) framework that adheres to the terminology defined in the eXtensible Access Control Markup Language (XACML) [1]. OnDAC is designed to operate on EVM-compatible blockchains [18] that support smart contract functionalities and utilize gas mechanisms to manage computational resources and incentivize network participation.

4.1 Architecture

OnDAC adopts a decentralized identity model to uniquely identify administrators and subjects. Administrators and subjects are authenticated based on Decentralized Identifiers (DIDs) [14]. Besides, OnDAC uses a multi-signature paradigm [7,9] to authenticate administrative operations, such as managing ACCs, attributes, and administrative parameters.

The core components of OnDAC include a Policy Administration Point (PAP), a Policy Information Point (PIP), a Policy Enforcement Point (PEP), and a Policy Decision Point (PDP), each serving a specific role in the authorization process.

As illustrated in Fig. 2, subjects interact directly with the PEP to submit access requests for a protected function. The PEP then queries the PDP for access decisions. The PDP, utilizing its integrated evaluation engine and subject attributes obtained from the PIP, makes an access decision and returns it to the PEP. If the decision is positive, the PEP proceeds to invoke the requested function that performs coded read and write operations on the global state. If the decision is negative, the PEP denies the access request and records the request information for auditing purposes. Administrators manage subject attributes

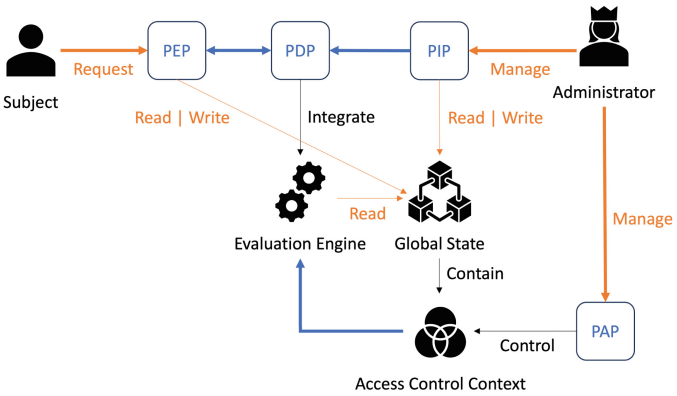

Fig. 2. The architecture of OnDAC.

through the PIP and control ACCs via the PAP. Notably, ACCs are a subset of global states.

4.2 Enforcement Patterns

The PEP provides two enforcement patterns to accommodate various access scenarios and requirements: injection pattern and delegation pattern.

Injection Pattern. In the injection pattern, the PEP is embedded into the protected function as an assertion statement that calls the PDP for an access decision. If the PDP returns a negative decision, the function execution is reverted to prevent unauthorized access. This pattern is particularly useful for ensuring that access control checks are performed consistently within the function's execution flow.

Programming languages like Solidity, commonly used for smart contract development on EVM-compatible blockchains, support the use of function modifiers. Function modifiers allow for encapsulating assertion statements into reusable components that can be applied to multiple functions. For instance, in Solidity, an assertion for access control using *ACP1* can be encapsulated into a modifier and applied to a protected function f.

Delegation Pattern. In the delegation pattern, the PEP is explicitly deployed as an independent smart contract. It makes a delegate call to the protected function if the decision from the PDP is positive. The protected functions must be configured such that they can only be called by the PEP to ensure that access control is strictly enforced. The delegation procedure is detailed in Algorithm 2, which takes an ACC Γ, policy arguments η, a protected function f, and its arguments χ as inputs.

Algorithm 2. Delegation procedure

Require: $\varepsilon \in \Gamma, \omega \in \Gamma, d \in M^d, f \in F, \omega$
Ensure: Function call result $\mathfrak{R}$ or exception
  $p \leftarrow Binding(\omega, f, \eta)$ ▷ Dynamic argument binding (Algorithm 1)
  if $Call(\varepsilon.Evaluate(p)) = \top$ **then**
    $\mathfrak{R} \leftarrow Call(f(\chi))$
  else
    Throw exception *Unauthorized*
  end if
  return $\mathfrak{R}$

Notably, the protected functions F in Γ are invoked exclusively by PEPs. Thus, each $f \in F$ must be designed to distinguish between subjects and function callers, i.e., the PEPs. For example, in Solidity, the variable *msg.sender* within f actually refers to the address of the PEP, not the requesting subject. This distinction is crucial for maintaining accurate and secure access control logic within the protected functions.

5 Conclusion

In this paper, we introduced OnDPE, a pioneering policy evaluation mechanism offering flexible granularity and upgradability and poised to function as the infrastructure for all types of on-chain access control systems. OnDPE facilitates the dynamic interpretation of ACPs during runtime to eliminate the need for additional contract deployments for administrative operations such as policy management. Concurrently, we developed OnDAC, a decentralized access control framework integrating OnDPE as its PDP. OnDAC also features an authentication layer grounded in a decentralized identity model and adopts a multi-signature paradigm. The design of OnDAC follows standardized terminology to bolster reproducibility and refinement efforts. Moving forward, our focus will center on optimizing the implementation of OnDPE to reduce gas consumption stemming from the evaluation engine. Additionally, we aim to introduce formal verification techniques to certify the implementation based on its semantics and ensure its correctness and robustness in practice.

References

1. Anderson, A., et al.: extensible access control markup language (xacml) version 1.0. Oasis (2003)
2. Belchior, R., Putz, B., Pernul, G., Correia, M., Vasconcelos, A., Guerreiro, S.: SSIBAC: self-sovereign identity based access control. In: 2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), pp. 1935–1943. IEEE (2020)

3. Chatterjee, A., Pitroda, Y., Parmar, M.: Dynamic role-based access control for decentralized applications. In: Chen, Z., Cui, L., Palanisamy, B., Zhang, L.-J. (eds.) ICBC 2020. LNCS, vol. 12404, pp. 185–197. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-59638-5_13
4. Dammak, M., Senouci, S.M., Messous, M.A., Elhdhili, M.H., Gransart, C.: Decentralized lightweight group key management for dynamic access control in IoT environments. IEEE Trans. Netw. Serv. Manag. **17**(3), 1742–1757 (2020), ISBN: 1932-4537
5. Ding, Y., Gervais, A., Wattenhofer, R., Sato, H.: Hunting DeFi vulnerabilities via context-sensitive concolic verification. In: Proceedings of the 2024 IEEE/ACM 46th International Conference on Software Engineering: Companion Proceedings, pp. 324–325. Association for Computing Machinery (2024)
6. Ding, Y., Sato, H.: Derepo: a distributed privacy-preserving data repository with decentralized access control for smart health. In: 2020 7th IEEE International Conference on Cyber Security and Cloud Computing (CSCloud)/2020 6th IEEE International Conference on Edge Computing and Scalable Cloud (EdgeCom), pp. 29–35. IEEE, New York, NY, USA (2020)
7. Ding, Y., Sato, H.: Self-sovereign identity as a service: architecture in practice. In: 2022 IEEE 46th Annual Computers. Software, and Applications Conference (COMPSAC), pp. 1536–1543. IEEE, Los Alamitos, CA, USA (2022)
8. Ding, Y., Sato, H.: Bloccess: enabling fine-grained access control based on blockchain. J. Netw. Syst. Manage. **31**(1), 1–34 (2023)
9. Ding, Y., Sato, H.: Model-driven security analysis of self-sovereign identity systems. In: 2023 IEEE 22nd International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), pp. 1687–1694. IEEE (2023)
10. Ding, Y., Sato, H., Machizawa, M.G.: Leveraging self-sovereign identity in decentralized data aggregation. In: 2022 Ninth International Conference on Software Defined Systems (SDS), pp. 1–8. IEEE, Paris, France (2022)
11. Ding, Y., Yu, J., Li, S., Sato, H., Machizawa, M.G.: Data aggregation management with self-sovereign identity in decentralized networks. IEEE Trans. Netw. Serv. Manag. (2024)
12. Ghaleb, A., Rubin, J., Pattabiraman, K.: Achecker: statically detecting smart contract access control vulnerabilities. In: 2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE), pp. 945–956. IEEE (2023)
13. Maesa, D.D.F., Mori, P., Ricci, L.: A blockchain based approach for the definition of auditable access control systems. Comput. Sec. **84**, 93–119 (2019), ISBN: 0167-4048
14. Reed, D., et al.: Decentralized identifiers (dids) v1. 0. Draft Community Group Report (2020)
15. Saini, A., Zhu, Q., Singh, N., Xiang, Y., Gao, L., Zhang, Y.: A smart-contract-based access control framework for cloud smart healthcare system. IEEE Internet of Things J.**8**(7), 5914–5925 (2020), ISBN: 2327-4662
16. Shafagh, H., Burkhalter, L., Ratnasamy, S., Hithnawi, A.: Droplet: decentralized authorization and access control for encrypted data streams. In: 29th USENIX Security Symposium (USENIX Security 20), pp. 2469–2486 (2020)
17. Tseitin, G.S.: On the complexity of derivation in propositional calculus. Automation Reasoning: 2: Classical Papers On Computational Logic 1967–1970, pp. 466–483 (1983), ISBN: 3642819575
18. Wood, G.: Ethereum: a secure decentralised generalised transaction ledger. Ethereum Project Yellow Paper **151**(2014), 1–32 (2014)